LLMForecaster: Improving Seasonal Event Forecasts with Unstructured Textual Data

Hanyu Zhang[*]
Georgia Institute of Technology
hzhang747@gatech.edu

Chuck Arvin[†]
Amazon
chuarvin@amazon.com

Dmitry Efimov
Amazon
defimov@amazon.com

Michael W. Mahoney
Amazon
zmahmich@amazon.com

Dominique Perrault-Joncas
Amazon
joncas@amazon.com

Shankar Ramasubramanian
Amazon
sramasub@amazon.com

Andrew Gordon Wilson
Amazon & NYU
wilsmman@amazon.com

Malcolm Wolff
Amazon
wolfmalc@amazon.com

Abstract

Modern time-series forecasting models often fail to make full use of rich unstructured information about the time series themselves. This lack of proper conditioning can lead to "obvious" model failures; for example, models may be unaware of the details of a particular product, and hence fail to anticipate seasonal surges in customer demand in the lead up to major exogenous events like holidays for clearly relevant products. To address this shortcoming, this paper introduces a novel forecast post-processor — which we call LLMForecaster — that fine-tunes large language models (LLMs) to incorporate unstructured semantic and contextual information and historical data to improve the forecasts from an existing demand forecasting pipeline. In an industry-scale retail application, we demonstrate that our technique yields statistically significantly forecast improvements across several sets of products subject to holiday-driven demand surges.

1 Introduction

Time series forecasting has a broad variety of uses across industry today, including in transportation, weather and retail settings. In modern retail settings, accurate demand forecasts are key to running an efficient supply chain. Improvements in forecast quality directly affect inventory efficiency, reducing stockouts, and enhancing customer satisfaction.

In recent years, deep neural networks have become a powerful tool for forecasting at scale. Deep learning models such as Recurrent Neural Networks (RNNs) [1; 2], Convolutional Neural Networks (CNNs) [3], and attention-mechanisms [4], have shown promising results as they extract complex features and adapt to various time series patterns. These architectures are often designed to learn auto-correlations and cross-correlations from data [5; 6; 7]. These kinds of models have successfully integrated exogenous features (e.g. past covariates, known future information, and static covariates) and achieved remarkable success in real-world problems, including traffic forecasting [8; 9], retail demand prediction [10; 11], power generation prediction [12], and energy consumption modeling [13].

[*] Work done while at Amazon

[†] Corresponding author

NeurIPS 2024 Workshop on Time Series in the Age of Large Models.

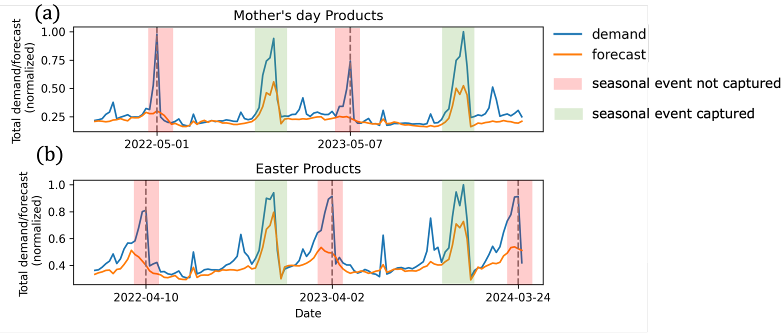

Figure 1: Aggregated demand and forecast for groups of products: (i) Mother's Day products; and (ii) Easter products. The vertical dashed lines mark the week prior to the holiday in question. In green, we show time segments where the production model anticipates event-driven demand surges — specifically large shopping events like Christmas. In red, we show time segments where the production model fails to anticipate event-driven demand surges.

While these models can effectively incorporate numerical or categorical exogenous features, other valuable features like product descriptions or customer reviews exist only as unstructured text. Because this information exists only as unstructured text which is difficult to featurize, these sources of information have been largely neglected or featurized in simple ways [7; 14; 15; 16].

