MECATS: MIXTURE-OF-EXPERTS FOR PROBABILISTIC FORECASTS OF AGGREGATED TIME SERIES

Anonymous authors
Paper under double-blind review

ABSTRACT

We introduce a mixture of heterogeneous experts framework called MECATS, which simultaneously forecasts the values of a set of time series that are related through an aggregation hierarchy. Different types of forecasting models can be employed as individual experts so that the form of each model can be tailored to the nature of the corresponding time series. MECATS learns hierarchical relationships during the training stage to help generalize better across all the time series being modeled and also mitigates coherency issues that arise due to constraints imposed by the hierarchy. We further build multiple quantile estimators on top of the point forecasts. The resulting probabilistic forecasts are nearly coherent, distribution-free, and independent of the choice of forecasting models. We conduct a comprehensive evaluation on both point and probabilistic forecasts and also formulate an extension for situations where change points exist in sequential data. In general, our method is robust, adaptive to datasets with different properties, and highly configurable and efficient for large-scale forecasting pipelines.

1 INTRODUCTION

Forecasting time-series with hierarchical aggregation constraints is a common problem in many practically important applications (Hyndman et al., 2011; 2016; Lauderdale et al., 2020; Taieb et al., 2017; Zhao et al., 2016). For example, retail sales and inventory records are normally at different granularities such as product categories, store, city and state (Makridakis et al., 2020; Seeger et al., 2016). Generating forecasts for each aggregation level is necessary in developing high-level and detailed view of marketing insights. Another prominent example is population forecast at multiple time granularities such as monthly, quarterly, and yearly basis (Athanasopoulos et al., 2017). Normally, data at different aggregation levels possess distinct properties w.r.t. sparsity, noise distribution, sampling frequency etc. A well-generalized forecasting model should not only focus on the accuracy for each time series, but also exhibit coherency across the hierarchical structure. However, generating forecasts for each time series independently, using common multivariate forecasting models, does not lead to forecasts that satisfy the coherency requirement.

A major group of works for hierarchical time series forecasting employ a two-stage (or post-training) approach (Ben Taieb & Koo, 2019; Hyndman et al., 2011; 2016; Wickramasuriya et al., 2015), where base forecasts are firstly obtained for each time series followed by a reconciliation among these forecasts. The reconciliation step involves computing a weight matrix P taking into account the hierarchical structure, which linearly maps the base forecasts to coherent results. This approach guarantees coherent results but relies on strong assumptions (Gaussian error assumption; (Hyndman et al., 2011; 2016; Wickramasuriya et al., 2015) also require unbiased base forecast assumption). Moreover, computing P is expensive because of matrix inversion, making this method unsuitable for large-scale forecasting pipelines. Another line of work attempts to learn inter-level relationships during the model training stage. (Han et al., 2021) provides a controllable trade-off between forecasting accuracy of single time series and coherency across the hierarchy, which connects reconciliation with learned parameters of forecasting models. However, one needs to modify the objective functions of the individual models, which is not always possible, especially for encapsulated forecasting APIs.

In this work, we propose Mixture-of-Experts for Coherent and Aggregated Time Series (MECATS) that brings together the power of multiple heterogeneous models. MECATS can also learn hierarchical relationships to generate coherent results and significantly improve the overall accuracy compared

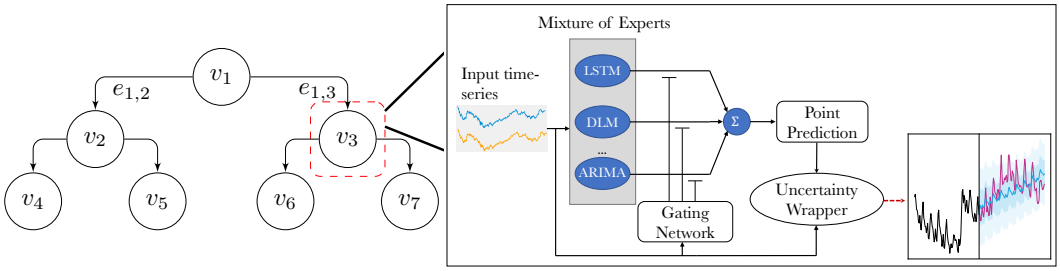


Figure 1: Overview of method: (left) a three-level structure for aggregated time series; bottom-level vertices: v_4 to v_7; aggregated-level vertices: v_1 to v_3; each vertex represent a uni-variate time series. (Right) the mixture-of-expert framework to generate probabilistic forecasts at each vertex.

with previous baselines. We also build an uncertainty wrapper on top of the mixture-of-experts (ME) predictions to simultaneously produce multiple quantile predictions that are reasonably coherent and independent to the choice of forecasting models. In summary, key contributions of `MECATS` include: 1. improve the performance of point prediction by learning to combine forecasts from a set of heterogeneous models; 2. build user-specified multiple quantile estimations that are model-free and approximately coherent. In addition, `MECATS` can also be applied on general multi-variate or univariate time series data, and is robust to abrupt changes in the data.

2 Related Works

Forecasting Models Models for time series span a wide range of categories. `ARIMA(p,d,q)` (Hyndman et al., 2008b; Zhang, 2003) is able to model many time series with trend and seasonality components, but its performance is normally poor on long term forecast and series with change-points. Dynamic regression models (Taylor & Letham, 2018) extend ARIMA by inclusion of other external variables, but the process requires expertise and experimentation. If all variables are concurrent, difficulties in forecasting external variables will result in poor forecasts. State-space models (Durbin & Koopman, 2012; Hyndman et al., 2008a; West et al., 1985) provides a more general and interpretable framework for modeling time series by sequentially updating information to give better estimates; Kalman filter (Welch et al., 1995) and exponential smoothing (Hyndman et al., 2008a) are both prominent examples. Deep Neural Networks (Becker et al., 2019; Krishnan et al., 2015; Lai et al., 2018; Rangapuram et al., 2018; Salinas et al., 2020; Wen et al., 2017) can improve the ability to model complex data with enough history. However, it is very difficult to obtain single model that works well in diverse situations. More discussions in hierarchical forecasting can be found in Appendix B and C.

Uncertainty Measurement A common choice to measure uncertainty for forecasting models is to add a Gaussian symmetric noise distribution to the point prediction. Bayesian methods make assumptions on prior and loss functions. However, these assumptions are not always fulfilled (Blundell et al., 2015; Iwata & Ghahramani, 2017; Kuleshov et al., 2018; Lakshminarayanan et al., 2016; Sun et al., 2019). Multiple observation noise models (Salinas et al., 2020) and normalizing flow (Durkan et al., 2019; Gopal & Key, 2021) can generalize to any noise distribution, but it is left to human's expertise to choose appropriate likelihood function. Quantile regression (Gasthaus et al., 2019; Han et al., 2021; Tagasovska & Lopez-Paz, 2018) avoids distributional assumptions and can be flexibly combined with many forecasting models. But extra efforts are needed to prevent quantile crossing, and (possibly) make each quantile coherent across the hierarchical structure.

Mixture of Experts ME (Jacobs et al., 1991; Masoudnia & Ebrahimpour, 2014; Yuksel et al., 2012) is a localized, cooperative ensemble learning (Hastie et al., 2009; Polikar, 2006; Rokach, 2010) method for enhancing the performance of machine learning algorithms. It captures a complex model space using a combination of simpler learners. The standard way to learn an ME model is to train a gating network and a set of experts using EM-based methods (Chen et al., 1999; Jordan & Jacobs, 1994; Ng & McLachlan, 2007; Xu et al., 1994; Yang & Ma, 2009). The gating network outputs either experts' weights (Chaer et al., 1997; 1998) or hard labels (Garmash & Monz, 2016; Shazeer et al., 2017). Prior works have studied ME for regular time series data, where ME showed success in allocating experts to the most suitable regions of input (Lu, 2006; Weigend et al., 1995). Hierarchical

ME has also been applied in the speech-processing literature (Chen et al., 1995; 1996a;b) for text dependent speaker identification. However, these works only employed experts of the same type with limited representation power (Cacciatore & Nowlan, 1994; Chaer et al., 1997; Weigend et al., 1995; Zeevi et al., 1997). Most recently, Bhatnagar et al. (2021) developed a library to forecast time series using heterogeneous models. The way they combine multiple predictions is through simple averaging or model selection based on a user defined metric. In this paper, we propose the `MECATS` framework that can not only learn the most suitable adaptive combination of a set of heterogeneous models, but also estimate arbitrary quantiles given the point prediction. `MECATS` is designed for multi-variate time series with hierarchical constraint, and its simplified variant also works for normal time series.

