SPADE-S: A Sparsity-Robust Foundational Forecaster

Malcolm Wolff [*†]
wolfmalc@amazon.com

Matthew Li[*†]
yumattli@amazon.com

Ravi Kiran Selvam[*†]
ravisel@amazon.com

Hanjing Zhu[*†]
hzhuad@amazon.com

Kin G. Olivares[†]
kigutie@amazon.com

Ruijun Ma[†]
ruijunma@amazon.com

Abhinav Katoch[†]
abkatoch@amazon.com

Shankar Ramasubramanian[†]
sramasub@amazon.com

Mengfei Cao[†]
mfcao@amazon.com

Roberto Bandarra[†]
rbmp@amazon.com

Rahul Gopalsamy[†]
rahulgo@amazon.com

Stefania La Vattiata
steflvs@gmail.com

Sitan Yang[†]
sitanyan@amazon.com

Michael W. Mahoney[†]
zmahmich@amazon.com

ABSTRACT

Despite significant advancements in time series forecasting, accurate modeling of time series with strong heterogeneity in magnitude and/or sparsity patterns remains challenging for state of the art deep learning architectures. We identify several factors that lead existing models to systematically under-perform on low magnitude and sparse time series, including loss functions with implicit biases toward high-magnitude series, training-time sampling methods, and limitations of time series encoding methods. To address these limitations, we introduce `SPADE-S`, a robust forecasting architecture with a novel multi-head convolutional encoder and a model arm specifically designed to handle sparse multi-variate time series. `SPADE-S` significantly reduces magnitude and sparsity-based systematic biases and improves overall prediction accuracy; empirical results demonstrate that `SPADE-S` outperforms existing state-of-the-art approaches across a diverse set of use-cases in demand forecasting. In particular, we show that, depending on the quantile forecast and magnitude of the series, `SPADE-S` can improve forecast accuracy by up to 15%. This results in P90 overall forecast accuracy gains of 2.21%, 6.58%, and 4.28%, and P50 forecast accuracy gains of 0.92%, 0.77%, and 1.95% respectfully, for each of three distinct datasets, ranging from 3 million to 700 million series, from a large online retailer.

1 INTRODUCTION

State-of-the-art (SOTA) time series forecasting architectures have become capable of accurately modeling multivariate time series trajectories at scale in modern supply chain optimization applications. Most notably: Wen et al. [20] introduced `MQCNN`, a convolutional neural network-based architecture designed for probabilistic forecasting of multi-variate time series, demonstrating effectiveness over prior methods in capturing complex inter-dependencies among input co-variates and the target variable; Eisenach et al. [6] proposed `MQTransformer`, which incorporates attention mechanisms to reduce forecast volatility and improve accuracy; and Wolff et al. [21] introduced `SPADE` (Split Peak Attention DEcomposition), an architecture which leverages exogenous future information and self-attention to separately address peak and off-peak demand dynamics. `SPADE`, in particular, has shown ability to product accurate worldwide time series demand forecasts with a single model, which has enabled substantial model consolidation in global retail operations, improving forecast accuracy, while significantly reducing operational costs and associated technical debt.

In spite of these successes, these and other neural network architectures continue to exhibit sub-optimal performance when training and evaluating on series with highly heterogeneous magnitudes [2, 6, 13, 14, 18, 20, 21]. In demand forecasting for supply chain optimization, where average series magnitudes are often referred to as "velocity", this performance issue manifests as systematic forecasting bias and consequent accuracy degradation based on the velocity of the product; however, it is a ubiquitous phenomenon in many time series forecasting applications due to a combination of model architecture, training objective function, and training data distribution. Since common objective functions and performance metrics implicitly favor high-magnitude series, biases against low-magnitude and sparse series not only exist, but can also be difficult to detect, e.g., as they are non-obvious through aggregate evaluation metrics. As inventory management transitions to more granular level forecasting, the corresponding demand forecasting dataset becomes sparser. As the proportion of low-magnitude and sparse data increases, the cumulative bias can significantly impact overall forecasting accuracy. See, e.g., Table 1: our "moderate-velocity" dataset (`DS3`, ≈3MM series) has as little as 9.62% sparse series, 63.7% of our "low-velocity" dataset (`DS1`, ≈700MM series) is sparse, and 90% of "extremely low-velocity" dataset (`DS2`, ≈100MM series) is sparse.

There are several related issues that arise with sparse and low-magnitude time series. First, convolutional encoder-based models have limitations with extremely sparse time series, as they tend to collapse predictive distributions. Figure 1 illustrates such sparsity-induced under-dispersion, depicting the result of simulating forecasts from a single-layer causal CNN over histories whose zero fraction s is varied (the full sampling procedure is detailed in Appendix A.1). As s rises from 0.0 to 0.9, the empirically estimated 80% prediction band produced by the encoder narrows and collapses toward zero, illustrating the model's shrinking uncertainty under

[*]Authors contributed equally to this research.
[†]Amazon SCOT Forecasting. New York, New York. USA.

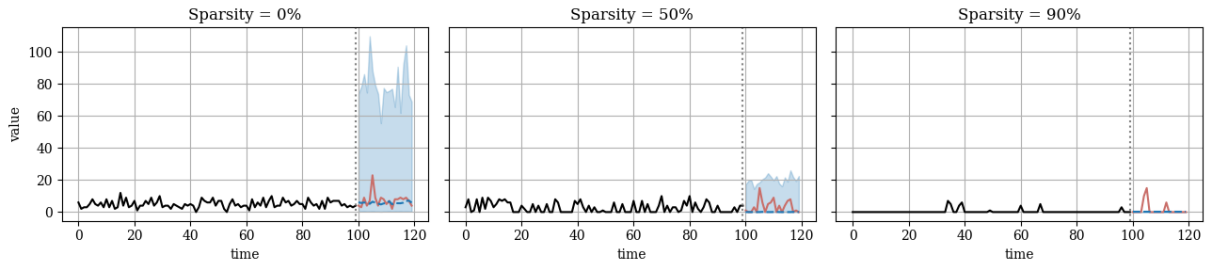


Figure 1: Convolutional forecast output by sparsity level; distributional collapse occurs at high levels of sparsity.

Category	52-week agg.	DS1	DS2	DS3
Super Fast	$(10000, \infty)$	0.05%	0.0007%	0.18%
Fast	$(365, 10000]$	1.6%	0.07%	11.77%
Medium	$(52, 365]$	4.7%	0.4%	29.32%
Slow	$(2, 52]$	17.3%	3.7%	43.40%
Super Slow	$(0, 2]$	12.7%	5.7%	5.71%
Zero	$\{0\}$	63.7%	90%	9.62%
Approx. N	(millions)	700	100	3

Table 1: Magnitude categorization of target time series calculated by 52 trailing weeks/365 trailing days, and distribution across datasets. We refer to zero demand in the prior 52 weeks/365 days as "extremely sparse series."

extremely sparse inputs. Second, convolutional encoders tend to integrate time series magnitudes inflexibly, resulting in biases dependent on the training data distribution; for example, in DS1, high magnitude series exhibit significant under-bias when sampling randomly, and significant over-bias of the lowest magnitude series when sampling proportional to magnitude (see Appendix A.2). Third, popular time series normalization techniques such as RevIN [8] are not well-compatible with forecasting tasks that requires a predictive distribution rather than a point forecast.