Nonetheless, descriptive and contextual information about the time series may meaningfully enhance forecast quality. In the retail setting, unstructured text describing the use and design of a product contains valuable information about whether this product will see surges in demand for upcoming seasonal events (holidays, back-to-school, etc.). This information is often difficult or impossible to learn from the time series themselves, for several reasons. Many products are new, with perhaps only one (or *fewer*) years of sales history. Sales history at the product level is also noisy, subject to spurious spikes and stockouts which distort the historical sales. These factors mean that the available historical sales are often insufficient to predict upcoming seasonal patterns reliably.

In Figure 1, we illustrate this problem, using forecasts from an MQ-Transformer model [10] trained on a retail dataset. We focus on products which are relevant to two key holiday related seasonal events: Mother's Day and Easter. For both groups of products, we see that the existing forecasting model appropriately anticipates surges in demand during the holiday season (between Black Friday and Christmas). The model responds reasonably to the holiday season (green bands), where we see elevated sales across many products, high customer traffic and numerous price discounts. By contrast, the model fails to anticipate surges in demand during the Mother's Day and Easter holiday periods themselves (red bands). Notably, the demand surges during Mother's Day and Easter are localized to small groups of products, aggregate customer traffic does not spike, and products are infrequently discounted. The failure to anticipate these localized surges in demand increases the risk of stockouts for key products during these events. Today, these defects must be addressed through human intervention, relying on human analysts to use their judgment and knowledge to identify products relevant to an upcoming event and adjust the forecasts accordingly.

Recent advancements in Large Language Models (LLMs) offer a promising avenue to address these challenges, enhancing predictive accuracy by combining rich textual information with traditional covariates. Gruver et al. [17] first showed that even LLMs with text-based pre-training can perform impressive zero-shot time series forecasting. Moreover, pre-trained LLMs have the capability to perform domain-specific predictive tasks by directly querying them with domain-specific instructions and knowledge [18; 19; 20; 21]. Xue et al. [22] extended this approach by generalizing prediction tasks to time series data, incorporating context and semantic information from historical data. LLM4TS[23] utilizes a two-stage fine-tuning process to improve the model's ability to handle time series data, even with limited data availability. Similarly, TEMPO [24] applies a Generative Pre-trained Transformer (GPT) to time series forecasting, using a prompt-based approach that tailors the model to complex temporal patterns and non-stationary data. Most existing research in time series forecasting has focused on using time series data as input to LLMs, exploring methods like tokenization of time series data [7; 15].

Despite these advancements, it remains an open question how to develop LLMs to integrate descriptive information about the time series themselves — for example, the description of the product corresponding to the sales in question. Further, these methods are often stand-alone forecasting models. This can be powerful, but in other real-world use cases we may already have a "good enough" forecasting solution in place. In those cases, we do not want to completely replace the existing solution — instead we only want to modify the forecasts to fix areas where the existing models has clear deficiencies.

Our work addresses both of these areas. Here we introduce a procedure, the *LLMForecaster*, to incorporate unstructured textual information and recommend forecast adjustments to improve the accuracy of an existing forecasting pipeline. Our procedure utilizes fine-tuned LLMs that incorporate both historical forecasts as well as unstructured information. This allows us to systematically improve forecast quality in cases where the existing model fails to anticipate holiday related demand surges. We show that our approach enhances the accuracy of demand forecasts in retail environments, empowering businesses to better manage seasonal fluctuations and optimize their operations.

2 Methodology

We introduce the LLMForecaster, designed to automate forecast adjustments to correct biases in existing demand forecasting models. Here, the existing model is an MQ-Transformer model trained on a large dataset of retail sales. This existing model generates initial predictions, denoted as $f_{i,t}$ for product i and target date t. We then train an additional model, incorporating unstructured text information such as product titles and descriptions ($\mathbf{x}_{i,t,\text{text}}$) and numeric features such as the price or forecast ($\mathbf{x}_{i,t,\text{num}}$) via prompts to an LLM. The model architecture is shown in Figure 2.