3 Probabilistic Forecasts of Aggregated Time Series

Figure 1 (left) shows a hierarchical graph structure with three levels. Each vertex represents time series data aggregated on different variables related through a domain-specific conceptual hierarchy (e.g., product categories, time granularities etc). We use $\{V, E\}$ to represent the graph structure, where $V := \{v_1, v_2, \ldots, v_7\}$ is the set of vertices and $E := \{e_{1,2}, e_{1,3}, \ldots, e_{3,7}\}$ is the set of edges. Let N be the number of vertices; $\{x^{v_i}_{1:T_i}\}_{i=1}^{N}$ be the set of univariate time series associated with the hierarchical structure, where $x^{v_i}_{1:T_i} = [x^{v_i}_1, x^{v_i}_2, \ldots, x^{v_i}_{T_i}]$, T_i is the forecasting start point, and $x^{v_i}_t \in \mathbb{R}$ denotes the value of i^{th} time series at time t. For the most common use case, we assume $e_{i,j} \in \{-1, 1\}$, and $x^{v_i}_t = \sum_{e_{i,j} \in E} e_{i,j}\, x^{v_j}_t$, which means data at parent vertices is (signed) summation of their children vertices. Ideally, the forecasts should satisfy:

$$P(x^{v_i}_{T_i+1:T_i+h} \mid x^{v_i}_{1:T_i}; \Theta_e), \quad \text{s.t. } |x^{v_i}_t - \sum_{e_{i,j} \in E} e_{i,j}\, x^{v_j}_t| = 0, \;\; \forall t \in [T_i + 1, T_i + h],$$

where h is the forecasting horizon and Θ_e represents the experts' parameters. Note that it is straightforward to extend linear aggregations to non-linear case and weighted edges. The $\{V, E\}$ representation is equivalent to the S matrix used in hierarchical forecasting literatures (Appendix B).

3.1 Point Forecasts using Mixture of Heterogeneous Experts

The ME framework enables a set of experts (Θ_e) and a gating network (Θ_g) to cooperate with each other on a complex task. We introduce a model set $\mathcal{M} = \{\mathsf{M}_1, \ldots, \mathsf{M}_\mathsf{L}\}$ which contains L experts with both high-variance and low-variance models. Let $\Theta = \{\Theta_e, \Theta_g\}$ be the collection of parameters, the prediction of $x^{v_i}_{T_i+1:T_i+h}$ can be written in terms of the experts and gating networks

$$P(\hat{x}^{v_i}_{T_i+1:T_i+h} \mid x^{v_i}_{1:T_i}, \Theta) = \sum_{l=1}^{\mathsf{L}} P(l \mid x^{v_i}_{1:T_i}, \Theta_g) \cdot P(\hat{x}^{v_i}_{T_i+1:T_i+h} \mid l, x^{v_i}_{1:T_i}, \Theta_e), \tag{1}$$

where $P(l \mid x^{v_i}_{1:T_i}, \Theta_g)$ in Eq (1) represents the l^{th} output of gating network, henceforth denoted by $g_l(x^{v_i}_{1:T_i}, \Theta_g)$ instead. Since the experts within $\mathcal{M}$ may have distinct ways to represent predictive uncertainty, it is unreasonable to simply combine their confidence intervals. We therefore restrict our discussion to point prediction $\mathbb{E}[\hat{x}^{v_i}_{T_i+1:T_i+h} \mid l, x^{v_i}_{1:T_i}, \Theta_e]$ from each expert for now. A commonly used method to train mixture-of-experts is through the EM algorithm (Jordan & Xu, 1995), where the latent variable and model parameters are iteratively estimated. Since many applications nowadays utilize complex models that are pre-trained in offline settings, we can simplify the EM procedure by disentangling the training of pre-specified experts and gates. Assume the time stamp $t_i \in [1, T_i]$, by having the set of experts $\{\mathsf{M}_l\}_{l=1}^{\mathsf{L}}$ trained offline on $x^{v_i}_{1:t_i}$, we first obtain the set of pre-trained experts that generate point predictions $\{\hat{x}^{v_i}_{t_i+1:T_i}(l)\}_{l=1}^{\mathsf{L}}$. The same set of time series is then processed by a sliding window that is used to train the gating network NN_g: $\mathrm{sw}(x^{v_i}_{1:t_i}) = [x^{v_i}_{1:\omega}, x^{v_i}_{2:\omega+1}, \ldots, x^{v_i}_{t_i-\omega+1:t_i}] \in \mathbb{R}^{(t_i-\omega+1)\times\omega\times 1}$, where ω is the window length, and $\{g_l(x^{v_i}_{1:T_i}, \Theta_g)\}_{l=1}^{\mathsf{L}} = \mathsf{NN}_g(\mathrm{sw}(x^{v_i}_{1:t_i}))$ is the set of weights.

We then train the gating network on a separate validation set. The outputs $\{g_l(x^{v_i}_{1:T_i}, \Theta_g)\}_{l=1}^{\mathsf{L}}$ captures the generalization ability of each pre-trained expert, i.e., higher $g_l(x^{v_i}_{1:T_i}, \Theta_g)$ indicates more importance of the l^{th} expert for node v_i. The objective for training the gating network at vertex v_i is

$$\mathcal{L}_{\text{recon}} = \mathcal{L}_{v_i}(\hat{x}^{v_i}_{t_i+1:T_i}, x^{v_i}_{t_i+1:T_i}) + \lambda_\mathsf{K} \cdot \|\hat{x}^{v_i}_{t_i+1:T_i} - \sum_{e_{i,j} \in E} e_{i,j}\, \hat{x}^{v_j}_{t_j+1:T_j}\|^2, \tag{2}$$

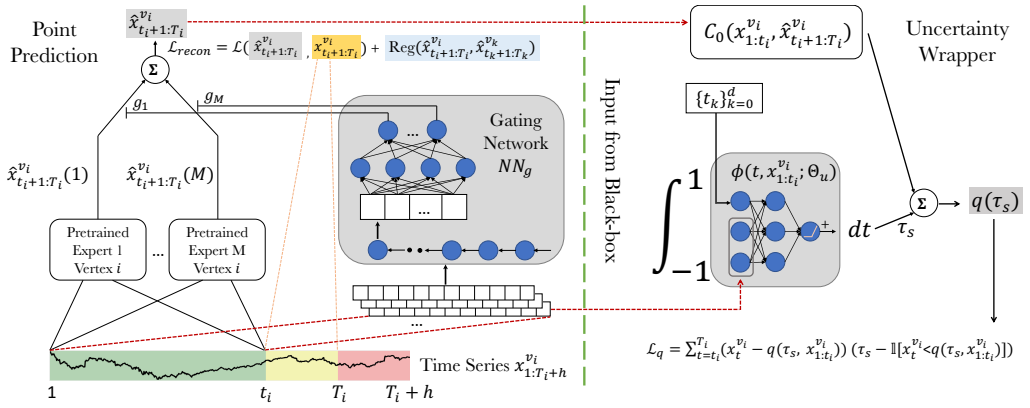


Figure 2: Model structure for generating probabilistic forecasts at vertex i. Left: point prediction generated by mixture-of-experts, $\mathcal{L}_{\text{recon}}$ is used to train gating network NN_g. Right: uncertainty wrapper built on top of black-box forecasting models to generate a set of quantile estimations.

where $\hat{x}^{v_i}_{t_i+1:T_i} = \sum_{l=1}^{\mathsf{L}} g_l(x^{v_i}_{1:T_i}, \Theta_g)\ \hat{x}^{v_i}_{t_i+1:T_i}(l)$, and K is the aggregation level of v_i. The loss $\mathcal{L}_{\text{recon}}$ not only involves forecast at v_i, but also forecasts at its child vertices. By minimizing $\mathcal{L}_{\text{recon}}$, we enable the gating network to learn across adjacent levels, and maximize the likelihood estimate of $x^{v_i}_{t_i+1:T_i}$ through controlling the weight on each expert. By using the regularized objective for gating network, we bypass adding regularization terms for each expert, which is often impractical. The overall structure of the point forecasting framework is shown in Figure 2 (left).