There is a large and continuing body of work on forecasting sparse series [e.g., 5, 7, 10, 11, 15–17, 22]. However, all of these methods share the core weakness that their objective function and internal scales operate in raw units, and do not adjust their training methodology nor their architecture to ensure accurate forecasts across all series magnitudes and sparsity levels. These concerns are becoming particularly acute for development of a foundational model for arbitrary time series forecasting problems; while the SPADE model has proven effective as a unified model for forecasting demand worldwide [21], expansion of the model to these additional use cases has exposed these emergent failure modes, and it is becoming critical that forecasting accuracy is independent of time series magnitudes. These challenges highlight a need for new methods that effectively address complexities inherent in forecasting diverse and sparse time series data.

Main contributions. In this paper, we propose SPADE-S, a sparsity-robust model architecture that provides an effective architectural solution to forecasting heterogeneous series-magnitudes and sparsity levels. Building upon the previously-developed SPADE forecasting model [21], we robustify the model to a wide range of series magnitudes and explicitly account for sparse time series, resulting in improved performance on these subsets without sacrificing overall accuracy, setting a new standard for reliable forecasting in retail and similar domains. Our main contributions are the following.

(i) **Problem Characterization.** We identify several factors that lead existing models to systematically under-perform on low-magnitude and sparse time series, including loss functions with implicit biases against higher-magnitude series, sampling methods in the training, and normalization limitations of time series encoding methods.
(ii) **A Novel Time Series Encoder.** We develop a novel multi-head dilated causal convolutional encoder module that provides critical flexibility to scale the architecture across highly diverse set of magnitudes in the time series data.
(iii) **Sparse Quantile Network.** We develop a novel sparse model arm which uses a parametric distribution to more accurately represent the behavior of sparse series without distributional collapse.
(iv) **Generalized Accuracy Improvements at Scale.** We demonstrate that our SPADE-S architecture is effective not only at scale, but robust to diverse use-cases—showing forecast improvements of up to 10.05%, 14.80%, and 6.10% depending on forecasted quantile and time series magnitude, for DS1, DS2, and DS3, respectively; this results in respective overall P50 accuracy improvements of 0.92%, 0.77%, and 1.95%, and overall P90 accuracy improvements of 2.21%, 6.58%, and 4.28%, for these three use cases.

2 METHODS

In this section, we first describe the general forecasting task and it's consequences for forecasting across diverse time series; and we then describe our architecture and it's novel contributions.

2.1 Forecasting Task

We consider a general product demand forecasting task [6, 20, 21], with forecast creation dates $t \in [T] \equiv \{1, \ldots, T\}$, products of interest $i \in \mathcal{I}$, and forecast horizons $h = (\text{lead-time}, \text{span}) \in \mathcal{H}$ (a combination of approximately 240 valid lead-time/span pairs over the next 52 weeks). We'll denote the size of $\mathcal{H}$ as $|\mathcal{H}|$, and

we'll denote all span-1 horizons as $\mathcal{H}_1$. Our input covariates consist of past information $\mathbf{x}^{(p)}_{[t]} \in \mathbb{R}^{T \times d_p}$, known future information $\mathbf{x}^{(f)}_{[t],\mathcal{H}_1} \in \mathbb{R}^{T \times d_f \times |\mathcal{H}_1|}$, and static information $\mathbf{x}^{(s)} \in \mathbb{R}^{d_s}$, and we represent the target variable at time t as $\mathbf{y}_{t,\mathcal{H}} \in \mathbb{R}^{T \times |\mathcal{H}|}$. The forecasting task estimates the following conditional distribution:

$$\mathbb{P}\left(\mathbf{y}_{t,\mathcal{H}} \mid \boldsymbol{\theta},\ \mathbf{x}^{(p)}_{[t]},\ \mathbf{x}^{(f)}_{[t],\mathcal{H}_1},\ \mathbf{x}^{(s)}\right). \tag{1}$$

We use weighted quantile loss (WQL) as our evaluation objective, where the WQL is defined as

$$\mathrm{WQL}(\mathbf{y}, \hat{\mathbf{y}}^{(q)}; q, \mathcal{I}, \mathcal{H}) = \frac{\sum_{i,t,h} \mathrm{QL}\left(y_{i,t,h},\ \hat{y}^{(q)}_{i,t,h}(\boldsymbol{\theta}); q\right)}{\sum_{i,t,h} y_{i,t,h}}, \tag{2}$$

and $\mathrm{QL}(y,\ \hat{y};\ q) = q(y - \hat{y})_+ + (1 - q)(\hat{y} - y)_+$ is the quantile loss function, $\hat{y}^{(q)}$ denotes the estimated quantile, and $\boldsymbol{\theta}$ denotes a model in the class of models Θ defined by the model architecture. We optimize $\boldsymbol{\theta}$ by minimizing the numerator of equation (2) summed across the quantiles of interest. See Appendix A.3 for details.

Magnitude-Bias of Common Loss Functions. Like most objectives, the Quantile Loss function introduces implicit inherent biases that warrant careful consideration. To illustrate this, let $(y_{i,t,h}, \hat{y}_{i,t,h})$ be the target and forecast for series i, time t, horizon h. Assume every series is forecast with the same *relative error* $r_{i,t,h} = r$ (e.g. a 10% miss everywhere). Then the absolute error equals $e_{i,t,h} = r\, y_{i,t,h}$. Let the pointwise loss satisfy

$$\ell(y, \hat{y}) \ = \ g(y)\, f(r), \qquad r \equiv \frac{\hat{y} - y}{y},$$

i.e. it factors into a scale term $g(y) > 0$ and a function $f(r) > 0$ of the relative error r. If g is strictly increasing, then the global sample loss

$$\mathcal{L} \ = \ \frac{1}{N} \sum_{i,t,h} \ell(y_{i,t,h}, \hat{y}_{i,t,h})$$

weights each series i in proportion to its total magnitude $w_i \propto \sum_{t,h} g(y_{i,t,h})$, so higher-magnitude series contribute disproportionately (see Appendix A.4 for the proof). In particular, the quantile loss can be written as $|q - \mathbf{1}_{y<\hat{y}}||e| = |q - \mathbf{1}_{y<\hat{y}}||r|y = g(y)f(r)$ for $g(y) = y$. Consequently, under equal relative error, time series with larger magnitudes exert strictly larger influence on the aggregate quantile loss.

