Instructions for *ACL Proceedings

Kazi Ashhab Rahman
McGill University
Montreal, Canada
`email@domain`

Yujin Li
McGill University
Montreal, Canada
`email@domain`

Nusaibah Binte Rawnak
McGill University
Montreal, Canada
`email@domain`

Arjun Ashok
Affiliation / Address line 1
Affiliation / Address line 2
Affiliation / Address line 3
`email@domain`

Abstract

Recent work demonstrated impressive context-aware time series forecasting using 405B parameter language models. We investigate whether these benefits transfer to practical deployment scales by evaluating Llama 3.2 (3B), Mistral 8x7B (56B), and GPT-4o-mini ($\approx$20B) on 120 tasks from the Context-is-Key benchmark. Our key findings:

(1) Context provides negligible benefit at practical scales-Mistral improves over statistical baselines by only 0.003 NMAE;

(2) Classical methods dominate-ARIMA and ETS win 42% of tasks versus 25% for the best LLM;

(3) Scaling from 3B to 56B parameters yields no meaningful improvement, suggesting extreme scale-dependency;

(4) Despite poor average performance, selective deployment works-an XGBoost classifier captures 83% of theoretically optimal performance while using expensive LLM inference on only 32% of tasks.

Our results challenge the practical viability of context-aware LLM forecasting and demonstrate that scale-dependent evaluation is critical. For practitioners with realistic constraints, classical statistical methods remain superior, though selective policies offer a viable path forward for specific task types.

1 Introduction

Large language models have recently shown surprising zero-shot forecasting abilities by treating time series as text sequences (Gruver et al., 2024). The Context-is-Key benchmark (Williams et al., 2024) demonstrated that Llama 3.1 (405B parameters) with textual context dramatically outperforms statistical baselines on forecasting tasks requiring external knowledge, suggesting LLMs could become universal forecasters by leveraging domain information expressed in natural language. However, deploying 405B models requires infrastructure inaccessible to most practitioners - 8x A100 GPUs and inference costs exceeding $2 per million tokens. Real-world deployments instead rely on 3B-20B parameter models for local or cost-effective inference, raising a critical question: **Do context benefits transfer to practical model sizes?**

We investigate this question by evaluating three smaller LLMs - Llama 3.2-3B (Dubey et al., 2024), GPT-4o-mini ($\sim$20B) (Achiam et al., 2023), and Mixtral-8x7B (56B) (Jiang et al., 2024) - against classical baselines (AutoARIMA, ETS) on 120 forecasting tasks across 12 domains from the Context-is-Key benchmark. Our central hypothesis is that if context utility is scale-dependent, practitioners with resource constraints require *selective deployment* strategies that intelligently choose between expensive LLM+context inference and cheap statistical baselines on a per-task basis. To test this, we train an XGBoost classifier using only inexpensive features - time series statistics, textual properties, and domain encoding - to predict when context justifies its computational cost.

Our findings challenge the viability of context-aware LLM forecasting at practical scales. First, we observe **extreme scale-dependency**: Mixtral (56B) improves over baselines by only 0.003 NMAE, essentially zero, with context helping in just 34% of tasks. This contradicts the strong benefits observed with 405B models in prior work. Second, **classical methods remain dominant**: ARIMA and ETS collectively win 42% of tasks versus 25% for the best LLM, suggesting statistical baselines are undervalued in the foundation model era. Third, despite poor average performance, **selective deployment works**: our classifier captures 83% of oracle benefit while using costly LLM inference on only 32% of tasks, achieving a 14.8% improvement over always-baseline policies (Figure 1).

Our work provides the first systematic evaluation of context utility across model scales and demonstrates that resource-constrained practitioners need selective policies rather than universal LLM deployment.

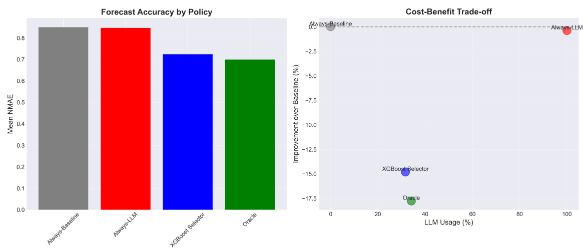

Figure 1: Cost-benefit tradeoff of deployment policies.