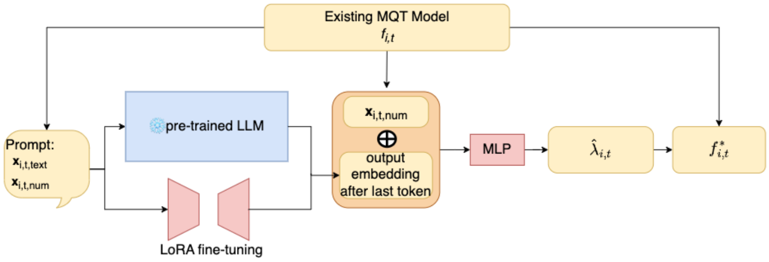
Existing MQT Model
$f_{i,t}$
Prompt:
$\mathbf{x}_{i,t,\text{text}}$
$\mathbf{x}_{i,t,\text{num}}$
pre-trained LLM
LoRA fine-tuning
$\mathbf{x}_{i,t,\text{num}}$
⊕
output embedding after last token
MLP
$\hat{\lambda}_{i,t}$
$f^*_{i,t}$

Figure 2: LLMForecaster incorporates text and numeric information through an LLM to rescale the raw prediction $f_{i,t}$, producing a better forecast $f^*_{i,t}$.

This additional model is trained to predict the scaling factor $\lambda_{i,t} = \log(\frac{y_{i,t}}{f_{i,t}})$, where $y_{i,t}$ represents the actual observed demand. We choose to predict a scaling factor, as it allows us to make improvements on top of an existing primary model, and helps deal with heavy-tailed response data. [25]. The model minimizes the absolute error between the true and predicted scaling factors:

$$\min_{\hat{\lambda}\in\Lambda} \sum_{i=1}^{n} \left|\lambda_{i,t} - \hat{\lambda}(\mathbf{x}_{i,t,\text{text}}, \mathbf{x}_{i,t,\text{num}})\right| \tag{1}$$

where n is the number of samples, and Λ is the model space.

We then use this scaling factor to rescale the original predictions $f_{i,t}$ into the adjusted forecast $f^*_{i,t}$, using the following transformation.

$$f^*_{i,t} = e^{\hat{\lambda}(\mathbf{x}_{i,t,\text{text}}, \mathbf{x}_{i,t,\text{num}})} f_{i,t} \tag{2}$$

The model depends on an input prompt, which contains product information, forecast values, and other contextual information formatting using a predefined template. An example of such a prompt is shown in Appendix A.1. The fine-tuned LLM generates an embedding vector, which is then adapted to the specific forecasting task using Low-Rank Adaptation (LoRA) [26]. This adapted embedding is concatenated with numerical features and fed into a Multi-Layer Perceptron (MLP) head, which outputs the scaling factor $\hat{\lambda}_{i,t}$. By fine-tuning the LLMForecaster on historical forecasts and demand, the model learns to identify patterns in forecast errors and provide accurate adjustments to the primary model's predictions, especially for products with significant holiday demand surges.

3 Experiment Results

We apply the LLMForecaster to refine predictions made by the existing global model, `MQ-Transformer` (`MQT`), at a lead time of 12 weeks. This lead time is selected to provide sufficient time to procure inventory in advance of holiday surges. The various features are processed into a prompt and fed into the pre-trained `MPT7b-Instruct` model[27], which is further fine-tuned for the forecasting task. Performance is evaluated using the weighted p_{50} quantile loss (wQL), defined as: $\text{wQL} = \sum_{i,t} 0.5|f_{i,t} - y_{i,t}|/\sum_{i,t} y_{i,t}$ where i is index of product and t is the forecast target date.

In this experiment, we train a single model capable of calibrating demand predictions across multiple holidays. We focus on five holidays known for strong seasonality: Halloween, Easter, Mother's Day, Father's Day, and Valentine's Day. The training dataset spans 88 weeks, from August 29th, 2021, to April 30th, 2023, including both holiday-related and non-holiday products. A holiday-related product is one with the holiday name in the item name or product description. The test period covers 48 weeks from May 7th, 2023, to March 3rd, 2024, and is divided into five distinct test sets, one for each target holiday. To ensure the LLM knows when events happen, we use a "Holiday-Encoding Prompt" that provides the LLM with the proximity of the target date to the relevant holiday (A.2).