Eq (2) provides a controllable trade-off between coherency and accuracy at v_i, which helps the model generalize better, and is more effective than two-stage reconciliation methods when applying on large-scale hierarchical structures. This formulation can also be combined with a bottom-up training approach in (Han et al., 2021), where the gating networks at the bottom level are first trained, and use the forecasting results to progressively reconcile higher-level gating networks, till the root is reached. In contrast, one needs to reconcile both higher (previously visited) and lower-level model at an intermediate vertex if the top-down training method is applied, since other forecasts at that intermediate level might have changed. This bottom-up training procedure can be run in parallel on training vertices at the same aggregation level because they are mutually independent. Therefore, one can efficiently obtain coherent forecasts through one pass of the bottom-up training.

We now discuss how `MECATS` can provide accurate and coherent forecasts and how it different from other methods such as SHARQ (Han et al., 2021) and MinT (Hyndman et al., 2011).

Proposition 1. *Coherency Condition for* `MECATS`

Assume k experts are assigned to v_i, and its corresponding child vertices. Each expert generates forecast $\{\hat{x}^j_{v_i}\}_{j=1}^k$, which are not necessarily unbiased. WLOG, assume $\hat{x}^1_{v_i} \leq \hat{x}^2_{v_i} \leq \cdots \leq \hat{x}^k_{v_i}$. If ground truth $x_{v_i} \in [\hat{x}^1_{v_i}, \hat{x}^k_{v_i}]$, then `MECATS` *can generate coherent forecasts.*

Remark Note that other reconciliation methods like MinT, requires forecasting model at every vertex to be unbiased. This is not a realistic assumption given the likelihood of changing dynamics at different aggregation levels. SHARQ also requires unbiasedness assumption at the bottom level vertex. However, the assumption of `MECATS` is weaker as it only requires the true value to lie in the convex hull formed by forecasts from each expert. Therefore, the increased number of heterogeneous experts brings more robustness to the forecasts. More discussions can be found in Appendix A.

3.2 Model-Free Multiple Quantile Forecasts

Adding uncertainty measurements to point forecasts provide a more comprehensive view of such forecasts. We resort to quantiles for characterizing uncertainty. Since there is neither distributional nor model assumptions in our framework, one cannot simply compute the quantiles of a given distribution, or combine the quantile loss from the base models. The target of uncertainty measurement is therefore

a black-box. Specifically, for each vertex v_i, one only has access to the input $x^{v_i}_{1:t_i}$ and output $\hat{x}^{v_i}_{t_i+1:T_i}$ of the ME framework.

We first introduce conditional quantile regression for our forecasting problem, which is used to train the uncertainty wrapper. For each quantile value $\tau \in [0, 1]$ and input $x^{v_i}_{1:t_i}$, the aim of quantile regression is to estimate the τ^{th} quantile function $q(\tau, x^{v_i}_{1:t_i}) := \inf\{\hat{x}^{v_i}_t \in \mathbb{R} : F(\hat{x}^{v_i}_t \mid x^{v_i}_{1:t_i}, \Theta) \geq \tau\}$, where F is the CDF function of $p(\hat{x}^{v_i}_t \mid x^{v_i}_{1:t_i}, \Theta)$ for any forecasting time stamp $t \in [t_i, T_i]$. The quantile estimation at each τ is achieved by minimizing the pinball loss function $\mathcal{L}_q$ defined by

$$\mathcal{L}_q(y_{1:T}, \tau) = \sum_{t=t}^{T} (y_t - q(\tau, y_{1:t})) \cdot (\tau - \mathbb{1}[y_t < q(\tau, y_{1:t})]) \tag{3}$$

where $\mathbb{1}[\cdot]$ is the indicator function. Ideally, the estimated quantiles $q(\tau, x^{v_i}_{1:t_i})$ are monotonically increasing w.r.t. τ. We call it quantile crossing if this condition is not satisfied. Quantile crossing is a common problem when $n \geq 2$ quantile estimators are evaluated by minimizing e.g., $\sum_{s=1}^{n} \mathcal{L}_q(x^{v_i}_{1:T_i}, \tau_s)$. Solutions from prior works include imposing additional constraints in model training, or post-processing the multiple quantile estimations. We introduce a novel approach to simultaneously generate multiple quantile estimations without quantile crossing issues. The core idea is to model the derivative of quantile estimations using neural networks:

$$\{q(\tau_s, y_{1:t}, \hat{y}_{t+1:T}; \Theta_u)\}_{s=1}^{n},\ \forall \tau_s \in [0, 1] \Leftarrow \int_{-1}^{1} \phi(t, y_{1:t}; \Theta_u)\, dt + C_0(y_{1:t}, \hat{y}_{t+1:T}) \tag{4}$$

where $\phi(t, x^{v_i}_{1:t_i}; \Theta_u)$ is parameterized by neural networks (Θ_u) whose outputs are made to be strictly positive; $C_0(x^{v_i}_{1:t_i}, \hat{x}^{v_i}_{t_i+1:T_i})$ is a data-driven coefficient, and $[-1, 1]$ represents the range of $2\tau_s - 1$ with $\tau_s \in [0, 1]$. We can then obtain any quantile estimations from the result of integration. By integrating upon a strictly positive estimator, the results are therefore monotonically increasing within $[-1, 1]$, and also w.r.t. τ_s. The quantile estimators are then trained using a pinball loss in Eq (3) to learn the network ϕ. Eq (3) and (4) together define our multiple quantile forecaster (Figure 2 right). This formulation has also been studied in unconstrained monotonic neural networks (Wehenkel & Louppe, 2019). By using this work, we can only obtain one quantile estimation at a time instead of generating the entire range of quantiles simultaneously. Furthermore, Eq (4) only needs to access the input and output of the ME framework, which means it does not make any assumption on forecasting models or gating networks, and is compatible with any black-box model.

Training of Multi-Quantile Generator In order to train the integrand neural network ϕ, an efficient solution is to use Clenshaw-Curtis quadrature to approximate the integral $\int_{-1}^{1} \phi(t, x^{v_i}_{1:t_i}; \Theta_u)\, dt$. Given $\int_{-1}^{1} \phi(t)\, dt$, we can transform this problem by a change of variable: $\int_0^{\pi} \phi(\cos\theta) \sin\theta\, d\theta$. Here, we define a mapping $T_k : [-1, 1] \to \mathbb{R}$ and its recurrent relationship

$$T_0(z) := 1, \quad T_1(z) := z, \quad T_{k+1}(z) := 2zT_k(z) - T_{k-1}(z), \ \ k \geq 1. \tag{5}$$

The recurrence in Eq (5) can also be written as $T_k(\cos\theta) = \cos(k\theta)$. The mapping T_k is called *Chebyshev polynomials*. Note that we can write $\phi(\cos\theta)$ as its Chebyshev polynomial approximation: $\phi(\cos\theta) = \frac{1}{2}c_0(x^{v_i}_{1:t_i}) + \sum_{k=1}^{\infty} c_k(x^{v_i}_{1:t_i}) \cdot T_k(\cos\theta)$, where $\{c_k(x^{v_i}_{1:t_i})\}_{k=0}^{\infty}$ are coefficients. Normally, a finite number of terms can achieve sufficient precision for approximation:

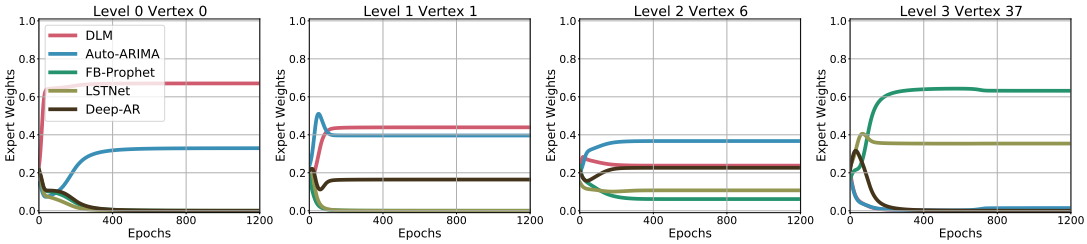
$$\phi(\cos\theta) \approx \frac{1}{2}\, c_0(x^{v_i}_{1:t_i}) + \sum_{k=1}^{d-1} c_k(x^{v_i}_{1:t_i}) \cdot T_k(\cos\theta) \tag{6}$$

is a d degree Chebyshev polynomial approxiamtion of $\phi(\cos\theta)$. Therefore we have

$$\int_0^{\pi} \phi(\cos\theta) \sin\theta\, d\theta = \int_0^{\pi} \phi(\cos\theta)\, d\cos\theta = \int_{-1}^{1} \left[\frac{1}{2}\, c_0(x^{v_i}_{1:t_i}) + \sum_{k=1}^{d-1} c_k(x^{v_i}_{1:t_i}) \cdot T_k(z) \right] dz,$$

where $z = 2\tau_s - 1, \forall \tau_s \in [0, 1]$. By the recurrent definition in Eq (5), the integral of the Chebyshev polynomial T_k corresponds to a new Chebyshev polynomial

$$\int T_0(z)\, dz = T_1(z), \quad \int T_1(z)\, dz = \frac{T_2(z)}{4} - \frac{T_0(z)}{4}, \quad \int T_k(z)\, dz = \frac{T_{k-1}(z)}{2(k-1)} - \frac{T_{k+1}(z)}{2(k+1)}.$$



(a) (b) (c) (d)

Figure 3: Pre-trained experts' weight curve from vertices at different aggregation levels. The most suitable representation of each time series can be found in diverse combinations of experts.

Therefore, the integration result can also be written in terms of Chebyshev polynomials: $\Phi(z) = \frac{1}{2}C_0(x^{v_i}_{1:t_i}) + \sum_{k=1}^{d-1} C_k(x^{v_i}_{1:t_i}) \cdot T_k(z)$, where the new Chebyshev coefficients $\{C_k(x^{v_i}_{1:t_i})\}_{k=1}^{d-1}$ can be obtained from the original coefficients

$$C_k(x^{v_i}_{1:t_i}) = \frac{c_{k-1}(x^{v_i}_{1:t_i}) - c_{k+1}(x^{v_i}_{1:t_i})}{4k}, \quad 0 < k < d-1, \quad C_{d-1}(x^{v_i}_{1:t_i}) = \frac{c_{d-2}(x^{v_i}_{1:t_i})}{4(d-1)}. \tag{7}$$

The above derivation shows that the integral operation in Eq (4) can be replaced by Chebyshev polynomial approximation below:

$$q(\tau_s, x^{v_i}_{1:t_i}, \hat{x}^{v_i}_{t_i+1:T_i}; \Theta_u) = \sum_{k=1}^{d-1} C_k(x^{v_i}_{1:t_i}) \cdot T_k(2\tau_s - 1) + C_0(x^{v_i}_{1:t_i}, \hat{x}^{v_i}_{t_i+1:T_i}). \tag{8}$$

Since the polynomials T_k can be obtained in a recurrent manner instead of explicitly computed (Clenshaw, 1955), estimating Chebyshev coefficients $\{c_k(x^{v_i}_{1:t_i})\}_{k=0}^{d-1}$ in Eq (6) is therefore a necessary step. The coefficients can be obtained from a linear transformation of the set of neural network outputs $\{\phi(t_k, x^{v_i}_{1:t_i}; \Theta_u)\}_{k=0}^{d-1}$ evaluated at $\{t_k\}_{k=0}^{d-1}$, where $t_k = \cos\left(\frac{\pi(k+\frac{1}{2})}{d}\right)$, $k \in [0, d)$ are the Chebyshev roots. The linear transformation ($\mathbb{R}^d \to \mathbb{R}^d$) is implemented by the Discrete Cosine Transform (DCT) algorithm, which gives $\{c_k(x^{v_i}_{1:t_i})\}_{k=0}^{d-1}$ by transforming the set of output scalars.

Connection to Point Forecast We now link the multiple quantile forecasts to point forecast by imposing the point forecast as a constraint of the distribution formulated by all the quantiles. Assume that $\hat{x}^{v_i}_{t_i+1:T_i}$ is constrained to be the median value of the quantiles, i.e., $q(0.5, x^{v_i}_{1:t_i}, \hat{x}^{v_i}_{t_i+1:T_i}) = \hat{x}^{v_i}_{t_i+1:T_i}$, we can obtain the value of coefficient C_0 as

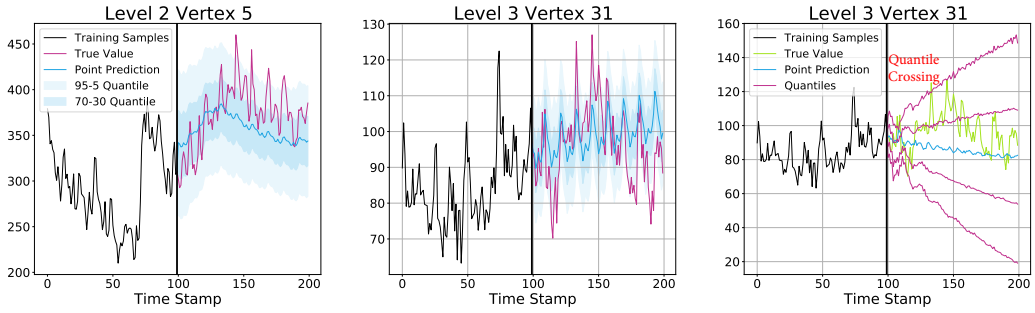
$$C_0(x^{v_i}_{1:t_i}, \hat{x}^{v_i}_{t_i+1:T_i}) = 2 \cdot \hat{x}^{v_i}_{t_i+1:T_i} - 2 \sum_{k=1, k \text{ even}}^{d-1} (-1)^{k/2} C_k(x^{v_i}_{1:t_i}). \tag{9}$$

In fact, one can impose the point forecast to be any summary statistic of the quantile distribution and the form of C_0 can be derived accordingly (details in Appendix D). Building this connection to the combined point forecast regularizes each quantile estimation to be more coherent.

In summary, MECATS shows some very good properties compared to previous methods. It can borrow strength from arbitrary forecasting models in a plug-and-play manner, leading to accurate and robust forecasts in all situations. The connection between its point forecasts and quantile estimations can also improve the performance of probabilistic forecasts. Additionally, MECATS tackles model dependency and quantile crossing problems that have not been addressed by SHARQ and also does not need strong assumptions made by the two-stage reconciliation methods.

4 Experiments

In this section, we evaluate MECATS from different aspects. Our experiments include: 1. coherent forecasts on hierarchically aggregated time series data (section 4.1 - 4.3); 2. real-time forecasting under change of time series dynamics (section 4.4). Our results show that MECATS can significantly improve performance and is adaptive to temporal sequences with different properties.



(a) (b) (c)

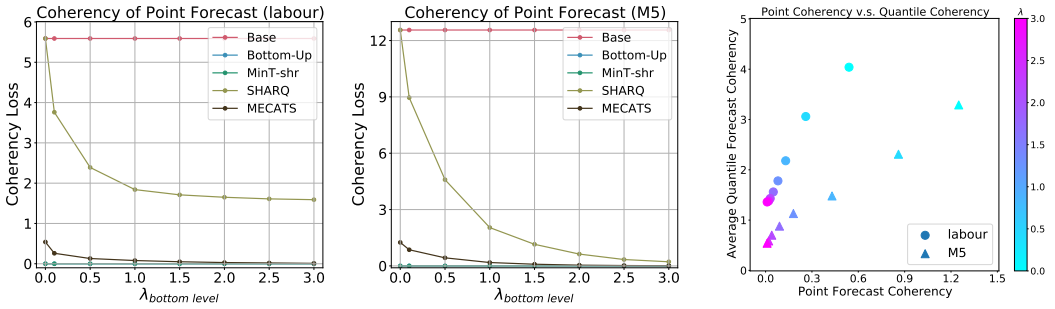
Figure 4: (a), (b) Probabilistic forecasting examples generated by MECATS at vertex 5 and 31 when $\tau_s = [0.05, 0.3, 0.5, 0.7, 0.95]$. (c) SHARQ at same τ_s, results showing mild quantile crossing.