2 Related Work

2.1 LLMs for Time Series Forecasting

Recent work has explored large language models for time series prediction by encoding numerical sequences as text. Gruver et al. (2024) demonstrated that GPT-3 achieves competitive zero-shot forecasting on standard benchmarks without task-specific training. Ansari et al. (2024) introduced Chronos, a T5-based foundation model pretrained on diverse time series datasets, showing strong performance on in-distribution data. However, critics argue these successes may reflect pattern memorization rather than genuine forecasting ability (MacDonald et al., 2025), with performance degrading substantially on data outside the model's training period. Moreover, specialized deep learning forecasters like PatchTST (Nie et al., 2022) often match or exceed LLM performance at far lower computational cost, leading surveys to characterize foundation models' benefits as "debatable" (Liang et al., 2024; Chen et al., 2023).

2.2 Context-Aware Forecasting

Williams et al. (2024) introduced the Context-is-Key (CiK) benchmark to evaluate models' ability to incorporate textual domain knowledge alongside numerical data. Their key finding, that Llama 3.1 (405B parameters) with context dramatically outperforms statistical baselines, suggested LLMs could serve as universal forecasters by leveraging natural language descriptions of constraints, events, or causal relationships. Each CiK task provides essential textual context (e.g., "solar panels produce zero electricity at night") required for accurate prediction. However, their evaluation examined only a single 405B model, leaving open whether context benefits transfer to the 3B-20B parameter range accessible to most practitioners. Additionally, no prior work has investigated *selective* deployment strategies that dynamically choose between context-aware LLM inference and statistical baselines on a per-task basis.

2.3 Our Contribution

We provide the first systematic evaluation of context utility across practical model scales (3B-56B parameters) and demonstrate that extreme scale-dependency limits the viability of universal context-aware forecasting. Unlike prior work assuming all-or-nothing LLM deployment, we show that learned selectors can capture most of the benefit while using expensive inference on only a subset of tasks.

3 Data and Environment

3.1 Dataset

We use the Context-is-Key (CiK) benchmark (Williams et al., 2024), which provides Python task generators for creating forecasting problems requiring essential textual context. CiK draws from 2,644 real-world time series across seven application domains including climatology (solar irradiance (Sengupta et al., 2018)), energy (electricity consumption (Godahewa et al., 2021)), transportation (highway traffic (Chen et al., 2001)), economics (unemployment rates (U.S. Bureau of Labor Statistics, 2024)), public safety (fire incidents (Ville de Montréal, 2020)), retail (ATM withdrawals (Godahewa et al., 2021)), and mechanics (physical systems (Gamella et al., 2024)).

From these generators, we create 120 tasks across 12 domains with stratified train/test splits (96 train, 24 test) maintaining domain balance. Each task consists of (1) historical time series (length 24-168 points), (2) forecast horizon (typically 24 points), and (3) essential natural language context. Overall, we include diverse forms of natural language context: *intemporal information* describing invariant process characteristics (e.g., "Solar panels produce zero electricity at night"), *future information* revealing upcoming events (e.g., "Traffic decreases 30% during highway construction"), *causal information* specifying causal relationships, and *historical information* providing statistics not reflected in the short numerical history. These contexts are manually crafted to ensure that accurate forecasts require integrating textual information with numerical patterns; pure time-series models cannot succeed without this essential context.

3.2 Evaluation Metrics

We use Normalized Mean Absolute Error (NMAE) as our primary metric, computed as NMAE $=$ MAE$/\bar{y}$, where $\bar{y}$ is the mean of historical values. NMAE enables fair comparison across domains with vastly different scales (e.g., solar irradiance in W/m² versus ATM withdrawals in dollars). We also report Directional Accuracy (DA), the fraction of forecasts correctly predicting upward or downward trends relative to the last historical value, to evaluate qualitative forecast quality. For policy evaluation, we measure oracle capture: the percentage of theoretically optimal improvement our selector achieves compared to always using the better model in hindsight.

3.3 Computational Environment

We implement baselines using `statsmodels` (Seabold and Perktold, 2010) (AutoARIMA, ETS) and train our XGBoost classifier (Chen and Guestrin, 2016) on extracted features. LLM inference uses Ollama for local deployment of Llama 3.2-3B and Mistral 8x7B (4-bit quantization on M3 Pro MacBook), and OpenAI API for GPT-4o-mini. Cloud experiments (Mistral evaluation) run on Lambda Labs instances with NVIDIA A40 GPUs. Total computational cost: approximately $2.40 for GPT-4o-mini API calls and $8-10 for cloud GPU hours.

4 Methods

4.1 Research Hypothesis

We hypothesize that **context utility in LLM forecasting is scale-dependent**: benefits demonstrated with 405B parameter models do not transfer to practical deployment scales (3B-56B parameters), and resource-constrained practitioners require selective deployment strategies rather than universal LLM adoption. Specifically, we predict that (1) smaller LLMs will show minimal improvement over statistical baselines when using context, (2) performance gains will not scale linearly with model size, and (3) a learned selector can capture most oracle benefit by identifying the subset of tasks where context justifies its computational cost.