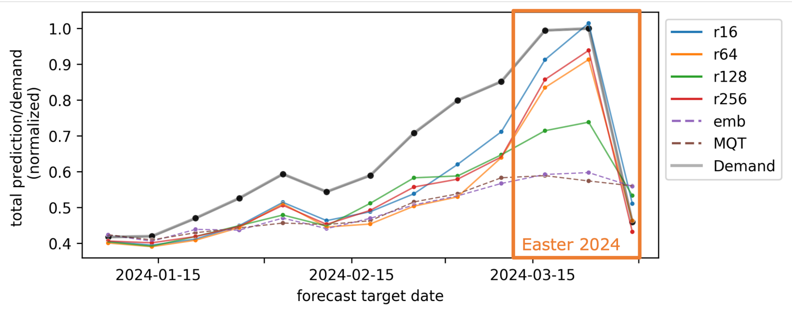

Figure 3: Example of aggregated forecasts on Easter products.

Figure 3 compares the aggregated forecasted demand with actual demand for Easter products. It shows several LLMForecaster models (`r16`, `r64`, `r128`, and `r256`, representing varying LoRA ranks), the `emb` baseline (where we do not apply LoRA fine-tuning) and the original `MQT` baseline. All iterations of the LLMForecaster approach anticipate the Easter demand surge, while our two baselines fail to do so. Table 1 presents wQL improvement results for the 48-week test sets demonstrating that the fine-tuned LLMForecaster models (`r16`, `r64`, `r128`, and `r256`) consistently outperform the baseline `MQT` and `emb` models across all five holiday datasets. We also conduct statistical significance testing of the improvement throughout the year - most of these improvements are statistically significant. More detailed empirical results are available in Appendix A.3, and discussion about Valentine's Day are available in Appendix A.4 .

By contrast, the `emb` model, in which we do not do the LoRA fine-tuning, shows no improvement over the `MQT` baseline. This underscores the importance of the LLMForecaster's sophisticated fine-tuning approach, leveraging LoRA fine-tuning to effectively learn the holiday-specific demand patterns.

Table 1: wQL improvement over the `MQT` baseline (in basis points) for different testsets
Significance Levels: *** $p < 0.001$, ** $p < 0.01$, * $p < 0.05$

Model	Halloween	Easter	Father's Day	Mother's Day	Valentine's Day
`r16`	**105*****	51***	**93*****	68***	**58*****
`r64`	103***	**60*****	88***	**70*****	33
`r128`	77***	32**	39***	10**	18
`r256`	70***	34*	48***	52**	17*
`emb`	8	-5	11**	-14	-16

4 Conclusion and Future Work

We introduced the LLMForecaster, a procedure which incorporates unstructured product-level information into numerical time series forecasts and implements forecast adjustments where they are likely to add value; and we demonstrated that the LLMForecaster model leads to statistically significant improvements to product-level demand forecast in large scale backtests in a retail setting.

In future work, we plan to experiment with a broader variety of prompting techniques, as well as hyperparameter optimization. We are actively exploring similar techniques to use LLMs as a tool to featurize data as inputs to deep learning models, rather than as a post-processor. We will also explore multimodal inputs like product images to further enhance forecast accuracy.

References

[1] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[2] Thomas Fischer and Christopher Krauss. Deep learning with long short-term memory networks for financial market predictions. *European journal of operational research*, 270(2):654–669, 2018.

[3] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

[4] Bryan Lim, Sercan Ö Arık, Nicolas Loeff, and Tomas Pfister. Temporal fusion transformers for interpretable multi-horizon time series forecasting. *International Journal of Forecasting*, 37(4):1748–1764, 2021.

[5] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 11106–11115, 2021.

[6] Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in neural information processing systems*, 34:22419–22430, 2021.

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

[8] Hao Zhang, Yajie Zou, Xiaoxue Yang, and Hang Yang. A temporal fusion transformer for short-term freeway traffic speed multistep prediction. *Neurocomputing*, 500:329–340, 2022.

[9] Yuxuan Zhou. Temporal fusion transformers model for traffic flow prediction. In *Proceedings of the 2nd International Conference on Big Data Economy and Digital Management, BDEDM 2023, January 6-8, 2023, Changsha, China*, 2023.

[10] Carson Eisenach, Yagna Patel, and Dhruv Madeka. Mqtransformer: Multi-horizon forecasts with context dependent and feedback-aware attention. *arXiv preprint arXiv:2009.14799*, 2020.

[11] David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. Deepar: Probabilistic forecasting with autoregressive recurrent networks. *International journal of forecasting*, 36(3):1181–1191, 2020.