4.1 Point Forecast

We implemented various combinations of different forecasting models and reconciliation methods. For forecasting models, both single models and mixtures are evaluated. We employed linear auto-regressive model (AR), RNN-GRU (Chung et al., 2014), LSTNet (Lai et al., 2018), and DeepAR (Salinas et al., 2020) as the single models, and a mix of dynamic linear models (DLM) (West & Harrison, 2006), Auto-ARIMA (Hyndman et al., 2008b), Facebook Prophet (Taylor & Letham, 2018) with LSTNet and DeepAR as the chosen set of experts. We also implemented ensemble averaging and model selection (MS) (Bhatnagar et al., 2021) to combine predictions. Since LSTNet and DeepAR are global models (Januschowski et al., 2020), which learn across a set of time series, we only need to train one model for the hierarchy. The rest of the experts are trained on univariate time series, where we assign one model for each vertex. In other words, one can include both global models and univariate models in MECATS. However, we will not discuss how to make the best selection of each heterogeneous expert, which is beyond the scope of this paper. In terms of reconciliation methods, we implemented state-of-the-art method SHARQ (in-training) (Han et al., 2021), post-training reconciliations (Wickramasuriya et al., 2015) and base forecast for comparison. We conducted our experiments on multi-step forecast using 4 public time-series datasets with hierarchical aggregation: Australian Labour, M5, Wikipedia webpage views and AE-Demand. Due to the space constraint, we defer the results of Wikipedia and AE-Demand datasets into Appendix H. Figure 3 illustrates the weight evolution for each expert as the gating network is trained and reconciled on vertices at each level using the Australian Labour dataset. According to the plots, the gating network emphasizes the DLM expert for time series at the top aggregated level; as the level becomes lower, DLM becomes less dominant and other experts begin to carry more weights at some vertices. Finally, at a bottom-level vertex where data becomes noisy, a different set of experts takes charge of the forecast. Results measured by MASE (Hyndman & Koehler, 2006) averaged across each random experiment are shown in Table 1 (also see Appendix H for full results). MECATS generates the best multi-step point forecast among all combinations of forecasting models and reconciliation methods. Our results show that the set of heterogeneous experts brought together by ME can significantly improve the forecasting quality compared to using a single model. In addition, we have also validated the effectiveness of in-training reconciliation for enhancing the generalization of different forecasting models.

4.2 Multiple Quantile Forecasts

We now evaluate the multi-quantile generator built on top of our multi-step point forecasts. The neural network models $\phi(t_k, x^{v_i}_{1:t_i}; \Theta_u)$ are trained on mini-batches of training data $\text{sw}(x^{v_i}_{1:t_i})$ generated by sliding windows, along with the set of Chebyshev roots $\{t_k\}_{k=0}^{d-1}$. We train d number of models ϕ where each of them is evaluated at one t_k. The Chebyshev polynomial coefficients $\{c_k(x^{v_i}_{1:t_i})\}_{k=0}^{d-1}$ in Eq (6) can then be obtained by applying type-II DCT transformations into d model outputs, where further computations using $\{c_k(x^{v_i}_{1:t_i})\}_{k=0}^{d-1}$ in Eq (7) and (8) produce quantile estimations $q(\tau_s), \forall \tau_s \in [0, 1]$. Procedures for generating point forecasts remain unchanged as the multi-quantile generator is an independent module. Figure 4 demonstrates probabilistic forecasting results on arbitrary chosen vertices in the M5 dataset. The multi-quantile forecasts can well-capture the



(a) (b) (c)

Figure 5: (a), (b) Coherent loss of point forecast w.r.t. regularization strength λ on labour and M5 dataset. (c) Relationship between point forecast coherency and average of quantile coherency.

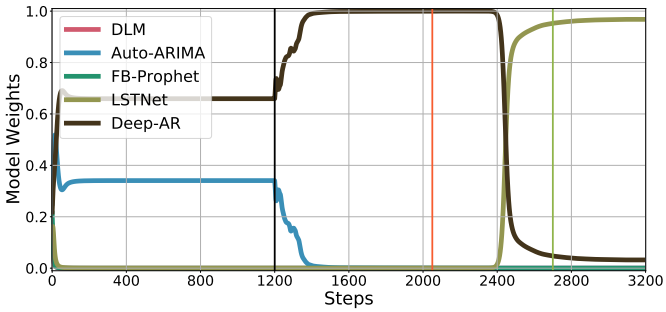
Model \ Recon	SHARQ	Base	Post-training		
			MinT-shr	MinT-sam	MinT-ols
AR	75.59±.50 (.194)	77.74±.33 (.207)	85.32±.60 (.197)	93.77±1.3 (.200)	91.72±1.3 (.202)
RNN	61.73±.32 (.126)	66.64±.50 (.157)	68.28±.77 (.132)	74.31±.31 (.133)	74.41±.81 (.136)
LSTNet	61.22±.50 (.113)	65.83±.53 (.137)	67.94±.54 (.125)	72.89±.49 (.130)	72.55±.81 (.130)
DeepAR	60.39±.34 (.121)	62.81±.38 (.134)	64.36±.43 (.135)	68.60±.69 (.141)	69.13±.82 (.147)
Average	63.77±.59 (.122)	64.72±.16 (.136)	64.21±.18 (.141)	69.14±.26 (.142)	68.98±.32 (.141)
MS	58.04±.35 (.120)	62.81±.38 (.134)	64.36±.43 (.135)	68.60±.69 (.141)	69.13±.82 (.147)
MECATS	**49.47±.33** (**.081**)	54.73±.08 (.089)	52.42±.19 (.098)	56.83±.06 (.104)	57.94±.24 (.103)
AR	70.16±.47 (.248)	70.94±.47 (.263)	82.05±.51 (.265)	85.69±.59 (.267)	85.60±.47 (.269)
RNN	64.18±.34 (.114)	66.35±.45 (.183)	73.67±.44 (.159)	77.82±.27 (.168)	77.26±.45 (.168)
LSTNet	65.78±.49 (.104)	66.54±.46 (.159)	72.75±.40 (.135)	77.24±.25 (.154)	77.45±.24 (.153)
DeepAR	62.16±.20 (.112)	63.22±.30 (.126)	71.22±.42 (.131)	74.35±.41 (.138)	74.59±.43 (.132)
Average	64.67±.39 (.137)	65.92±.21 (.139)	73.26±.19 (.146)	78.51±.09 (.152)	77.93±.16 (.151)
MS	60.87±.32 (.123)	63.01±.11 (.124)	69.74±.12 (.129)	72.42±.32 (.134)	73.69±.15 (.134)
MECATS	**50.97±.31** (**.085**)	57.34±.07 (.087)	62.46±.33 (.096)	69.39±.24 (.097)	68.82±.29 (.106)

Table 1: Forecasting performance measured by averaged $\text{MASE}^{\downarrow}$ and $\text{CRPS}^{\downarrow}$ (within bracket) on Australian Labour (upper) and M5 competition (lower) data. All experiments are repeated 5 times.

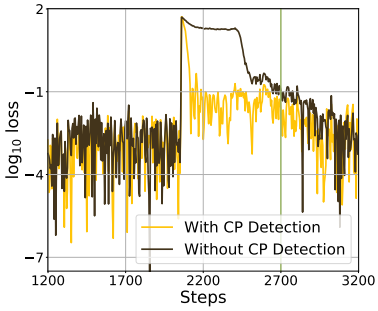
uncertainty in difference time series. As a comparison, we also show an example of SHARQ trained with LSTNet model, where the multi-quantile estimations are less constrained by the mean forecast. Note that SHARQ is a very strong baseline, providing results superior to specific observation noise model such as Gaussian or negative binomial distribution. However, quantile crossing is possible since SHARQ cannot guarantee strict monotonicity w.r.t τ_s. We use CRPS (Matheson & Winkler, 1976) to measure the probabilistic forecasts as shown in Table 1, our baseline uncertainty estimations include Gaussian noise for post-training methods and the multiple quantile regression of SHARQ.