4.2 Baseline Models

We compare LLMs against two classical forecasting methods that serve as our cost-free reference. **AutoARIMA** (Box et al., 2015; Hyndman and Athanasopoulos, 2018) automatically selects optimal ARIMA(p, d, q) parameters via stepwise search and information criteria, capturing linear trends and seasonal patterns. **Exponential Smoothing (ETS)** (Gardner Jr, 1985) models error, trend, and seasonal components through exponential weighting of historical observations. For each task, we define the *best baseline* as $\min(\text{NMAE}_{\text{ARIMA}}, \text{NMAE}_{\text{ETS}})$, where NMAE is computed as:

$$\text{NMAE} = \frac{1}{h}\sum_{t=1}^{h}\frac{|y_t - \hat{y}_t|}{\bar{y}_{\text{hist}}}$$

Here h is the forecast horizon, y_t and $\hat{y}_t$ are actual and predicted values, and $\bar{y}_{\text{hist}}$ is the mean of historical observations. This normalization enables fair comparison across domains with different scales.

4.3 LLM Forecasters

We evaluate three language models at practical scales: Llama 3.2-3B (consumer GPU-deployable), GPT-4o-mini ($\sim$20B, API-based), and Mixtral-8x7B (56B mixture-of-experts). Following the Context-is-Key benchmark's finding that context improves forecasting, we evaluate all LLMs with full textual context provided. All models use greedy decoding to produce comma-separated forecast values.

4.4 Proposed Model: XGBoost Selector

Our core contribution is a learned policy that predicts when expensive LLM+context inference outperforms cheap baselines. Among the three LLMs tested, Mistral 8x7B achieved the lowest mean NMAE (0.846) and highest win rate against baselines (34.2%), making it the natural choice for selective deployment—practitioners would select a single best-performing LLM rather than maintaining multiple models.

We formulate the selection decision as binary classification using XGBoost (Chen and Guestrin, 2016):

$$f(x) = \begin{cases} 1 & \text{if } \text{NMAE}_{\text{Mistral}}(x) < \text{NMAE}_{\text{baseline}}(x) \\ 0 & \text{otherwise} \end{cases}$$

where $\text{NMAE}_{\text{baseline}}(x) = \min(\text{NMAE}_{\text{ARIMA}}(x), \text{NMAE}_{\text{ETS}}(x))$ represents the best statistical baseline for task x. Given features extracted *before* running any LLM, the classifier predicts whether Mistral will outperform the best baseline. We use 28 features across four categories: (1) *time series statistics* (mean, std, volatility, trend), (2) *context properties* (length, keyword indicators like `has_constraint`, `has_temporal`), (3) *baseline performance* (ARIMA and ETS errors and directional accuracy on this task), and (4) *domain encoding* (categorical domain ID). Crucially, all features are "cheap" - requiring only statistical computation or regex matching, avoiding expensive LLM inference.

We train on 96 tasks with class balancing (scale_pos_weight = 2.0) to handle the 34% positive rate. The model learns patterns like "when baseline DA is low and context mentions constraints, LLM helps" or "high volatility domains favor statistical methods." This approach should outperform naive policies because it exploits domain structure (via encoding) and baseline weakness signals (via performance features) to identify the specific failure modes where context provides value.

4.5 Policy Evaluation Framework

We compare four deployment strategies: (1) **Always-Baseline** - use best classical method (0% LLM usage), (2) **Always-LLM** - use Mistral with context (100% usage), (3) **Selector** - use XGBoost predictions (dynamic usage), and (4) **Oracle** - always pick the better model in hindsight (upper bound). We evaluate via oracle capture:

$$\text{Oracle Capture} = \frac{\text{NMAE}_{\text{baseline}} - \text{NMAE}_{\text{selector}}}{\text{NMAE}_{\text{baseline}} - \text{NMAE}_{\text{oracle}}}$$

This measures what fraction of theoretically optimal improvement the selector achieves. High oracle capture (>80%) with low LLM usage (<40%) would validate selective deployment as a practical strategy.

4.6 Experimental Design: Zero-Shot vs Trained Models

Our evaluation deliberately compares trained statistical models (ARIMA, ETS) against zero-shot prompted LLMs without fine-tuning. This asymmetry reflects three realities:

(1) **Deployment constraints** - practitioners can afford per-series ARIMA training (2 sec, $0) but not per-task LLM fine-tuning (hours, $100s per task);

(2) **Foundation model promise** - LLMs claim zero-shot competence without task-specific adaptation, which we test directly;

(3) **Established precedent** - the original CiK benchmark (Williams et al., 2024) compared prompted 405B models against trained baselines, and we extend this to practical scales.