Although prioritizing high-volume items can be reasonable in, e.g. supply-chain optimization, the implicit magnitude-based weighting becomes problematic when training unified, multi-purpose forecasters, if those forecasters do not have sufficient capacity to model heterogeneity in series-level magnitudes. In product demand forecasting, for example, since the loss scales with absolute demand, (i) larger marketplaces, (ii) already successful products, and (iii) use-cases with inherently bigger signals all exert disproportionate influence, creating a self-reinforcing bias that can erode accuracy for emerging markets, new items, and low-scale applications—precisely the scenarios a truly universal model must handle equitably. ´

2.2 SPADE-S Architecture

Our model architecture, depicted in Figure 2, introduces novel contributions to address diverse time series magnitudes and sparsity levels, including a masked multi-head dilated convolutional encoder, sparse series routing and a sparse quantile network, which we describe below.

Multi-Head Convolutional Encoder. Drawing inspiration from multi-head self-attention mechanisms [19] where different heads learn complementary input representations, we propose multi-head convolutional encoder that achieves similar representational diversity but with significantly lower computational overhead.

Parallel convolutional encoding has also been leveraged in other contexts: speeding up spectrogram inversion with parallel convolution, where each convolution learns a different interpolation pattern [1]; applying convolutional filters in parallel over each input series to condition on several covariates before combining them into a residual stack [3]; and leveraging several parallel dilated convolutional heads with different rates in a multi-scale graph wavenet architecture for wind speed forecasting [12].

Unlike prior parallel convolution variants that handcraft each head's receptive field or filter type, we show that by simply instantiating multiple identical dilated convolution stacks in parallel, we achieve the robustness and uncertainty benefits of an ensemble without manual head engineering or expensive training of separate models.

Specifically, we calculate

$$\begin{aligned} \mathbf{e}^{(p)}_{[t],g} &= \mathrm{Convolution}_g\left(\widetilde{\mathbf{x}}^{(\mathbf{p})}_{[\mathbf{t}]}\right) \\ \mathbf{e}^{(p)}_{[t]} &= \mathrm{Linear}\left(\mathbf{e}^{(p)}_{[t],1}, \ldots, \mathbf{e}^{(p)}_{[t],G}\right), \end{aligned} \tag{3}$$

where $\widetilde{\mathbf{x}}^{(p)}_{[t]}$ are historical series inputs that have been peak-filtered (as in `SPADE` [21]), and each individual head is a dilated causal convolution encoder seen in prior work [6, 20, 21]. Our method combines the variance-reduction power of ensembles [4] with the efficiency of a single, shared-structure encoder.

Sparse Series Routing. To effectively route extremely sparse time series to a separate model arm, we leverage known information about the time series in a manner similar to `SPADE` [21]. We first separate the input batch into "sparse" and "non-sparse" series with the `SparsityMask` module. We define "sparse" as having zero aggregate demand in the trailing 52 weeks, and not being classified as a new product, which we identify by date of first recorded product listing. The "non-sparse" series are routed to the main encoder, which contains our masked multi-head convolutional encoder; and the "sparse" series are routed to our `SparseQuantileNetwork`.

Sparse Quantile Network. The `SparseQuantileNetwork` first uses a patched MLP to estimate the parameters of a simple parametric distribution, dis-aggregates those parameters across the horizons in $\mathcal{H}$, and uses the horizon-specific parameters to produce a quantile forecast via an inverse cumulative distribution function (ICDF). Note that this general frame works for any simple parameteric distribution; one could estimate parameters from an, e.g., Truncated Shifted Gamma (TSG) distribution and dis-aggregate using it's additive properties (see Appendix A.5). However, the ICDF of a Gamma has no closed form solution, requiring sample path generation which significantly increases training and inference costs for tail quantiles. One could also use the additivity of i.i.d. gaussian

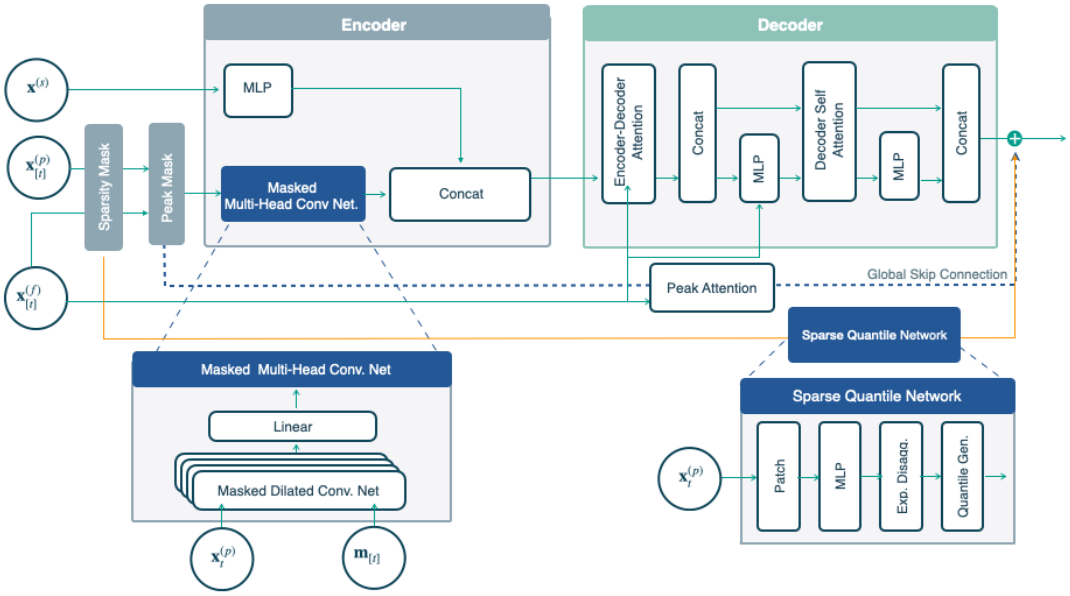


Figure 2: SPADE-S architecture, including modules to address diverse time series magnitudes, including masked multi-head dilated convolutional encoder, sparse series routing and a sparse quantile network.

parameters to generate quantiles for a truncated normal (see Appendix A.5)—results we illustrate in our ablation study (Appendix B).

However, to retain both computational simplicity as well as to capture the heavy-tail characteristics of empirical product demand distributions, we assume the quantiles across horizons h to come from a simpler exponential distribution, whose simple ICDF is $\hat{y}^{(q)}_{[t],\mathcal{H}} = -h(\text{span}, \vartheta_{[t]}) \ln(1-q)$, where $h(\text{span}, \vartheta_{[t]}) = \frac{\text{span}}{\varsigma} \vartheta_{[t]}$ represents the exponential scale parameter, assuming that span-1 demand variables are perfectly correlated. To summarize, the sparse model arm does the following:

$$\begin{aligned} \mathbf{s}^{(p)}_{[t]} &= \text{Patch}\left(\mathbf{x}^{(p)}_{[t]}\right) \\ \vartheta_{[t]} &= \text{MLP}\left(\mathbf{s}^{(p)}_{[t]}\right) \\ \hat{y}^{(q)}_{[t],\mathcal{H}} &= F^{-1}\left(q;\ h(\text{span},\ \vartheta_{[t]})\right), \end{aligned} \tag{4}$$

where $h(\text{span}, \vartheta_{[t]})$ is a function of span, and $F^{-1}(q)$ denotes the quantile function for the particular parametric distribution used. Restriction of the forecast to a simple parametric distribution is critical in robust estimation of sparse quantile forecasts. In this case, we always forecast quantiles no more than P50 to be zero, as any continuous distribution would result in P50 over-bias, and learned probability masking results in difficult-to-learn architectures using discrete optimization methods.