4.3 Coherency Analysis

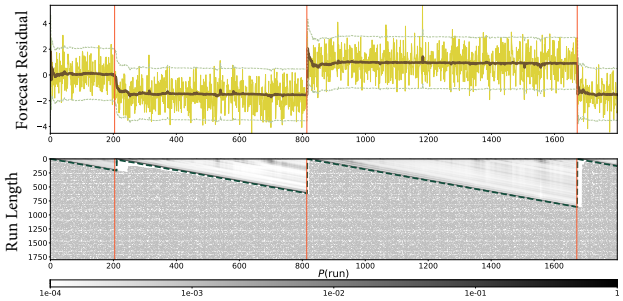
Proposition 1 states that `MECATS` can generate coherent point forecasts given the ground truth time series lies in the convex hull formed by forecasts from each expert. However, during empirical studies, we found it not easy to eliminate the coherent loss, due to various reasons such as inappropriate choice of experts or bad training of gating networks. We then tune parameter λ that controls the strength of penalty for coherent loss. We observe that by choosing λ in an appropriate range, coherent loss can be mitigated without sacrificing forecasting accuracy. Figure 5 (a), (b) have shown that `MECATS` can achieve nearly coherent results. Note that post-processing methods such as MinT, can reduce the coherent loss to a very small value at the cost of decreased accuracy, since the summation of bottom-level forecasts is forced to be aligned with higher-level forecasts. Figure 5 (c) shows that the improved coherency of point forecast can positively affect the coherency of quantiles. However, since the additive property does not hold for quantiles (Han et al., 2021), there still exists non-zero coherent loss in certain quantiles. More results can be found in Appendix H.



(a) Experts' weights with change-points



(b) Online predictive loss



(c) Online change-point detection on WTI Oil Price data

MSE	BOCPD-MS	ME	ME-CP
EIA	3.15±.04	2.67±.02	**2.65±.02**
Snow	0.68±.02	0.60±.01	**0.58±.01**
Nile	0.55±.09	0.57±.04	**0.54±.05**
Bee	1.74±.22	1.32±.13	**1.29±.12**

(d) Predictive loss on real data

Table 2: Results on improving mixture of heterogeneous experts using online change-point detection.

4.4 Forecasting Under Change of Dynamics

We show that MECATS can be robust to change of dynamics in time series data. Since ME has been proven to be reasonably adaptive to abrupt changes (Chaer et al., 1997; Weigend et al., 1995), we further alleviate the jump of loss during the transient period in adapting to new dynamics with the help of Bayesian online change-point detection (BOCPD) (Adams & MacKay, 2007): after detecting change-points, we re-initialize the weights to its average value. We assume new samples come with a batch size of 1, and both gating network and experts' parameters $\Theta = \{\Theta_g, \Theta_e\}$ are updated at each step. Table 2 (a) demonstrates the original gating networks' behavior when change-point occurs at step 2050 (step 1200 is the starting point of online updates) on simulated data. The network adapts to the new dynamics at around step 2700. With the help of change-point detection, loss during this transient period can be greatly reduced (Table 2 (b)). We follow the procedure in the change-point detection literature (Adams & MacKay, 2007; Knoblauch & Damoulas, 2018) to evaluate our method on several real-world sequential data. Table 2 (c) and (d) show that our method can detect most changes and also quantitatively outperform BOCPD with model selection (Knoblauch & Damoulas, 2018), detailed experiment settings and backgrounds can be found in the Appendix E.

5 Conclusion

Forecasting hierarchically aggregated time series is a key but understudied problem with many applications. In this work, we propose MECATS that utilizes mixture-of-experts to adaptively bring together the power of multiple forecasting models. MECATS learns to combine a set of heterogeneous experts while also performs multiple quantile estimations in a model-free manner. MECATS has demonstrated superior results in both accuracy and coherency in hierarchical time series forecasting, and is robust to change of dynamics in sequential data. The substantially more flexibility that MECATS offers in terms of using a customized mix of heterogeneous experts for improving a specific time series forecast comes at an increased computational cost. It also introduces a few more hyper-parameters. Our future research will focus on ameliorating the costs when a large number of forecasts need to be done. In particular, we would like to further investigate situations where data resources are more constrained, such as short sequences or sequences with missing entries.

Ethics Statement Forecasts that are coherent and accurate, adaptive to changes in the nature of the time series, and also yield confidence intervals, result in solutions that are more reliable and trustworthy, critical requirements for widespread adoption and safe deployment. So our social impact is mostly positive since robustness and reliability of a predictive model are important aspects of responsible AI solutions. In addition, using a modular structure like mixture of heterogeneous experts may allow the use of simpler base models that are more interpretable by humans, without sacrificing the accuracy of the overall solution. In addition, this work does not involve human subjects, and also does not raise any discrimination/bias/fairness concerns. All information of public datasets, models and hyper-parameters can be found in Appendix F.

Reproducibility We have submitted the source code of this work in a zip file as part of the supplementary materials. Implementation instructions can also be found under the same file.

REFERENCES

Ryan Prescott Adams and David JC MacKay. Bayesian online changepoint detection. *arXiv preprint arXiv:0710.3742*, 2007.

Samaneh Aminikhanghahi and Diane J Cook. A survey of methods for time series change point detection. *Knowledge and information systems*, 51(2):339–367, 2017.

George Athanasopoulos, Rob J Hyndman, Nikolaos Kourentzes, and Fotios Petropoulos. Forecasting with temporal hierarchies. *European Journal of Operational Research*, 262(1):60–74, 2017.

Philipp Becker, Harit Pandya, Gregor Gebhardt, Cheng Zhao, C James Taylor, and Gerhard Neumann. Recurrent kalman networks: Factorized inference in high-dimensional deep feature spaces. In *International Conference on Machine Learning*, pp. 544–552. PMLR, 2019.

Souhaib Ben Taieb and Bonsoo Koo. Regularized regression for hierarchical forecasting without unbiasedness conditions. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1337–1347, 2019.

Aadyot Bhatnagar, Paul Kassianik, Chenghao Liu, Tian Lan, Wenzhuo Yang, Rowan Cassius, Doyen Sahoo, Devansh Arpit, Sri Subramanian, Gerald Woo, et al. Merlion: A machine learning library for time series. *arXiv preprint arXiv:2109.09265*, 2021.

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural network. In *International Conference on Machine Learning*, pp. 1613–1622. PMLR, 2015.

Timothy W Cacciatore and Steven J Nowlan. Mixtures of controllers for jump linear and non-linear plants. In *Advances in neural information processing systems*, pp. 719–726, 1994.

Wassim S Chaer, Robert H Bishop, and Joydeep Ghosh. A mixture-of-experts framework for adaptive kalman filtering. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 27 (3):452–464, 1997.

Wassim S Chaer, Robert H Bishop, and Joydeep Ghosh. Hierarchical adaptive kalman filtering for interplanetary orbit determination. *IEEE transactions on aerospace and electronic systems*, 34(3): 883–896, 1998.

Ke Chen, Dahong Xie, and Huisheng Chi. Speaker identification based on the time-delay hierarchical mixture of experts. In *Proceedings of ICNN'95-International Conference on Neural Networks*, volume 4, pp. 2062–2066. IEEE, 1995.

Ke Chen, Dahong Xie, and Huisheng Chi. Combine multiple time-delay hmes for speaker identification. In *Proceedings of International Conference on Neural Networks (ICNN'96)*, volume 4, pp. 2015–2020. IEEE, 1996a.

Ke Chen, Dahong Xie, and Huisheng Chi. A modified hme architecture for text-dependent speaker identification. *IEEE Transactions on Neural Networks*, 7(5):1309–1313, 1996b.

Ke Chen, Lei Xu, and Huisheng Chi. Improved learning algorithms for mixture of experts in multiclass classification. *Neural networks*, 12(9):1229–1252, 1999.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

Charles W Clenshaw. A note on the summation of chebyshev series. *Mathematics of Computation*, 9 (51):118–120, 1955.

Germund Dahlquist and Åke Björck. *Numerical methods in scientific computing, volume I*. SIAM, 2008.

James Durbin and Siem Jan Koopman. *Time series analysis by state space methods*. Oxford university press, 2012.

Conor Durkan, Artur Bekasov, Iain Murray, and George Papamakarios. Neural spline flows. *arXiv preprint arXiv:1906.04032*, 2019.

Ekaterina Garmash and Christof Monz. Ensemble learning for multi-source neural machine translation. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pp. 1409–1418, 2016.

Jan Gasthaus, Konstantinos Benidis, Yuyang Wang, Syama Sundar Rangapuram, David Salinas, Valentin Flunkert, and Tim Januschowski. Probabilistic forecasting with spline quantile function rnns. In *The 22nd international conference on artificial intelligence and statistics*, pp. 1901–1910. PMLR, 2019.

Achintya Gopal and Aaron Key. Normalizing flows for calibration and recalibration, 2021. URL `https://openreview.net/forum?id=H8VDvtm1ij8`.