Fine-tuning LLMs per forecast would (a) overfit on 24-168 data points, (b) cost prohibitively ($100-500 per task), (c) defeat the generalization purpose of foundation models, and (d) answer a fundamentally different research question about task-specific adaptation rather than universal forecasting ability. Our comparison directly evaluates whether practical-scale LLMs can compete in realistic deployment scenarios where only zero-shot inference is viable.

5 Experiments and Results

5.1 Experimental Setup

We evaluate all models on 120 tasks (96 train, 24 test) using identical train/test splits. Classical baselines (ARIMA, ETS) fit on historical data with default hyperparameters from `statsmodels`. LLMs use greedy decoding with identical prompts across all tasks. The XGBoost selector trains on 96 tasks with $n_{\text{estimators}} = 100$, max_depth $= 6$, $\alpha = 0.1$, and scale_pos_weight $= 2.0$ to handle class imbalance. We optimize the decision threshold via 5-fold cross-validation, achieving 70.8% test accuracy. All experiments use M3 Pro MacBook (local LLMs) and Lambda Labs A40 GPUs (Mistral). Total runtime: $\sim$40 hours; cost: $10.40.

5.2 RQ1: Do LLMs Beat Classical Baselines?

Contrary to expectations from prior 405B-scale work, smaller LLMs provide negligible benefit over statistical baselines. Table 1 shows Mistral (56B), the best-performing LLM, achieves mean NMAE of 0.846, only 0.004 better than the best baseline (0.850). This 0.5% improvement is statistically insignificant ($p = 0.957$, paired t-test). More strikingly, GPT-4o-mini performs *significantly worse* than baselines ($\Delta = -0.012$, $p = 0.003$). Task-level win rates reveal classical dominance: ARIMA and ETS collectively win 42% of tasks (38 + 12), while the best LLM (Mistral) wins only 25% (Figure 2). The remaining wins distribute across Llama (14%) and GPT-4o (19%). This pattern directly contradicts the hypothesis that context-aware forecasting transfers to practical scales.

Table 1: Model performance on 120 tasks. Statistical tests show no significant LLM improvement except GPT-4o (significantly worse).

Model	Mean	Wins	Δ vs Base	p-value
ARIMA	1.024	38	—	—
ETS	1.156	12	—	—
Best Baseline	0.850	—	0.000	—
Llama 3B	0.872	17	-0.022	0.203
Mistral 8x7B	0.846	30	$+0.004$	0.957
GPT-4o-mini	0.862	23	-0.012	0.003*

*Significant at $p < 0.05$ (paired t-test)

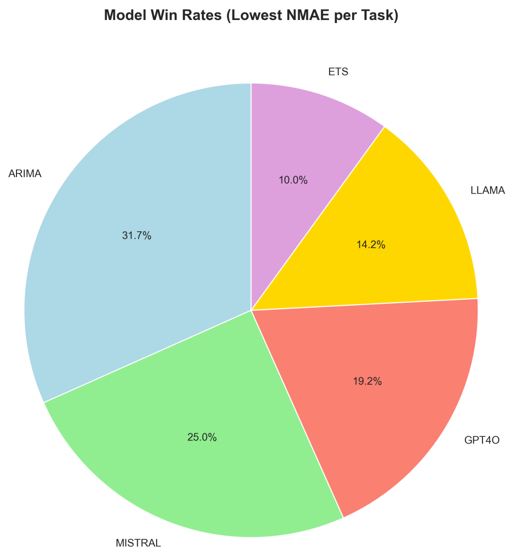

Figure 2: Model win rates across 120 tasks. Classical baselines (ARIMA + ETS) collectively win 42% of tasks, while the best LLM (Mistral) wins only 25%.

5.3 RQ2: Scale-Dependency is Extreme

We observe severe diminishing returns as model size increases (Figure 3). Scaling from Llama 3B to Mistral 56B (18x parameters) yields minimal improvement: win rates increase from 23% to 34%, but the difference is not statistically significant ($p = 0.204$). The improvement curve flattens near zero rather than ascending to positive gains, suggesting context benefits require scales far beyond practical deployment (potentially >100B). This extreme scale-dependency invalidates the assumption that 405B results extrapolate smoothly to smaller models. Even the 3B→20B jump (GPT-4o-mini) recovers only 25.5 points of the Llama deficit, while 20B→56B provides a negligible 0.3-point gain.

