Appendix

A Parameterizaion of likelihood function for DeepAR

This paper follows the parameterization methods described in the references [6, 17] for l_θ of DeepAR. To be specific, we parameterize the model for a Gaussian likelihood distribution by using two affine functions:

$$l_\theta(z|\mathbf{h}) = \mathcal{N}(z; \mu_\theta(\mathbf{h}), \sigma_\theta(\mathbf{h})) \tag{7}$$
$$\mu_\theta(\cdot) = \mathbf{w}_\mu^T \mathbf{h} + b_\mu, \quad \sigma_\theta(\cdot) = \text{softplus}(\mathbf{w}_\sigma^T \mathbf{h} + b_\sigma)$$

where $\mathcal{N}(\cdot)$ is a likelihood function of normal distribution. The negative binomial distribution is parameterized as follows:

$$l_\theta(z|\mathbf{h}) = \mathcal{BN}(z; \mu_\theta(\mathbf{h}), \alpha_\theta(\mathbf{h})) = \frac{\Gamma(z + 1/\alpha)}{\Gamma(z+1)\Gamma(1/\alpha)} \Big(\frac{1}{1+\alpha\mu}\Big)^{\frac{1}{\alpha}} \Big(\frac{\alpha\mu}{1+\alpha\mu}\Big)^z \tag{8}$$
$$\mu_\theta(\cdot) = \text{softplus}(\mathbf{w}_\mu^T \mathbf{h} + b_\mu), \quad \alpha_\theta(\cdot) = \text{softplus}(\mathbf{w}_\alpha^T \mathbf{h} + b_\alpha)$$

B Estimation of joint distribution of future time series for DeepState

Unlike the auto-regressive models such as DeepAR, DeepState uses the observation values to compute the posterior distribution of latent state at each time step. Especially, DeepState integrates Kalman filtering [14, 15] into the model to estimate the analytical solution for the posterior: $p(\mathbf{l}_T^{(i)}|z_{1:T}^{(i)})$. From the estimated posterior, DeepState predict the conditional joint distribution of future time series as follows:

$$\begin{aligned} P(z_{T+1:T+\tau}^{(i)}|z_{1:T}^{(i)}, \mathbf{x}_{1:T+\tau}^{(i)}) &= \prod_{t=T+1}^{T+\tau} p(z_t^{(i)}|z_{1:t-1}^{(i)}; \Theta_{0:t-1}^{(i)}) \\ &= \int p(\mathbf{l}_T^{(i)}|z_{1:T}^{(i)}; \Theta_{0:T}^{(i)}) \left[\prod_{t=T+1}^{T+\tau} p(z_t^{(i)}|\mathbf{l}_t^{(i)}; \Theta_t^{(i)}) p(\mathbf{l}_t^{(i)}|\mathbf{l}_{t-1}^{(i)}; \Theta_t^{(i)})\right] d\mathbf{l}_{T:T+\tau}^{(i)}. \end{aligned} \tag{9}$$

Although the joint distribution is analytically tractable, it is evaluated by Monte Carlo approaches for the sake of implementation convenience following the reference paper.

C Additional Results

The mean and standard deviation value reported in following tables and figures are calculated over hyperparameter samples.

C.1 DeepAR

To illsutrate the effect of important hyperparameters to the performance of DeepAR, we report the result with different scaling methods, output distribution, context length, and number of RNN layers. As reported in Section 4.1, applying scaling method is essential to achieve desirable performance. The longer the context length is, the better the RMSE and the mean NQL are. The output distribution and number of RNN layers slightly varies the performance: negative binomial distribution is shown to be better fits to our EC dataset, and the performance drops as the number of layers increases probably due to the overfitting.

C.2 DeepState

Primarily, we address the performance of DeepState with fully-learnable (FL) SSMs and partially-learnable (PL) SSMs. We perform experiments with different context lengths and report each metrics's best mean score in Table C.2 and Fig. C.2. The performance of both models approximately converges after 56 days of context lengths. The model with FL SSMs shows inaccurate and imprecise performance, and this appears to be a problem of the inherent lack of representation power in the model and non-identifiability problem [18, 19] caused by SSMs' extreme flexibility.

Secondly, we compare the DeepState with different scaling methods: mean and median scalers. The performance degrades significantly when the scaler is not used. Empirically found that the model without scaler occasionally raise error when it calls Cholesky decomposition operator in Kalman filtering.

Table 2: RMSE, MAPE, and Mean NQL by scaling method for DeepAR and DeepState models.

Model	RMSE	MAPE	Mean NQL[0.1, 0.9]
DeepAR w/ mean scaler	81.37±2.60	0.303±0.013	0.140±0.008
DeepAR w/ median scaler	**81.06±1.07**	**0.300±0.008**	**0.139±0.006**
DeepAR w/o scaler	94.08±1.59	0.308±0.005	0.162±0.007
DeepState w/ mean Scaler	**87.88 ± 11.00**	**0.342 ± 0.022**	**0.169 ± 0.028**
DeepState w/ median scaler	114.68 ± 124.45	0.365 ± 0.118	0.211 ± 0.140
DeepState w/o scaler	150.91 ± 101.84	0.671 ± 0.297	0.427 ± 0.238

Table 3: RMSE, MAPE, and Mean NQL by scaling method for DeepAR and DeepState models.