3 RESULTS

We evaluate SPADE-S on three diverse large-scale product forecasting applications. See Table 1 for a summary. Each of these datasets not only have different distribution of series magnitudes (see, e.g., Table 1), but also variable input and product demand characteristics, which we describe below. In what follows, we'll review the experimental setup of each of these applications, including their particular data and forecasting task, and then we will present our main results.

3.1 Setup of Our Empirical Evaluation

Low velocity series forecasts (D1). Our first use case is world-wide listing level forecasts for online retail products. In this use case, we aim to predict listing demand across the entire forecast horizon $\mathcal{H}$. The training data is weekly grain information spanning 260 weeks from 2017 to 2022 and consisting of hundreds of millions of unique listings. The backtest data consists of the subsequent 52 weeks in 2023 after the training period. The time series in the backtest are a total population of nearly one billion unique listing series.

Extremely low velocity forecasts (D2). Our second use case is forecasting weekly product demand for an online retailer per geographic area. Unlike the prior use case, the relevant forecast is up to a 10 week horizon. The training data is weekly grain information spanning 260 weeks from 2017 to 2022 of over 100 million series. Similar to D1, we include features to capture exogenous information such as holidays and promotions. For the backtest data, we use a uniform random sample of over 100 million series for the subsequent 52 weeks in 2023 after the training period. As is shown in Table 1, more than 90% of the series are sparse (categorized as "Zero"), while fewer than 1% of the series are considered "Medium" or faster.

Moderate velocity forecasts (D3). Our third use case is forecasting daily demand for products at the store level. In this use-case, we aim to forecast across an entire forecasting horizon $\mathcal{H}$ = 91 days and various spans ranging consisting of a total of 285 lead-time/span

combinations. Since the product selection for stores is limited compared to online marketplaces, it is very unlikely to retain offers for the slowest products, and hence the % of super-fast and fast categories are much higher up to 12% vs <2% for other use-cases. The training data consist of daily grain information spanning 730 days, starting from 2022 to 2023 and totaling over 3 million unique series.

The dataset also include features to capture exogenous information such as holidays and promotions along with static information about the product like brand, category etc. The backtest data consist of subsequent 365 days after the end of training period for each product-store time series starting from 2024.

3.2 Main Empirical Results

Our main empirical results are presented in Table 2. Results are displayed as the percent difference in P50 and P90 quantile loss relative to the baseline model. The baseline models are `SPADE` [21] for `D1` and `D2`, and `MQTransformer` [6] for `D3`[1]. Results are decomposed by the time series magnitude categorizations shown in Table 1, along with the proportion of evaluation data falling into each of these categorizations.

Category	Metric	D1	D2	D3
All	P50	-0.92%	-0.77%	-1.95%
	P90	-2.21%	-6.58%	-4.28%
Super Fast	P50	-0.77%	-2.00%	-1.94%
	P90	-1.20%	-10.30%	-3.28%
Fast	P50	-0.72%	-3.90%	-2.99%
	P90	-1.02%	-14.80%	-6.10%
Medium	P50	-1.01%	-2.50%	-1.03%
	P90	-1.38%	-12.20%	-2.52%
Slow	P50	-0.95%	1.60%	-0.01%
	P90	-2.50%	-3.60%	-0.51%
Super Slow	P50	-0.94%	0.80%	-0.01%
	P90	-5.54%	-0.30%	-1.03%
Zero	P50	-4.37%	0.40%	-0.45%
	P90	-10.05%	0.20%	0.27%

Table 2: Main results of `SPADE-S` by task and series magnitudes defined in Table 1, compared to benchmark models.

As shown in Table 2, we find general improvement across all magnitude categories and use-cases. Moreover, our magnitude level results suggest that baseline models tend to favor the construction of accurate forecasts for the highest magnitude targets to the detriment of lower magnitude targets. `SPADE-S` alleviates this issue. For `D1`, `SPADE-S` notably shows P90 forecast improvements on "slow", "super slow", and "zero" products of 2.50%, 5.54%, and 10.05%, respectively, and a P50 forecast improvement of 4.37% for "zero" products—improvements primarily driven by significant reductions in over-bias of the forecast, as seen in Appendix Table 3.

For `D2`, we also observe significant improvement on P90 forecast—10.30% on "Super Fast", 14.80% on "Fast", and 12.20% on "Medium" series. Sparse series routing prevented zero-value products from biasing faster-moving products, reducing under-prediction and improving overall accuracy. Detailed results are in Appendix B.2.

For `D3`, we also observe large improvements in high magnitude time series categories—with P90 improvements of 3.28%, 6.10% and 2.52% in "super fast", "fast", and "medium" categories, primarily driven by reduction in the under-bias of the forecast. Since the number of zero magnitude time series is relatively less (i.e, <10%) compared to national/regional use-cases (i.e., >63% to 90%), routing them to sparse ARM is not necessary to achieve overall improvements in both high and low magnitude time series categories.

4 DISCUSSION

`SPADE-S` is a significant advancement in addressing heterogeneous time series data characterized by varying magnitudes and sparsity patterns. Empirical results on three separate massive internal datasets reveal several key insights worth examining. Most notably, the substantial improvements in forecast accuracy across different use cases validate our initial hypothesis that existing models systematically under-perform on low-magnitude and sparse time series. In forecasting across nearly 1 billion series in `D1`, `SPADE-S` achieved up to 10% improvement in accuracy, depending on the quantile forecast and magnitude of the series. Similarly, forecasting across over 100 million series and over 3 million series in `D2` and `D3` showed improvements of up to 15% and 6% respectively. These results suggest that the model's benefits are transferable across varying dataset size and complexity. Moreover, P90 accuracy gains of 2.21% for `D1`, 6.58% for `D2`, and 4.28% for `D3` indicate that `SPADE-S` handles tail estimation more effectively than existing approaches, while the P50 gains of 0.92% and 0.77% and 1.95% respectively suggest that the model maintains strong performance across median cases.

These findings have significant practical implications for industries relying on large-scale time series forecasting, particularly in retail and supply chain management. The ability to forecast more accurately across varying magnitudes and sparsity patterns could lead to improved inventory management, reduced waste, and more efficient resource allocation. Moreover, the success of our multi-head convolutional encoder opens new avenues for research in handling heterogeneous multivariate time series, and this architectural innovation could potentially be adapted for other applications beyond demand forecasting.