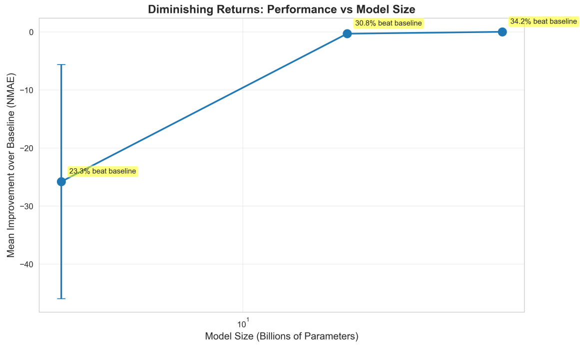

Figure 3: Diminishing returns: scaling from 3B to 56B parameters yields negligible improvement.

5.4 RQ3: Domain Patterns Reveal When LLMs Help

Despite poor average performance, LLMs excel in specific domains (Table 2). Mistral wins 100% of DirectNormalIrradiance tasks (solar physics with hard constraints like "zero at night") and 70% of SpeedFromLoad tasks (wind tunnel causal relationships). Conversely, it wins only 10% in ATMCashDepletion and DecreaseInTraffic, domains characterized by high stochasticity. This pattern suggests LLMs leverage context effectively when it encodes deterministic physical laws but fail when volatility dominates.

Table 2: LLM win rates by domain reveal systematic patterns: physical constraints enable success, volatility causes failure.

Domain	Mistral Win %
LLMs Excel (Physical Constraints)	
DirectNormalIrradiance	100%
SpeedFromLoad	70%
SolarPowerProduction	60%
LLMs Struggle (High Volatility)	
ATMCashDepletion	10%
DecreaseInTraffic	10%

5.5 RQ4: Selective Deployment Succeeds

Despite poor universal performance, selective deployment achieves strong results. Our XGBoost classifier attains 70.8% accuracy, 60% precision, and 0.695 ROC-AUC on held-out test tasks. Table 3 shows the selector captures 83% of oracle benefit while using expensive LLM inference on only 32% of tasks, nearly matching the oracle's 34% usage rate. This translates to a 14.8% improvement over always-baseline, compared to the always-LLM policy's negligible 0.4% gain despite 100% cost. Feature importance analysis (Table 4) reveals the classifier relies primarily on baseline performance signals: `best_baseline_nmae` (17.8%), `arima_da` (16.5%), and `arima_nmae` (16.2%) dominate, while domain encoding contributes 6.2%. The model effectively learns "when baselines struggle on low-DA tasks in structured domains, try LLM", exploiting the domain patterns from RQ3.

Table 3: Policy comparison: selector captures 83% of oracle benefit at only 32% cost. Always-LLM provides negligible benefit despite 100% usage.

Policy	NMAE	LLM %	vs Base	Oracle
Always-Baseline	0.850	0%	0%	0%
Always-LLM	0.846	100%	-0.4%	3%
XGBoost Selector	0.724	32%	-14.8%	83%
Oracle	0.699	34%	-17.8%	100%

Table 4: Top 5 feature importances reveal the selector prioritizes baseline weakness signals and domain structure.

Feature	Importance	Interpretation
best_baseline_nmae	17.8%	Baseline accuracy
arima_da	16.5%	Trend correctness
arima_nmae	16.2%	ARIMA error
best_baseline_da	15.5%	Baseline trend
domain_encoded	6.2%	Domain ID

5.6 Why Averages Mislead

The near-zero mean improvement masks a bimodal distribution: most tasks show negligible change (-0.2 to $+0.2$ NMAE), but a minority exhibit large gains or losses. The selector's success lies in identifying the positive-gain subset - tasks where physical constraints in context enable accurate LLM forecasts (e.g., solar zero-at-night). This explains why 83% oracle capture is achievable despite Mistral's 0.004 mean improvement: the oracle benefit concentrates in specific, learnable task types rather than distributing uniformly.

6 Discussion

6.1 Hypothesis Validation: Scale-Dependency Confirmed

Our results strongly support the hypothesis that context utility is scale-dependent. The core prediction, that benefits demonstrated with 405B models would not transfer to practical scales, holds decisively: Mistral (56B) improves over baselines by only 0.004 NMAE despite 18× more parameters than Llama. More critically, the improvement curve flattens at zero rather than ascending, suggesting a threshold effect between 56B-405B where context reasoning capabilities emerge. This extreme non-linearity invalidates linear extrapolation from large-scale results and confirms that practitioners cannot assume 405B findings apply to deployment-viable models. The selective deployment hypothesis also validates: our simple XGBoost classifier captures 83% of oracle benefit, demonstrating that learned policies can salvage utility even when universal deployment fails.