Xing Han, Sambarta Dasgupta, and Joydeep Ghosh. Simultaneously reconciled quantile forecasting of hierarchically related time series. In *International Conference on Artificial Intelligence and Statistics*, pp. 190–198. PMLR, 2021.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. Ensemble learning. In *The elements of statistical learning*, pp. 605–624. Springer, 2009.

Rob Hyndman, Anne B Koehler, J Keith Ord, and Ralph D Snyder. *Forecasting with exponential smoothing: the state space approach*. Springer Science & Business Media, 2008a.

Rob J Hyndman and Anne B Koehler. Another look at measures of forecast accuracy. *International journal of forecasting*, 22(4):679–688, 2006.

Rob J Hyndman, Yeasmin Khandakar, et al. Automatic time series forecasting: the forecast package for r. *Journal of statistical software*, 27(3):1–22, 2008b.

Rob J Hyndman, Roman A Ahmed, George Athanasopoulos, and Han Lin Shang. Optimal combination forecasts for hierarchical time series. *Computational statistics & data analysis*, 55(9): 2579–2589, 2011.

Rob J Hyndman, Alan J Lee, and Earo Wang. Fast computation of reconciled forecasts for hierarchical and grouped time series. *Computational statistics & data analysis*, 97:16–32, 2016.

Tomoharu Iwata and Zoubin Ghahramani. Improving output uncertainty estimation and generalization in deep learning via neural network gaussian processes. *arXiv preprint arXiv:1707.05922*, 2017.

Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87, 1991.

Tim Januschowski, Jan Gasthaus, Yuyang Wang, David Salinas, Valentin Flunkert, Michael Bohlke-Schneider, and Laurent Callot. Criteria for classifying forecasting methods. *International Journal of Forecasting*, 36(1):167–177, 2020.

Michael I Jordan and Robert A Jacobs. Hierarchical mixtures of experts and the em algorithm. *Neural computation*, 6(2):181–214, 1994.

Michael I Jordan and Lei Xu. Convergence results for the em approach to mixtures of experts architectures. *Neural networks*, 8(9):1409–1431, 1995.

Jeremias Knoblauch and Theodoros Damoulas. Spatio-temporal bayesian on-line changepoint detection with model selection. In *International Conference on Machine Learning*, pp. 2718–2727. PMLR, 2018.

Rahul G Krishnan, Uri Shalit, and David Sontag. Deep kalman filters. *arXiv preprint arXiv:1511.05121*, 2015.

Volodymyr Kuleshov, Nathan Fenner, and Stefano Ermon. Accurate uncertainties for deep learning using calibrated regression. In *International Conference on Machine Learning*, pp. 2796–2804. PMLR, 2018.

Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long-and short-term temporal patterns with deep neural networks. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pp. 95–104, 2018.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *arXiv preprint arXiv:1612.01474*, 2016.

Benjamin E Lauderdale, Delia Bailey, Jack Blumenau, and Douglas Rivers. Model-based pre-election polling for national and sub-national outcomes in the us and uk. *International Journal of Forecasting*, 36(2):399–413, 2020.

Zhiwu Lu. A regularized minimum cross-entropy algorithm on mixtures of experts for time series prediction and curve detection. *Pattern Recognition Letters*, 27(9):947–955, 2006.

S Makridakis, E Spiliotis, and V Assimakopoulos. The m5 accuracy competition: Results, findings and conclusions. *Int J Forecast*, 2020.

Saeed Masoudnia and Reza Ebrahimpour. Mixture of experts: a literature survey. *Artificial Intelligence Review*, 42(2):275–293, 2014.

James E Matheson and Robert L Winkler. Scoring rules for continuous probability distributions. *Management science*, 22(10):1087–1096, 1976.

Kevin P Murphy. Conjugate bayesian analysis of the gaussian distribution. *def*, $1(2\sigma 2)$:16, 2007.

Shu-Kay Ng and Geoffrey J McLachlan. Extension of mixture-of-experts networks for binary classification of hierarchical data. *Artificial Intelligence in Medicine*, 41(1):57–67, 2007.

Robi Polikar. Ensemble based systems in decision making. *IEEE Circuits and systems magazine*, 6 (3):21–45, 2006.

Syama Sundar Rangapuram, Matthias W Seeger, Jan Gasthaus, Lorenzo Stella, Yuyang Wang, and Tim Januschowski. Deep state space models for time series forecasting. *Advances in neural information processing systems*, 31:7785–7794, 2018.

Syama Sundar Rangapuram, Lucien D Werner, Konstantinos Benidis, Pedro Mercado, Jan Gasthaus, and Tim Januschowski. End-to-end learning of coherent probabilistic forecasts for hierarchical time series. In *International Conference on Machine Learning*, pp. 8832–8843. PMLR, 2021.

Lior Rokach. Ensemble-based classifiers. *Artificial intelligence review*, 33(1):1–39, 2010.

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.

Matthias Seeger, David Salinas, and Valentin Flunkert. Bayesian intermittent demand forecasting for large inventories. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pp. 4653–4661, 2016.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017.

Shengyang Sun, Guodong Zhang, Jiaxin Shi, and Roger Grosse. Functional variational bayesian neural networks. *arXiv preprint arXiv:1903.05779*, 2019.

Natasa Tagasovska and David Lopez-Paz. Single-model uncertainties for deep learning. *arXiv preprint arXiv:1811.00908*, 2018.

Souhaib Ben Taieb, James W Taylor, and Rob J Hyndman. Coherent probabilistic forecasts for hierarchical time series. In *International Conference on Machine Learning*, pp. 3348–3357. PMLR, 2017.

Sean J Taylor and Benjamin Letham. Forecasting at scale. *The American Statistician*, 72(1):37–45, 2018.

Charles Truong, Laurent Oudre, and Nicolas Vayatis. Selective review of offline change point detection methods. *Signal Processing*, 167:107299, 2020.

Gerrit JJ van den Burg and Christopher KI Williams. An evaluation of change point detection algorithms. *arXiv preprint arXiv:2003.06222*, 2020.

Antoine Wehenkel and Gilles Louppe. Unconstrained monotonic neural networks. *arXiv preprint arXiv:1908.05164*, 2019.

Andreas S Weigend, Morgan Mangeas, and Ashok N Srivastava. Nonlinear gated experts for time series: Discovering regimes and avoiding overfitting. *International journal of neural systems*, 6 (04):373–399, 1995.

Greg Welch, Gary Bishop, et al. An introduction to the kalman filter. 1995.

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

Mike West and Jeff Harrison. *Bayesian forecasting and dynamic models*. Springer Science & Business Media, 2006.

Mike West, P Jeff Harrison, and Helio S Migon. Dynamic generalized linear models and bayesian forecasting. *Journal of the American Statistical Association*, 80(389):73–83, 1985.

Shanika L Wickramasuriya, George Athanasopoulos, Rob J Hyndman, et al. Forecasting hierarchical and grouped time series through trace minimization. *Department of Econometrics and Business Statistics, Monash University*, 105, 2015.

Shanika L Wickramasuriya, George Athanasopoulos, and Rob J Hyndman. Optimal forecast reconciliation for hierarchical and grouped time series through trace minimization. *Journal of the American Statistical Association*, 114(526):804–819, 2019.

Lei Xu, Michael Jordan, and Geoffrey E Hinton. An alternative model for mixtures of experts. *Advances in neural information processing systems*, 7:633–640, 1994.

Yan Yang and Jinwen Ma. A single loop em algorithm for the mixture of experts architecture. In *International Symposium on Neural Networks*, pp. 959–968. Springer, 2009.

Seniha Esen Yuksel, Joseph N Wilson, and Paul D Gader. Twenty years of mixture of experts. *IEEE transactions on neural networks and learning systems*, 23(8):1177–1193, 2012.

Assaf J Zeevi, Ron Meir, and Robert J Adler. Time series prediction using mixtures of experts. *Advances in neural information processing systems*, pp. 309–318, 1997.

G Peter Zhang. Time series forecasting using a hybrid arima and neural network model. *Neurocomputing*, 50:159–175, 2003.

Liang Zhao, Feng Chen, Chang-Tien Lu, and Naren Ramakrishnan. Multi-resolution spatial event forecasting in social media. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pp. 689–698. IEEE, 2016.