[12] Hanyu Zhang, Mathieu Tanneau, Chaofan Huang, V Roshan Joseph, Shangkun Wang, and Pascal Van Hentenryck. Asset bundling for hierarchical forecasting of wind power generation. *Electric Power Systems Research*, 235:110771, 2024.

[13] Peijun Zheng, Heng Zhou, Jiang Liu, and Yosuke Nakanishi. Interpretable building energy consumption forecasting using spectral clustering algorithm and temporal fusion transformers architecture. *Applied Energy*, 349:121607, 2023.

[14] Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation modeling for general time series analysis, 2023.

[15] 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. Chronos: Learning the language of time series. *arXiv preprint arXiv:2403.07815*, 2024.

[16] Ruofeng Wen, Kari Torkkola, Balakrishnan Narayanaswamy, and Dhruv Madeka. A multi-horizon quantile recurrent forecaster. *arXiv preprint arXiv:1711.11053*, 2017.

[17] Nate Gruver, Marc Finzi, Shikai Qiu, and Andrew Gordon Wilson. Large language models are zero-shot time series forecasters. In *Advances in Neural Information Processing Systems*, 2023.

[18] Yushan Jiang, Zijie Pan, Xikun Zhang, Sahil Garg, Anderson Schneider, Yuriy Nevmyvaka, and Dongjin Song. Empowering time series analysis with large language models: A survey. *arXiv preprint arXiv:2402.03182*, 2024.

[19] Xinli Yu, Zheng Chen, Yuan Ling, Shujing Dong, Zongyi Liu, and Yanbin Lu. Temporal data meets llm–explainable financial time series forecasting. *arXiv preprint arXiv:2306.11025*, 2023.

[20] Xinglei Wang, Meng Fang, Zichao Zeng, and Tao Cheng. Where would i go next? large language models as human mobility predictors. *arXiv preprint arXiv:2308.15197*, 2023.

[21] Xin Liu, Daniel McDuff, Geza Kovacs, Isaac Galatzer-Levy, Jacob Sunshine, Jiening Zhan, Ming-Zher Poh, Shun Liao, Paolo Di Achille, and Shwetak Patel. Large language models are few-shot health learners. *arXiv preprint arXiv:2305.15525*, 2023.

[22] Hao Xue and Flora D Salim. Promptcast: A new prompt-based learning paradigm for time series forecasting. *IEEE Transactions on Knowledge and Data Engineering*, 2023.

[23] Ching Chang, Wen-Chih Peng, and Tien-Fu Chen. Llm4ts: Two-stage fine-tuning for time-series forecasting with pre-trained llms. *arXiv preprint arXiv:2308.08469*, 2023.

[24] Defu Cao, Furong Jia, Sercan O Arik, Tomas Pfister, Yixiang Zheng, Wen Ye, and Yan Liu. Tempo: Prompt-based generative pre-trained transformer for time series forecasting. *arXiv preprint arXiv:2310.04948*, 2023.

[25] Nilesh Tripuraneni, Dhruv Madeka, Dean Foster, Dominique Perrault-Joncas, and Michael I. Jordan. Meta-analysis of randomized experiments with applications to heavy-tailed response data, 2023.

[26] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[27] MosaicML NLP Team. Introducing mpt-7b: A new standard for open-source, commercially usable llms, 2023. Accessed: 2023-05-05.

A Appendix

A.1 Prompting details

The following template is used to apply on the provided text features and numerical features. The part that is being inserted based upon data is indicated with blue square brackets, []. As discussed in A.2, we also include information if the target date is in the near vicinity of a particular holiday.

> Pretend you are a sales analyst preparing for the Halloween season. You need to adjust the current model's prediction for a specific product's sales week of [forecast target date].
>
> ### Instruction:
>
> Today's date: [forecast creation date].
>
> Product Title: [product title]
>
> List Price: $[list price].
>
> Item Created At: [item creation date].
>
> Product Group Type: [product group]
>
> Description: [product description]
>
> Bullet Points: [bullet points]
>
> The current prediction for week [forecast target date]'s sales is [p50] units with a 90th percentile [p90] units. Please provide your adjusted prediction for next week's sales volume considering today's date, the season, the holiday, and the product. Explain your reasoning.