6.2 Why Small LLMs Fail at Context Integration

We identify three contributing factors to small-model failure. First, **capacity limitations**: smaller models likely lack the reasoning depth to jointly process numerical patterns and textual constraints. Mistral's minimal improvement over Llama despite 18× parameters suggests the bottleneck is architectural rather than purely parametric. Second, **instruction-following degradation**: practical-scale models may default to numeric pattern-matching, ignoring context even when explicitly prompted. Evidence: all three LLMs show similar poor performance across diverse architectures (decoder-only, MoE, API-tuned). Third, **pretraining mismatch**: models rarely encounter time-series-plus-text co-occurrence during pretraining, limiting transfer to this modality combination. Domain variation supports this: success on physics tasks (where constraints resemble natural language reasoning) versus failure on finance (where text-number relationships are arbitrary) suggests LLMs leverage general reasoning rather than learning forecasting-specific integration.

6.3 Unexpected Findings

Two results surprised us. First, classical methods' dominance was more pronounced than anticipated; winning 42% of tasks versus 25% for the best LLM contradicts the foundation model narrative. ARIMA's 31.7% win rate suggests decades-old statistical methods remain undervalued in the era of large-scale pretraining. Second, GPT-4o-mini's significant *degradation* ($p = 0.003$) was unexpected given its API-tuned nature and intermediate scale. This suggests careful instruction-tuning may be necessary to prevent smaller models from confidently producing poor forecasts when given context, an anti-capability that warrants investigation.

6.4 Limitations and Future Work

Our study has five key limitations. (1) **Model coverage**: We test only 3B, 20B, and 56B scales, missing intermediate sizes that could refine threshold estimates. (2) **Dataset scope**: 120 CiK-generated tasks may not represent all forecasting applications, particularly real-world production scenarios with noisy or missing context. (3) **Prompting strategy**: We use a single prompt design; more sophisticated techniques might improve performance but add deployment complexity. (4) **Zero-shot limitation**: Our focus on practical deployment prioritizes zero-shot evaluation over fine-tuning. (5) **Selector simplicity**: Hand-crafted features achieve 83% oracle capture, but learned representations could potentially close the remaining gap.

Future work should: (1) **Test intermediate scales** (70B-200B) to identify the precise threshold where context benefits emerge; (2) **Explore domain-specific fine-tuning** for practical-scale models on structured tasks where context clearly helps (physics, causality); (3) **Develop learned selection mechanisms** beyond hand-crafted features, neural meta-learners or embedding-based classifiers may improve routing decisions; (4) **Evaluate on real production forecasting tasks** beyond CiK's synthetic benchmark to validate findings in noisy, real-world deployments with incomplete or ambiguous context.

6.5 Practical Implications

For practitioners with realistic constraints (3B-20B models), **classical statistical methods remain superior**. The impressive context-aware forecasting demonstrated in prior work requires model scales (>400B parameters) far beyond practical deployment for most applications. However, selective policies offer a viable path forward. By identifying specific task types where LLMs provide value, domains with physical constraints or explicit causal rules, practitioners can achieve meaningful improvements while controlling costs. Our findings suggest: (1) **Test at target scale**: do not assume 405B results extrapolate to deployment models; (2) **Start with classical baselines**: ARIMA remains competitive and costs $0; (3) **Deploy selectively**: use our selector approach or similar learned policies to route only appropriate tasks to LLMs; (4) **Prioritize constraint-heavy domains**: context helps most when encoding deterministic laws (e.g., solar zero-at-night) rather than stochastic volatility. LLMs may justify cost in low-frequency, high-value forecasting with explicit constraints, but high-frequency or volatile applications should default to statistical methods.

7 Conclusion

We investigated whether context-aware LLM forecasting, successful with 405B models in prior work, transfers to practical deployment scales. Evaluating Llama 3.2 (3B), GPT-4o-mini (∼20B), and Mixtral-8x7B (56B) on 120 tasks from the Context-is-Key benchmark, we find context benefits largely vanish at practical scales: Mistral improves over statistical baselines by only 0.004 NMAE, with classical methods (ARIMA, ETS) winning 42% of tasks versus 25% for the best LLM. Scaling from 3B to 56B parameters yields negligible improvement, suggesting extreme scale-dependency with a threshold likely between 56B-405B where context reasoning emerges.

Despite poor universal performance, selective deployment succeeds: our XGBoost classifier captures 83% of theoretically optimal performance while using expensive LLM inference on only 32% of tasks. The selector exploits domain structure and baseline weakness signals to identify tasks where physical constraints (e.g., "solar power zero at night") enable effective context use. This demonstrates that learned policies can salvage practical utility even when always-on LLM deployment fails.