Model	RMSE	MAPE	Mean NQL[0.1, 0.9]
DeepAR w/ 3-layer GRU	80.21 ± 0.34	**0.288 ± 0.002**	**0.135 ± 0.002**
DeepAR w/ 5-layer GRU	**79.91 ± 0.40**	0.294 ± 0.007	0.136 ± 0.001
DeepState w/ 3-layer GRU	120.45 ± 89.05	0.467 ± 0.268	0.271 ± 0.210
DeepState w/ 5-layer GRU	**115.20 ± 103.49**	**0.450 ± 0.203**	**0.266 ± 0.182**

Table 4: RMSE, MAPE, and Mean NQL by output distribution for DeepAR models.

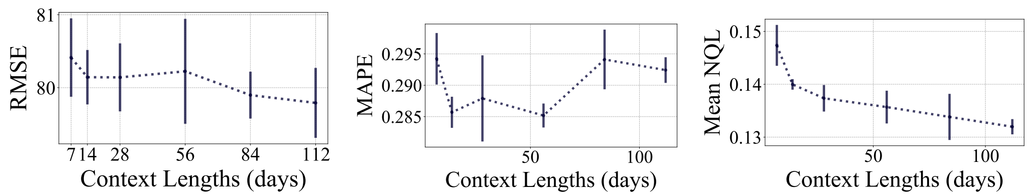
Model	RMSE	MAPE	Mean NQL[0.1, 0.9]
DeepAR w/ Gaussian output	**80.50±0.87**	0.297±0.004	0.140±0.006
DeepAR w/ Negative Binomial output	80.87±1.27	**0.296±0.008**	**0.138±0.006**

Table 5: RMSE, MAPE, and Mean NQL by SSM architecture for DeepState models.

Model	RMSE	MAPE	Mean NQL[0.1, 0.9]
DeepState w/ partially-learnable SSMs	**93.83 ± 11.51**	**0.373 ± 0.077**	**0.200 ± 0.042**
DeepState w/ fully-learnable SSMs	141.82 ± 131.80	0.545 ± 0.304	0.338 ± 0.256

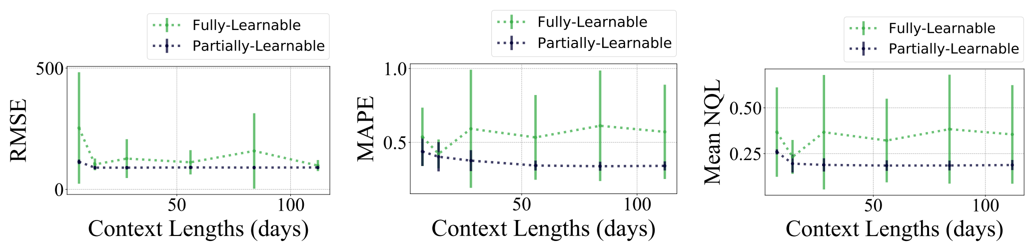
Table 6: RMSE, MAPE, and Mean NQL by the seasonality modes for Prophet models.

Model	RMSE	MAPE	Mean NQL[0.1, 0.9]
Prophet w/ additive seasonality	**83.89±1.98**	0.461±0.038	**0.202±0.009**
Prophet w/ multiplicative seasonality	97.02±16.85	**0.460±0.044**	0.222±0.010



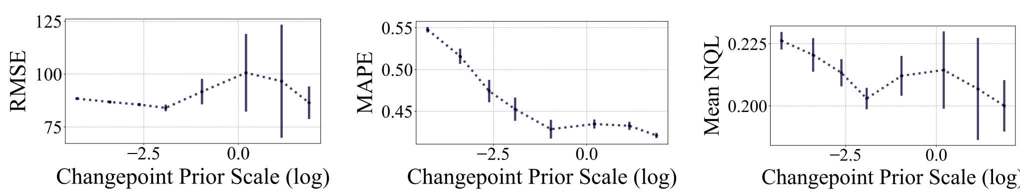
(a) RMSE (b) MAPE (c) Mean NQL[0.1, 0.9]

Figure 3: RMSE, MAPE, and Mean NQL by the context lengths for DeepAR models.



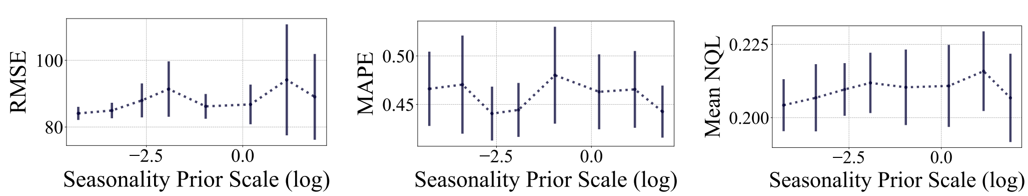
(a) RMSE (b) MAPE (c) Mean NQL[0.1, 0.9]

Figure 4: RMSE, MAPE, and Mean NQL by the context lengths for DeepState models.



(a) RMSE (b) MAPE (c) Mean NQL[0.1, 0.9]

Figure 5: RMSE, MAPE, and Mean NQL by the changepoint prior scales for Prophet models.



(a) RMSE (b) MAPE (c) Mean NQL[0.1, 0.9]

Figure 6: RMSE, MAPE, and Mean NQL by the seasonality prior scales for Prophet models.