5 CITATIONS

REFERENCES

[1] Sercan Ö. Arık, Heewoo Jun, and Gregory Diamos. Fast spectrogram inversion using multi-head convolutional neural networks. *arXiv preprint arXiv:1808.06719*, 2018. Multi-head CNN for spectrogram inversion (MCNN).

[2] Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. In *International Conference on Learning Representations*, 2018.

[3] Anastasia Borovykh, Sander M. Bohte, and Cornelis W. Oosterlee. Conditional time series forecasting with convolutional neural networks. *arXiv preprint arXiv:1703.04691*, 2017. WaveNet-inspired CNN with parallel conv heads for

[1] `D3` does not observe the same extreme holiday- and promotion-related spikes as other demand forecasting problems, so `PeakAttention` is unnecessary.

multivariate conditioning.
[4] Leo Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.
[5] J. D. Croston. Forecasting and stock control for intermittent demands. *Operational Research Quarterly*, 23(3):289–303, 1972.
[6] Carson Eisenach, Yagna Patel, and Dhruv Madeka. MQTransformer: Multi-Horizon Forecasts with Context Dependent and Feedback-Aware Attention. *Computing Research Repository.*, 8 2020.
[7] Rafael S. Gutierrez, Adriano O. Solis, and Somnath Mukhopadhyay. Lumpy demand forecasting using neural networks. *International Journal of Production Economics*, 111(2):409–420, 2008.
[8] Donghwa Kim, Dongjin Seo, Jaehyeon Lee, and Jaewoo Kang. Reversible instance normalization for accurate time-series forecasting. In *International Conference on Learning Representations (ICLR)*, 2022. OpenReview cGDAkQo1C0p.
[9] Diederik P. Kingma and Jimmy Ba. ADAM: A method for stochastic optimization, 2014. cite arxiv:1412.6980Comment: Published as a conference paper at the 3rd International Conference for Learning Representations (ICLR), San Diego, 2015.
[10] Nikolaos Kourentzes. Intermittent demand forecasts with neural networks. *International Journal of Production Economics*, 143(1):198–206, 2013.
[11] Konstantinos Nikolopoulos, Aris A. Syntetos, John E. Boylan, Fotios Petropoulos, and Vassilios Assimakopoulos. An aggregate–disaggregate intermittent demand approach (adida) to forecasting: an empirical proposition and analysis. *Journal of the Operational Research Society*, 62(3):544–554, 2011.
[12] Neetesh Rathore, Pradeep Rathore, Arghya Basak, Sri Harsha Nistala, and Venkataramana Runkana. Multi scale graph wavenet for wind speed forecasting. *arXiv preprint arXiv:2109.15239*, 2021. Graph-WaveNet with inception-style multi-dilation heads.
[13] David Salinas, Valentin Flunkert, and Jan Gasthaus. Deepar: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting*, 36(3):1181–1191, 2020.
[14] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*, volume 27, pages 3104–3112, 2014.
[15] Aris A. Syntetos and John E. Boylan. The accuracy of intermittent demand estimates. *International Journal of Forecasting*, 21(2):303–314, 2005.
[16] Ruud H. Teunter, Aris A. Syntetos, and M. Zied Babai. Intermittent demand: Linking forecasting to inventory obsolescence. *European Journal of Operational Research*, 214(3):606–615, 2011.
[17] Ali Caner Turkmen, Yuyang Wang, and Tim Januschowski. Intermittent demand forecasting with deep renewal processes. *CoRR*, abs/1911.10416, 2019.
[18] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. In *Proceedings of the 9th ISCA Speech Synthesis Workshop*, 2016.
[19] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.
[20] Ruofeng Wen, Kari Torkkola, Balakrishnan Narayanaswamy, and Dhruv Madeka. A Multi-horizon Quantile Recurrent Forecaster. In *31st Conference on Neural Information Processing Systems NIPS 2017, Time Series Workshop*, 2017.
[21] Malcolm Wolff, Kin G. Olivares, Boris Oreshkin, Sunny Ruan, Sitan Yang, Abhinav Katoch, Shankar Ramasubramanian, Youxin Zhang, Michael W. Mahoney, Dmitry Efimov, and Vincent Quenneville-Bélair. ♠ SPADE ♠ split peak attention decomposition. In *Thirty-Eighth Annual Conference on Neural Information Processing Systems NeurIPS 2024*, volume Time Series in the Age of Large Models Workshop, pages –, Vancouver, Canada, 2024. NeurIPS 2024.
[22] G. Peter Zhang, Yusen Xia, and Maohua Xie. Intermittent demand forecasting with transformer neural networks. *Annals of Operations Research*, 339(1):1051–1072, 2024.

APPENDICES

A SUPPLEMENTARY DETAILS

A.1 Convolutional collapse simulation details.

We construct a synthetic experiment to illustrate how overall sparsity degrades the predictive dispersion produced by a convolutional encoder. For each sparsity level $s \in \{0, 0.5, 0.9\}$ we generate a 100-step history $y_{1:100}$ from a Poisson(5) process and randomly set a fraction s of its entries to zero. The history is fed into a single-layer causal CNN with a 24-lag, two-channel, exponentially-weighted kernel whose coefficients are normalised to sum to one; a learned scalar bias is added to the convolutional output. Letting μ denote the last convolved value and σ the standard deviation of the most recent 30 non-zero observations, we draw 500 Monte-Carlo sample paths for the 20-step forecast horizon according to $\hat{y}^{(d)}_{t+h} = \mu + \sigma z^{(d)}_h$ with $z^{(d)}_h \sim \mathcal{N}(0,1)$. The empirical 10th, 50th and 90th percentiles across the draws yield an 80% prediction interval $[\hat{y}^{(10)}, \hat{y}^{(90)}]$ whose width—and ultimately its collapse toward zero—is visualised as sparsity increases. A future trajectory with the same sparsity pattern is overlaid to highlight the encoder's growing under-dispersion and its failure to capture demand resurgence.

A.2 Bias Trade-off by Sampling Scheme

To show the under-bias over-bias trade-off by sampling scheme, we use D1 training and backtest data described in section 3. We train a baseline SPADE [21] model, which serves as the same baseline as our main results, using series magnitude-based importance sampling with a cutoff quantile of 0.8—i.e., we sample more frequently in proportion to series magnitude for products with magnitudes above the P80 quantile, and use uniform sampling below P80, where the uniform weight is equivalent to a P80 magnitude. Our experimental model trains the same SPADE architecture, but uses series magnitude based importance sampling for the entire magnitude distribution—i.e., all products are sampled according to their series magnitudes. Figure 3 shows that the experimental model, which samples according to velocity across the entire distribution, shows over-bias on P90 for "super slow" products, and extreme over-bias on at P90 for "zero" products, while improving on both under-bias and over-bias for the remaining products. However, the extreme over-bias on "zero" products, given that they make up for over 60% of the total population, results in overall over-bias of the model.