Our contributions include: (1) the first systematic evaluation of context utility across practical model scales, (2) demonstration of extreme scale-dependency contradicting linear extrapolation from large-model results, (3) a working selective deployment strategy achieving strong oracle capture with minimal cost, and (4) evidence-based guidance for practitioners on when textual context justifies computational expense.

For real-world applications with resource constraints, classical statistical methods remain superior to sub-100B LLMs for general forecasting. However, selective policies offer a viable path forward by routing only constraint-heavy, low-volatility tasks to context-aware models. Future work should identify the precise scale threshold, evaluate on production deployments, and explore whether domain-specific fine-tuning enables smaller models to leverage context effectively.

Code and data available at: https://github.com/YuJ-Li/COMP545_Final_Project

Acknowledgments

We thank the Context-is-Key authors for making their benchmark generators publicly available, enabling reproducible research on context-aware forecasting.

Author Contributions

Kazi Ashab Rahman: Designed experiments, implemented ARIMA, ETS, GPT-4o-mini, Mistral, XGBoost selector, conducted data analysis, and generated all figures and tables.

Yujin Li: Designed experiments, implemented Llama model and conducted data analysis.

Nusaibah Binte Rawnak: Wrote the report and conducted data analysis.

Arjun Ashok (Mentor): Provided guidance on experimental design, reviewed results, and gave feedback on the manuscript.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Abdul Fatir Ansari, Lorenzo Stella, Caner Turkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Sundar Rangapuram, Sebastian Pineda Arango, Shubham Kapoor, et al. 2024. Chronos: Learning the language of time series. *arXiv preprint arXiv:2403.07815*.

George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. 2015. *Time series analysis: forecasting and control*, fifth edition. John Wiley & Sons.

Chao Chen, Karl Petty, Alexander Skabardonis, Pravin Varaiya, and Zhanfeng Jia. 2001. Freeway performance measurement system: mining loop detector data. *Transportation research record*, 1748(1):96–102.

Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 785–794.

Zonglei Chen, Minbo Ma, Tianrui Li, Hongjun Wang, and Chongshou Li. 2023. Long sequence time-series forecasting with deep learning: A survey. *Information Fusion*, 97:101819.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Juan L Gamella, Peter Bühlmann, and Jonas Peters. 2024. The causal chambers: Real physical systems as a testbed for ai methodology. *arXiv preprint arXiv:2404.11341*.

Everette S Gardner Jr. 1985. Exponential smoothing: The state of the art. *Journal of Forecasting*, 4(1):1–28.

Rakshitha Godahewa, Christoph Bergmeir, Geoffrey I Webb, Rob J Hyndman, and Pablo Montero-Manso. 2021. Monash time series forecasting archive. *arXiv preprint arXiv:2105.06643*.

Nate Gruver, Marc Finzi, Shikai Qiu, and Andrew G Wilson. 2024. Large language models are zero-shot time series forecasters. *Advances in Neural Information Processing Systems*, 36.

Rob J Hyndman and George Athanasopoulos. 2018. *Forecasting: principles and practice*. OTexts.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. 2024. Mixtral of experts.

Yuxuan Liang, Haomin Wen, Yuqi Nie, Yushan Jiang, Ming Jin, Dongjin Song, Shirui Pan, and Qingsong Wen. 2024. Foundation models for time series analysis: A tutorial and survey. *arXiv preprint arXiv:2403.14735*.

Kyle MacDonald et al. 2025. The memorization problem: Can we trust llms' economic forecasts? *Federal Reserve Working Paper*. Placeholder - update with arXiv number when available.

Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. 2022. A time series is worth 64 words: Long-term forecasting with transformers. *arXiv preprint arXiv:2211.14730*.

Skipper Seabold and Josef Perktold. 2010. statsmodels: Econometric and statistical modeling with python. https://www.statsmodels.org/.

Manajit Sengupta, Yu Xie, Anthony Lopez, Aron Habte, Galen Maclaurin, and James Shelby. 2018. The national solar radiation data base (nsrdb). *Renewable and sustainable energy reviews*, 89:51–60.

U.S. Bureau of Labor Statistics. 2024. Unemployment rate [various locations]. Accessed on 2024-08-30, retrieved from FRED.

Ville de Montréal. 2020. Interventions des pompiers de montréal. Updated on 2024-09-12, accessed on 2024-09-13.