> ### Response Format: - Prediction: [Your Adjusted Prediction] units - Reasoning: [Explanation]
>
> ### Response:

A.2 Holiday encoding prompt

To improve the LLMForecaster's ability to capture holiday-driven demand patterns, we have implemented a "Holiday-Encoding Prompt". This prompt provides the model with contextual information about the temporal relationship between the target forecast date and surrounding holidays. For example, for a forecast targeting the week of June 1, 2024, the prompt would include:

> The mother's day at 2024-05-12 is 3 weeks before 2024-06-01.
>
> The father's day at 2024-06-16 is 2 weeks after 2024-06-01.

By including these details about the proximity of the target date to relevant holidays, we aim to help the model better identify the appropriate demand patterns. This helps by ensuring the LLM knows precisely when a given event will take place. This is especially important for "moving holidays" like Easter, where the exact date can vary by as much as 35 days from year to year. This Holiday-Encoding Prompt is expected to significantly improve the LLMForecaster's performance in accurately predicting sales for various holiday periods. As shown in Figure 4, the proposed LLMForecaster with the Holiday-Encoding Prompt is able to identify the Easter spike at the end of March 2024, while removing the prompt fails to capture the Easter-related demand surge.

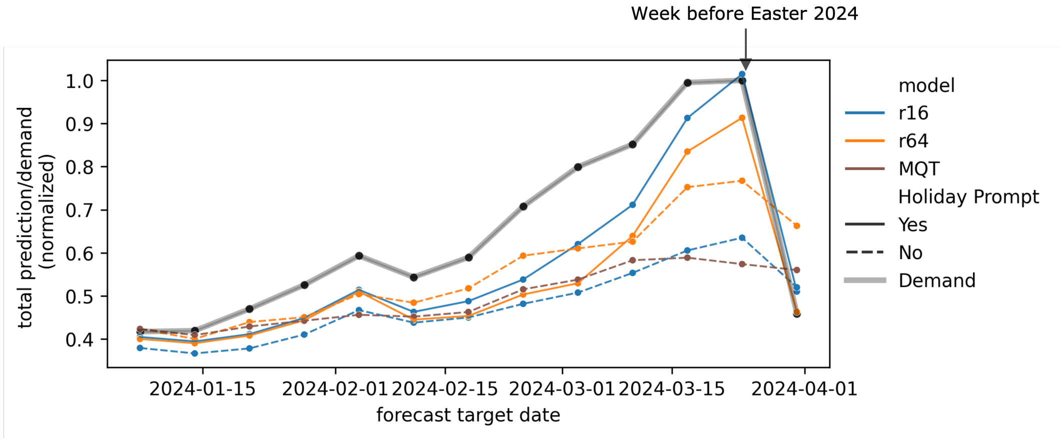

Figure 4: Total demand and prediction for Easter products with and without *Holiday-Encoding Prompt*

A.3 Accuracy results throughout the year

Here we show how the LLMForecaster changes forecast accuracy for different groups of products throughout the year. First, we show the results from a t-test measuring the weekly change in quantile loss, over the 48 weeks in the test period. Across the various product sets, we generally see statistically significant improvements over the MQT baseline. The only exception is for Valentine's Day products, which show improvements which are not statistically significant.

Next, we show the change in forecast accuracy from our LLMForecaster model versus the MQT baseline. Positive numbers indicate weeks in which the LLMForecaster model improved accuracy over the existing baseline. For each group examined, the LLMForecaster generally improves accuracy throughout the year.

Table 2: pairwise t-tests of wQL for different testsets and models.
Significance Levels: *** $p < 0.001$, ** $p < 0.01$, * $p < 0.05$