A.3 Training Methods

Let $\boldsymbol{\theta}$ be a model that resides in the class of models Θ defined by the model architecture. Let $\mathcal{A}$ the dataset's products, and $\mathcal{H}$ the horizon defined by lead times and spans. We train a quantile regression model by minimizing the following multi-quantile loss:

$$\min_{\boldsymbol{\theta}} \sum_q \sum_i \sum_t \sum_h \mathrm{QL}\left(y_{i,t,h},\ \hat{y}^{(q)}_{i,t,h}(\boldsymbol{\theta});\ q\right), \tag{5}$$

for products $i \in \mathcal{I}$, time t and horizon $h \in \mathcal{H}$, and $\hat{\underline{y}}^{(q)}$ denotes the estimated quantile[2]. We optimize SPADE-S using stochastic gradient descent with *Adaptive Moments* (ADAM; [9]).

[2]During training, demand and forecasts are normalized by the length of the horizon h.

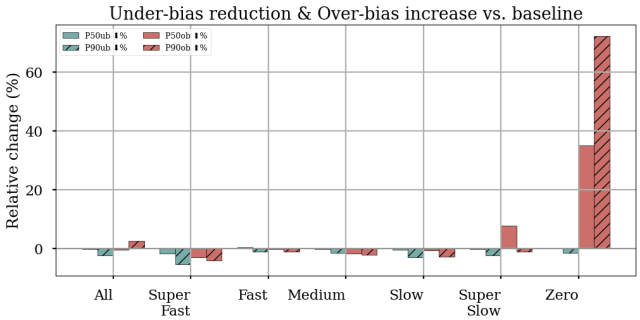


Figure 3: Relative change of under-bias and over-bias versus baseline forecast by sampling scheme for D1 dataset.

A.4 Magnitude-Bias of Common Loss Functions

Proposition. Let $(y_{i,t,h}, \hat{y}_{i,t,h})$ be the target and forecast for series i, time t, horizon h. Assume every series is forecast with the *same relative error* $r_{i,t,h} = r$ (e.g. a 10% miss everywhere). Then the absolute error equals $e_{i,t,h} = r\, y_{i,t,h}$. Let the pointwise loss satisfy

$$\ell(y, \hat{y}) \;=\; g(y)\, f(r), \qquad r := \frac{\hat{y} - y}{y},$$

i.e. it factors into a scale term $g(y) > 0$ and a function $f(r) > 0$ of the relative error r. If g is strictly increasing, then the global sample loss

$$\mathcal{L} \;=\; \frac{1}{N} \sum_{i,t,h} \ell(y_{i,t,h}, \hat{y}_{i,t,h})$$

weights each series i in proportion to its total magnitude $w_i \propto \sum_{t,h} g(y_{i,t,h})$, so higher-magnitude series contribute disproportionately.

Proof. Insert $\ell = g(y)f(r)$ into $\mathcal{L}$ and regroup by series:

$$\mathcal{L} = \frac{1}{N} \sum_i \sum_{t,h} g(y_{i,t,h}) f(r_{i,t,h}) = \sum_i \left(\frac{\sum_{t,h} g(y_{i,t,h})}{N} \right) \left(\frac{\sum_{t,h} g(y_{i,t,h}) f(r_{i,t,h})}{\sum_{t,h} g(y_{i,t,h})} \right)$$

The first factor is the series weight w_i, which grows with y because g is increasing, proving the bias. □

We examine three common losses in this regime.

1. MSE (Mean–Squared Error).

$$\ell_{\text{MSE}} \;=\; (\hat{y} - y)^2 \;=\; e^2 \;=\; r^2\, y^2 \quad \Longrightarrow \quad g(y) = y^2.$$

Thus, under equal relative error its effective weight grows quadratically with magnitude, so large-scale series dominate the summed loss.

2. CRPS (Continuous Ranked Probability Score). For many location-scale forecast families (e.g. Normal, Laplace) one can show

$$\text{CRPS}(F, y) \;=\; \sigma\, \varphi(r) \quad \text{with} \quad r = \frac{\hat{\mu} - y}{\sigma},$$

where σ is the predictive scale and φ depends only on the standardised error. If forecasts keep afixed relative spread, $\sigma = \kappa\, y$, then

$$\text{CRPS} \propto y, \quad \text{i.e.} \quad g(y) = y,$$

again privileging high-magnitude series.

3. Quantile (Pinball) Loss. For a τ-quantile forecast $\hat{q}$, the pinball loss is $\ell_\tau = (\tau - \mathbf{1}_{y<\hat{q}})(y - \hat{q}) = |\tau - 1_{y<\hat{q}}|\, |e|$. Equal relative error implies $|e| = |r|y$, giving $g(y) = y$.

A.5 Sparse Quantile Network Parametric Distributions.

Gamma. For i.i.d. Gamma variables $X_i \sim \text{Gamma}(k_i, \vartheta)$,

$$\sum_i X_i \sim \text{Gamma}\Big(\sum_i k_i,\ \vartheta\Big),$$

We can first estimate the scale ϑ for the maximal forecasted span ς in the horizon set $\mathcal{H}$, and then disaggregate the distribution under the assumption the shape parameter is proportional to span, using the associated distribution to generate a quantile forecast. However, the inverse quantile function does not have a closed form, which necessitates generation of many sample paths to produce a backpropagatable quantile estimate.

Truncated Normal. If one instead assumes $X_1, \ldots, X_n \overset{\text{iid}}{\sim} \mathcal{N}(\mu, \sigma^2)$. Then

$$S = \sum_{i=1}^{n} X_i \sim \mathcal{N}\big(n\mu,\ n\sigma^2\big), \quad \bar{X} = \frac{1}{n} S \sim \mathcal{N}\Big(\mu,\ \frac{\sigma^2}{n}\Big).$$

Thus aggregation of n iid normals gives

$$\mu_S = n\mu, \qquad \sigma_S^2 = n\,\sigma^2,$$

and dis-aggregation (recovering the original parameters) is

$$\mu = \frac{\mu_S}{n}, \qquad \sigma^2 = \frac{\sigma_S^2}{n}.$$

This can be leveraged to decompose estimated parameters by span. Then the inverse quantile function is

$$\text{For } X \sim \mathcal{N}(\mu, \sigma^2),\ F_X(x) = \Phi\big(\tfrac{x-\mu}{\sigma}\big) \quad \Longrightarrow \quad F_X^{-1}(p) = \mu + \sigma\, \Phi^{-1}(p),$$

$$\Phi^{-1}(p) = \sqrt{2}\, \text{erf}^{-1}(2p - 1),$$

and the result can be turned into a truncated normal with a ReLU on the quantile forecast produced.