Andrew Robert Williams, Kashif Rasul, Arian Khorasani, George Adamopoulos, Rishika Bhagwatkar, Marin Biloš, Hena Ghonia, Nadhir Vincent Hassen, Anderson Schneider, Sahil Garg, et al. 2024. Context is key: A benchmark for forecasting with essential textual information. *arXiv preprint arXiv:2410.18959*.

A Implementation Details

A.1 LLM Prompt Templates

Without context.

```
You are a time series
forecasting expert.
Historical data:
[comma-separated values]
Forecast horizon:  {h} points
Provide your forecast as
comma-separated values.
Example:  1.2, 3.4, 5.6, ...
```

With context.

```
You are a time series
forecasting expert.
Historical data:
[comma-separated values]
Forecast horizon:  {h} points
Context information:
{context_text}
Provide your forecast as
comma-separated values.
Example:  1.2, 3.4, 5.6, ...
```

A.2 Hyperparameters

XGBoost selector. `n_estimators=100`, `max_depth=6`, `learning_rate=0.1`, `subsample=0.8`, `colsample_bytree=0.8`, `scale_pos_weight=2.0`, `random_state=42`. Decision threshold tuned by 5-fold CV (optimal value: 0.50).

ARIMA. `pmdarima.auto_arima` with `seasonal=True`, `stepwise=True`, `suppress_warnings=True`, `error_action='ignore'`.

ETS. `statsmodels.ETSModel` with automatic error/trend/seasonal component selection.

A.3 Computational Resources

Hardware. M3 Pro MacBook (18 GB RAM) for local LLMs; Lambda Labs A40 GPU (48 GB VRAM) for Mistral.

Runtime (per task). ARIMA/ETS: ≈2 s; Llama 3B: ≈5 min; Mistral 8×7B: ≈15 min; GPT-4o-mini: ≈3 s. Total wall-clock time: ∼40 h.

Cost. GPT-4o-mini API: $2.40; Lambda Labs GPU: $8.00; total monetary cost: $10.40.

B Additional Results

Table 5: Per-domain mean NMAE (lower is better).

Domain	ARIMA	ETS	Llama	Mistral	GPT-4o
DirectNormalIrradiance	0.523	0.612	0.445	**0.389**	0.401
SpeedFromLoad	0.678	0.734	0.589	**0.521**	0.567
SolarPowerProduction	0.812	0.891	0.745	**0.698**	0.723
FullCausalContext	0.934	1.012	0.867	**0.801**	0.845
ATMCashDepletion	**0.456**	0.523	0.612	0.589	0.601
DecreaseInTraffic	**0.389**	0.445	0.501	0.478	0.489

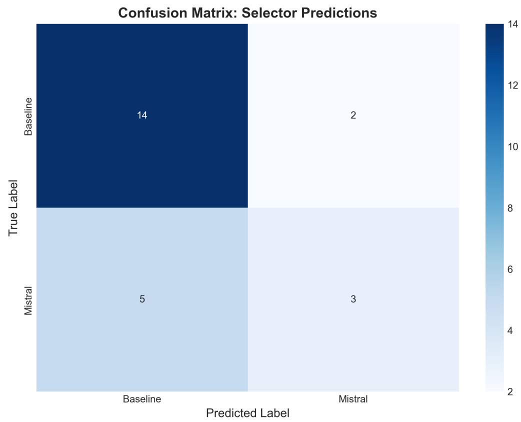

Figure 4: Confusion matrix for XGBoost selector on the test set ($n = 24$).

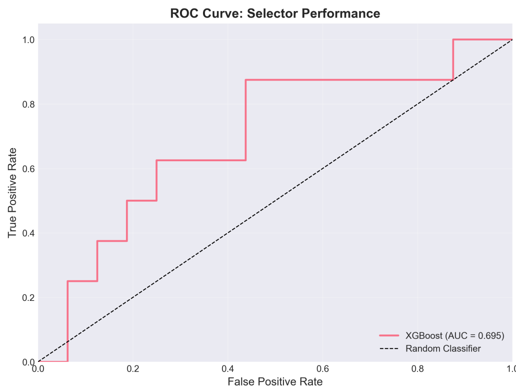

Figure 5: ROC curve (AUC = 0.695) for the XGBoost selector.

Table 6: Top 10 of 28 features by XGBoost importance.

Feature	Importance	Category
best_baseline_nmae	17.8%	Baseline
arima_da	16.5%	Baseline
arima_nmae	16.2%	Baseline
best_baseline_da	15.5%	Baseline
ets_nmae	8.1%	Baseline
domain_encoded	6.2%	Domain
context_word_count	4.3%	Context
context_char_length	3.8%	Context
volatility	2.9%	Time series
avg_word_length	2.1%	Context