	Halloween		Easter		Father's day		Mother's day		Valentine's day	
	`MQT`	`emb`	`MQT`	`emb`	`MQT`	`emb`	`MQT`	`emb`	`MQT`	`emb`
`r16`	-5.4^{***}	-5.5^{***}	-3.6^{***}	-3.7^{***}	-5.0^{***}	-3.6^{***}	-3.8^{***}	-4.2^{***}	-3.6^{***}	-4.4^{***}
`r64`	-5.3^{***}	-5.7^{***}	-3.8^{***}	-4.1^{***}	-4.7^{***}	-3.6^{***}	-3.7^{***}	-4.0^{***}	-1.7	-2.2^{*}
`r128`	-4.2^{***}	-4.4^{***}	-3.4^{**}	-3.6^{***}	-3.9^{***}	-2.3^{*}	-2.4^{**}	-2.7^{**}	-1.9	-2.7^{**}
`r256`	-4.2^{***}	-4.3^{***}	-2.3^{*}	-2.5^{*}	-3.5^{***}	-2.2^{*}	-3.5^{**}	-3.9^{***}	-2.4^{*}	-3.1^{**}
`emb`	-1.2	-	0.6	-	-3.1^{**}	-	0.2	-	2.0	-

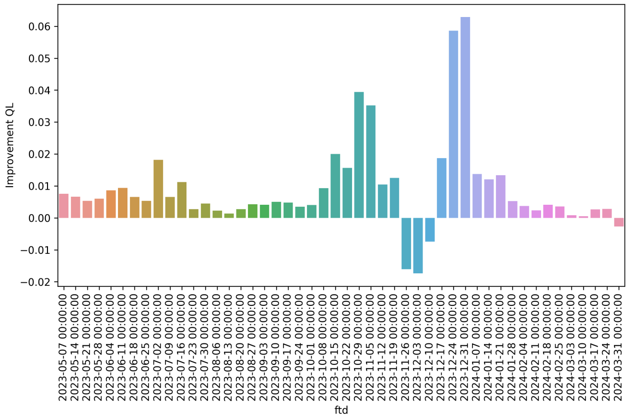

Figure 5: Forecast accuracy change for Halloween products

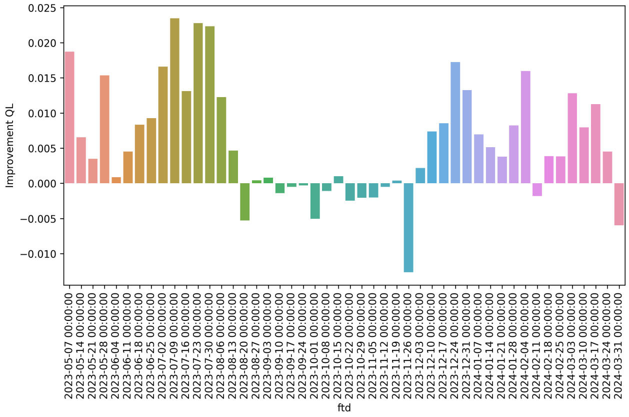

Figure 6: Forecast accuracy change for Mother's Day products

A.4 Valentine's Day

The results shown in Tables 1 and 2 indicate a relatively small and sometimes not statistically significant improvement for the Valentine's Day products compared to the other holiday categories. This can be attributed to the unique nature of the Valentine's Day holiday and its shifting position within the calendar week. For other holidays like Halloween, Easter, Mother's Day, and Father's Day, the dates are fixed on either Saturdays or Sundays, so the overall demand distribution during the holiday period remains relatively consistent. By contrast, Valentine's Day is fixed on February 14th, which can fall on different days of the week. That means that last-minute shopping, for example, may take place in the week of Valentine's Day or the prior week. In this experiment, the training dataset contained Valentine's Day falling on Monday or Tuesday - with minimal time to shop during the week of the holiday, last-minute shopping took place primarily in the prior week. In the test set, Valentine's Day occurring on a Wednesday, so consumers had more time to shop for the holiday during the week itself. When plotting the total prediction and demand on Figure 10, this effect is clearly observed. While the LLMForecaster models are able to capture the Valentine's Day demand spike compared to the baseline models, they tend to over-predict the weeks before Valentine's Day