B ABLATION STUDIES

We run a number of architectural ablation studies to compare different methodologies. All ablations follow the same experimental pipeline as our main results. In the ablations below, `Spade V0` refers to the original `SPADE` architecture [21]. "Adjusted cutoff quantile" is a parameter of our training sampling scheme, which uses magnitude-based importance sampling; the cutoff quantile is the quantile in which every observation below the magnitude q receives uniform weight, with weight equivalent to the quantile q. "Rule-based" override layer refers to a `Sparse Quantile Network` that forces all sparse series to forecast 0 for P50 and 0 for P90. MoE attempts to learn the encoder head mixture through the average mean and variance of the historic target values. Poseterior analysis reveals that the learned weight is approximately uniform. The learned P90 layers experiment with multiple different distributions in the `SparseQuantileNetwork`.

- **V9.** Spade V0
- **V10.** Spade V0 + Adjusted cutoff quantile (0.8 to 0.1)
- **V11.** Spade V0 + Rule-based override layer

V12. Spade V0 + Adjusted cutoff quantile (0.8 to 0.1) + Rule-based override layer
V13. Spade V0 + MoE with soft routing (6 experts)
V15. Spade V0 + learned P90 layer (truncated normal distribution on raw input)
V16. Spade V0 + MoE (6 experts) + learned P90 layer (truncated normal distribution on raw input)
V17. Spade V0 + learned P90 layer (exponential distribution on raw input)
V18. Spade V0 + MoE (6 experts) + learned P90 layer (exponential distribution on raw input)
V19. Main Model (Figure 2)

B.1 D1 Ablations

Table 3 shows detailed experimental results on the D1 dataset. We find that V19, the main model, is the consistently most successful model across all velocities and overall. It's worth noting that V13, which has an MoE encoder without a sparse arm, has nearly competitive performance outside of the "Zero" category, but the sparse arm has an important effect on this category that influences the overall quantile loss, given it makes up for over 60% of the dataset.

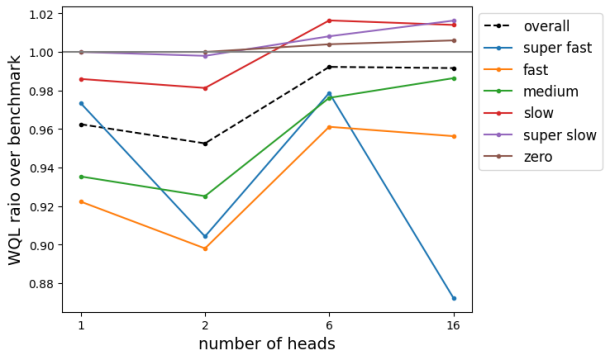
		Version									
Category	**Metric**	**V9**	**V10**	**V11**	**V12**	**V13**	**V15**	**V16**	**V17**	**V18**	**V19**
All	P50ql		0.04%	-0.42%	-0.46%	-0.58%	-0.29%	-0.50%	-0.38%	-0.79%	**-0.92%**
	P90ql		2.38%	-1.33%	-1.38%	-1.33%	-0.22%	-1.44%	-1.27%	-1.94%	**-2.21%**
	P50ob		-0.54%	-4.90%	-0.12%	-3.63%	-1.15%	0.12%	-2.24%	-6.65%	-2.18%
	P50ub		0.32%	1.88%	-0.67%	0.99%	0.13%	-0.89%	0.54%	2.29%	-0.32%
	P90ob		2.41%	-5.57%	-4.26%	-4.05%	-2.16%	-2.29%	-5.03%	-8.04%	-4.81%
	P90ub		2.41%	3.72%	2.08%	1.97%	2.19%	-0.33%	3.29%	5.37%	0.99%
Super Fast	P50ql		-0.06%	0.13%	-0.26%	-0.51%	0.06%	-0.26%	0.13%	-0.26%	**-0.77%**
	P90ql		0.22%	0.00%	-0.33%	-0.98%	0.00%	-0.44%	0.00%	-0.65%	**-1.20%**
	P50ob		-3.14%	-4.66%	0.17%	-2.46%	-1.86%	0.51%	-2.63%	-4.75%	-0.51%
	P50ub		1.79%	3.07%	-0.51%	0.67%	1.28%	-0.72%	1.79%	2.46%	-0.87%
	P90ob		-4.12%	-2.98%	-1.99%	-1.40%	-1.83%	-1.38%	-2.86%	-4.20%	-1.93%
	P90ub		5.49%	3.74%	1.76%	-0.44%	2.42%	0.66%	3.74%	3.74%	-0.22%
Fast	P50ql		-0.38%	-0.30%	-0.34%	-0.55%	-0.26%	-0.38%	-0.34%	**-0.72%**	**-0.72%**
	P90ql		-0.32%	-0.32%	-0.51%	-0.83%	-0.32%	-0.76%	-0.19%	-0.89%	**-1.02%**
	P50ob		-0.28%	-3.73%	1.07%	-3.27%	-0.56%	0.56%	-1.30%	-6.78%	-0.96%
	P50ub		-0.44%	1.74%	-1.23%	1.06%	-0.07%	-0.96%	0.24%	2.90%	-0.55%
	P90ob		-1.27%	-2.13%	-1.36%	-2.19%	-1.57%	0.16%	-1.95%	-5.74%	-0.79%
	P90ub		1.07%	2.13%	0.67%	1.07%	1.46%	-1.86%	2.26%	5.73%	-1.20%
Medium	P50ql		-0.44%	-0.69%	-0.57%	-0.66%	-0.41%	-0.69%	-0.47%	**-1.23%**	-1.01%
	P90ql		-0.49%	-0.45%	-0.89%	-1.02%	-0.89%	-1.30%	-0.45%	**-1.58%**	-1.38%
	P50ob		-1.79%	-3.86%	0.37%	-4.28%	1.38%	1.79%	-0.28%	-6.67%	-2.16%
	P50ub		0.26%	1.01%	-1.03%	1.25%	-1.30%	-1.97%	-0.55%	1.63%	-0.38%
	P90ob		-2.14%	-2.60%	-1.67%	-3.33%	-1.88%	0.65%	-0.95%	-5.48%	-1.69%
	P90ub		1.59%	2.27%	0.08%	1.93%	0.34%	-3.86%	0.17%	3.36%	-1.01%
Slow	P50ql		0.15%	-0.57%	-0.62%	-0.17%	-0.37%	-0.40%	-0.55%	**-0.62%**	**-0.95%**
	P90ql		-0.08%	-1.18%	-1.84%	-1.61%	-2.02%	**-2.94%**	-1.84%	-1.97%	-2.50%
	P50ob		-0.67%	-6.97%	-3.53%	-2.34%	-2.43%	-0.33%	-3.77%	-6.40%	-4.44%
	P50ub		0.44%	1.67%	0.41%	0.59%	0.34%	-0.42%	0.59%	1.42%	0.29%
	P90ob		-2.90%	-6.74%	-6.37%	-6.36%	-7.01%	-6.91%	-7.28%	-8.77%	-7.17%
	P90ub		3.04%	4.93%	3.19%	3.62%	3.48%	1.40%	4.16%	5.51%	2.66%
Super Slow	P50ql		1.53%	-0.63%	-0.29%	0.08%	-0.67%	0.36%	-0.77%	-0.25%	**-0.94%**
	P90ql		0.75%	-4.07%	-5.04%	-3.51%	-4.91%	-4.98%	-5.20%	-4.58%	**-5.54%**
	P50ob		7.56%	-13.87%	-4.37%	-6.79%	-8.44%	-0.94%	-10.57%	-11.87%	-12.51%
	P50ub		0.22%	2.23%	0.56%	1.54%	1.01%	0.62%	1.31%	2.23%	1.54%
	P90ob		-1.13%	-15.49%	-15.21%	-13.02%	-15.29%	-13.69%	-16.19%	-17.16%	-17.78%
	P90ub		2.49%	6.44%	4.31%	5.23%	4.62%	3.04%	4.89%	6.99%	5.72%
Zero	P50ql		6.16%	-3.50%	-1.64%	-2.68%	-2.39%	-3.68%	-3.02%	-3.19%	**-4.37%**
	P90ql		32.57%	**-10.36%**	-6.76%	-3.71%	7.17%	-2.73%	-8.81%	-9.61%	**-10.05%**
	P50ob		35.06%	-28.00%	-14.44%	-22.11%	-19.85%	-26.07%	-24.59%	-25.96%	-32.91%
	P50ub		0.05%	1.67%	1.06%	1.42%	1.31%	1.05%	1.54%	1.61%	1.64%
	P90ob		72.17%	-35.67%	-26.22%	-15.05%	10.41%	-12.68%	-32.28%	-34.92%	-35.40%
	P90ub		1.65%	9.40%	8.46%	5.15%	4.65%	5.05%	9.52%	10.17%	9.74%

Table 3: Ablation studies for multi-magnitude mixture of experts on dataset D1.

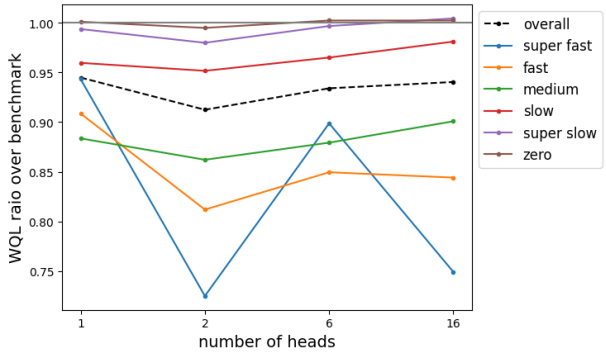
B.2 D2 Ablations

In this section, we present detailed experiment results for dataset D2. Table 4 summarizes the overall quantile loss from multiple models listed in Appendix B. Overall, V11 performs best on P50 and V19 performs the best on P90. In contrast to V11, V16 and V18, V19 improves over "Super Fast" category. Despite the fact that the fraction of "Super Fast" series is smaller than 0.01%, the impact of super fast products to the quantile loss metrics are large due to its high demand (large magnitude). For example, the loss of sales of such high demand product can easily lead to large monetary loss and customer dissatisfaction.

For all models, the major improvement on P90 comes from higher level of forecast, indicated by worse P90ob and better P90ub. With more than 90% zero series, the benchmark model (V9) tends to provide lower level of forecast due to jointly training of products with different velocity. With sparse series routing, we avoid such impact of zero series in training. As a result, the forecast for non-zero categories are now higher and have lower quantile loss.



(a) P50



(b) P90

Figure 4: WQL of all velocity groups over different number of heads in V19 model

		Version				
Category	**Metric**	**V9**	**V11**	**V16**	**V18**	**V19**
All	P50ql		**-1.36%**	0.87%	-0.45%	-0.77%
	P90ql		-3.39%	-1.18%	-5.49%	**-6.58%**
	P50ob		-0.33%	11.08%	45.75%	32.86%
	P50ub		-1.57%	-1.21%	-9.87%	-7.63%
	P90ob		6.97%	6.83%	26.88%	14.58%
	P90ub		-6.37%	-3.49%	-14.82%	-12.68%
Super Fast	P50ql		3.3%	14.2%	16.6%	**-2.0%**
	P90ql		-0.6%	7.9%	11.5%	**-10.3%**
	P50ob		-5.4%	28.7%	-8.8%	6.3%
	P50ub		5.1%	11.0%	22.0%	-3.8%
	P90ob		-8.8%	13.4%	9.2%	20%
	P90ub		-0.3%	7.6%	11.5%	-11.6%
Fast	P50ql		-2.9%	1.3%	-0.8%	**-3.9%**
	P90ql		-7.3%	-3.3%	-8.6%	**-14.8%**
	P50ob		-5.1%	1.2%	32.0%	34.1%
	P50ub		-2.1%	1.4%	-12.7%	-17.6%
	P90ob		6.6%	4.4%	31.6%	26.8%
	P90ub		-9.8%	-4.7%	-15.9%	-22.4%
Medium	P50ql		**-3.0%**	0.2%	-2.7%	-2.5%
	P90ql		-6.6%	-4.4%	**-13.0%**	-12.2%
	P50ob		-0.3%	11.1%	48.9%	32.5%
	P50ub		-3.9%	-3.4%	-19.8%	-14.0%
	P90ob		7.7%	10.1%	36.1%	22.3%
	P90ub		-11.8%	-9.7%	-30.9%	-24.7%
Slow	P50ql		**-0.5%**	0.8%	0.3	1.6%
	P90ql		-1.8%	0.5%	**-4.1%**	-3.6%
	P50ob		2.9%	19.7%	60.7%	33.7%
	P50ub		-1.0%	-2.1%	-9.0%	-3.3%
	P90ob		4.1%	4.0%	11.0%	12.5%
	P90ub		-4.9%	-1.3%	-3.9%	-11.9%
Super Slow	P50ql		0.2%	0.6%	0.3%	0.8%
	P90ql		-0.7%	0.0%	**-1.1%**	-0.3%
	P50ob		11.0%	31.2%	41.0%	26.8%
	P50ub		-0.1%	-0.3%	-1.0%	0.0%
	P90ob		5.6%	8.0%	11.0%	-1.4%
	P90ub		-2.1%	-1.8%	-3.9%	-0.1%
Zero	P50ql		0.9%	0.5%	0.4%	0.4%
	P90ql		0.0%	0.1%	**-0.3%**	0.2%
	P50ob		63.2%	41.2%	44.2%	30.1%
	P50ub		0.0%	-0.1%	-0.3%	0.0%
	P90ob		29.6%	18.6%	22.8%	-2.9%
	P90ub		-2.6%	-1.6%	-2.3%	0.5%

Table 4: Ablation Studies for different methods dealing with sparse series on D2 dataset

All multi-head models (V16, V18 and V19) in Table 4 have 6 heads. Here, we vary the number of heads for V19, the best performing model, and analyze the WQL in Figure 4. In general, we observe that WQL from 2 heads model is the lowest across almost all velocity groups in both P50 and P90. This suggests that for D2 dataset which has a large number of zero series, 2 heads are enough, and perform better than larger number of heads.