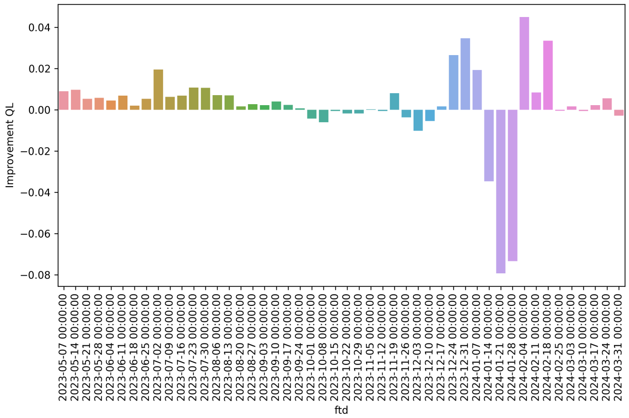

Figure 7: Forecast accuracy change for Valentine’s Day products

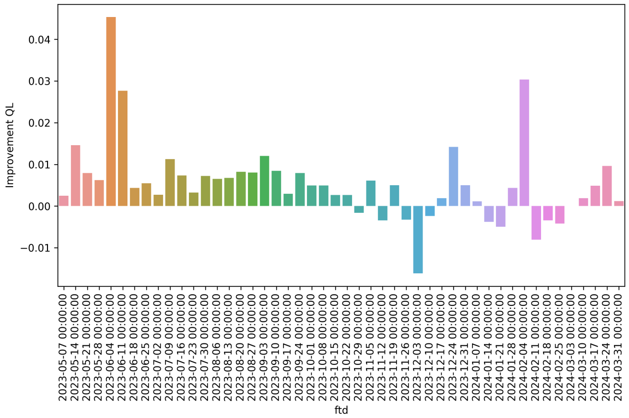

Figure 8: Forecast accuracy change for Father’s Day products

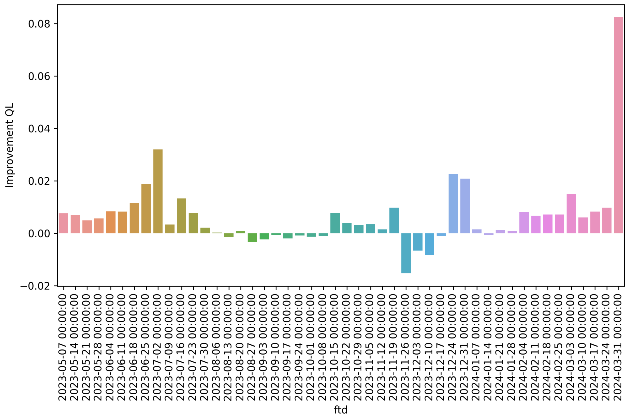

Figure 9: Forecast accuracy change for Easter products

and significantly under-predict the demand during the actual Valentine’s Day week in 2024. This issue could potentially be addressed by training the model with data spanning multiple years, or by incorporating daily demand patterns into the training process. This would help the LLMForecaster better account for the shifting position of Valentine’s Day within the calendar week and improve its ability to accurately predict the associated demand patterns.

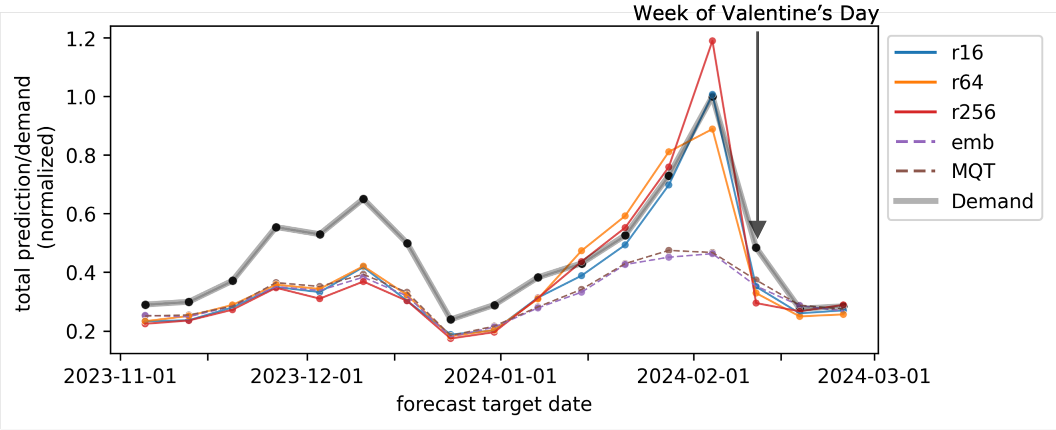

Figure 10: Total demand prediction for Valentine's Day ASINs