A PROOF OF PROPOSITION

Proposition 1. *Coherency Condition for* `MECATS`

Assume k experts are assigned to v_i, and its corresponding child vertices. Each expert generates forecast $\{\hat{x}^j_{v_i}\}^k_{j=1}$, which are not necessarily unbiased. WLOG, assume $\hat{x}^1_{v_i} \leq \hat{x}^2_{v_i} \leq \cdots \leq \hat{x}^k_{v_i}$. If ground truth $x_{v_i} \in [\hat{x}^1_{v_i}, \hat{x}^k_{v_i}]$, then `MECATS` *can generate coherent forecasts.*

PROOF. Given vertices $V = \{v_1, v_2, \ldots, v_n\}$, where v_1 is the root vertex and $\{v_j\}^n_{j=2}$ are its children. For each expert, we obtain the following hierarchical forecasts

$$\hat{x}^1_{v_1} = \sum_{j=2}^{n} \hat{x}^1_{v_j} + \delta_1; \quad \hat{x}^2_{v_1} = \sum_{j=2}^{n} \hat{x}^2_{v_j} + \delta_2 \quad \ldots \quad \hat{x}^k_{v_1} = \sum_{j=2}^{n} \hat{x}^k_{v_j} + \delta_k, \tag{10}$$

where $\{\delta_i\}^k_{i=1} \in \mathbb{R}$ is the coherent error of expert i. The gating network generates a set of weights $\{w^i_{v_j}\}^k_{i=1} \in [0, 1]$ for experts at v_j. We can then rewrite Eq (10) as follows

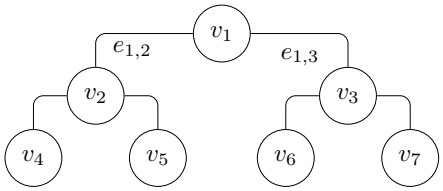
$$\sum_{i=1}^{k} w^i_{v_1} \hat{x}^i_{v_1} = \sum_{j=2}^{n} \sum_{i=1}^{k} w^i_{v_j} \hat{x}^i_{v_j} + \sum_{i=1}^{k} w_i \delta_i. \tag{11}$$

Given $x_{v_j} \in [\hat{x}^1_{v_j}, \hat{x}^k_{v_j}]$, then $\{\delta_i\}^k_{i=1}$ cannot be all strictly positive or negative. Therefore, we can find a set of weights $\{w_i\}^k_{i=1}$ such that $\sum^k_{i=1} w_i \delta_i = 0$, so `MECATS` is coherent. □

Remark The ideal value of w_i in Eq (11) can be written as $w_i = \frac{w^i_{v_1} \hat{x}^i_{v_1} - \sum^n_{j=2} w^i_{v_j} \hat{x}^i_{v_j}}{\hat{x}^i_{v_1} - \sum^n_{j=2} \hat{x}^i_{v_j}}$. By minimizing $\mathcal{L}_{\text{recon}}$, the gating network should generate weights $\{w^i_{v_j}\}^k_{i=1}$ such that $\sum^k_{i=1} w^i_{v_j} \hat{x}^i_{v_j} = x_{v_j}$, since $x_{v_j} \in [\hat{x}^1_{v_j}, \hat{x}^k_{v_j}]$ can be written as the convex combination of each expert's forecast. The generated weights should also satisfy $w^i_{v_1} \hat{x}^i_{v_1} - \sum^n_{j=2} w^i_{v_j} \hat{x}^i_{v_j} = w_i \delta_i$. There is normally a trade-off between accuracy and coherency in empirical evaluations.

B POST-TRAINING AND IN-TRAINING RECONCILIATION

In this section, we briefly discuss our baseline approaches: post-training (Ben Taieb & Koo, 2019; Hyndman et al., 2011; 2016; Wickramasuriya et al., 2015) and in-training (Han et al., 2021) reconciliation methods mentioned in the main paper as additional reference.



$$S = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ & & \mathbf{I}_4 & \end{bmatrix}$$

Figure 6: Left: an example of a three-level structure for aggregated time series data. Right: The associated S matrix that defines the hierarchical structure.

B.1 OLS AND MINT METHOD

Given the example of three-level hierarchical structure in the main paper, define a mapping matrix S, that encapsulates the mutual relationships among each vertex (Figure 6). Denote $b_t \in \mathbb{R}^m$, $a_t \in \mathbb{R}^k$ as the observations at time t for the m and k series at the bottom and aggregation level(s), respectively. Then $S \in \{0, 1\}^{n \times m}$, and each entry S_{ij} equals to 1 if the i^{th} aggregated series contains the j^{th} bottom-level series, where $i = 1, ..., k$ and $j = 1, ..., m$. Denote $\mathcal{I}_T = \{y_1, y_2, ..., y_T\}$ as the time series data observed up to time T; $\hat{b}_T(h)$ and $\hat{y}_T(h)$ as the h-step ahead forecast on the bottom-level and all levels based on $\mathcal{I}_T$. Let $\hat{e}_T(h) = y_{T+h} - \hat{y}_T(h)$ be the h-step ahead conditional

base forecast errors and $\hat{\beta}_T(h) = \mathbb{E}[\hat{b}_T(h) \mid \mathcal{I}_T]$ be the bottom-level mean forecasts. We then have $\mathbb{E}[\hat{y}_T(h) \mid \mathcal{I}_T] = S \cdot \hat{\beta}_T(h)$.

The goal of the post-training (or two-stage) reconciliation approach is to compute some appropriately selected matrix $P \in \mathbb{R}^{m \times n}$ to combine the base forecasts linearly: $\tilde{y}_T(h) = SP\hat{y}_T(h)$. Here, P maps the base forecasts $\hat{y}_T(h)$ into forecasts at the most disaggregated level and are then summed up by S to obtain the reconciled forecasts. We define $\tilde{y}_T(h)$ as the reconciled forecasts which are coherent by construction. Assume $\mathbb{E}[\hat{e}_T(h) \mid \mathcal{I}_T] = 0$, then the reconciled forecasts $\tilde{y}_T(h)$ will be unbiased if and only if $SPS = S$, which leads to the unbiased base forecast assumption:

$$\mathbb{E}[\tilde{y}_T(h) \mid \mathcal{I}_T] = \mathbb{E}[\hat{y}_T(h) \mid \mathcal{I}_T] = S \cdot \hat{\beta}_T(h) \tag{12}$$

The optimal combination approach proposed by (Hyndman et al., 2011), is based on the assumption in Eq (12) and solve the regression problem below using the ordinary least square (OLS) method:

$$\hat{y}_T(h) = S \cdot \hat{\beta}_T(h) + \varepsilon_h, \tag{13}$$

where ε_h is the independent coherency error with zero mean and $\mathrm{Var}(\varepsilon_h) = \Sigma_h$. The OLS estimator of $\hat{\beta}_T(h)$ taken into account the coherency error is given by

$$\tilde{\beta}_T(h) = (S^\top \Sigma_h^\top S)^{-1} S^\top \Sigma_h^\top \hat{y}_T(h), \tag{14}$$

which is an unbiased, minimum variance estimator. The optimal P is $(S^\top \Sigma_h^\top S)^{-1} S^\top \Sigma_h^\top$. The reconciled mean estimation can therefore be obtained accordingly. We further define the reconciliation error as $\tilde{e}_T(h) = y_{T+h} - \tilde{y}_T(h)$, the original problem can also be formulated as

$$\min_{P \in \mathcal{P}} \mathbb{E}[\|\tilde{e}_T(h)\|_2^2 \mid \mathcal{I}_T] \quad \text{subject to } \mathbb{E}[\tilde{y}_T(h) \mid \mathcal{I}_T] = \mathbb{E}[\hat{y}_T(h) \mid \mathcal{I}_T] \tag{15}$$

If the assumption 12 still holds, then minimizing Eq (15) reduces to

$$\min_{P \in \mathcal{P}} \mathrm{Tr}(\mathrm{Var}[\tilde{e}_T(h) \mid \mathcal{I}_T]) \quad \text{subject to (12)}, \tag{16}$$

where Tr(.) denotes the trace of a matrix that corresponds to a trace minimization problem (MinT). In (Wickramasuriya et al., 2019), the proposed optimal solution of P obtained by solving this problem is given by

$$P = (S^\top W_h^{-1} S)^{-1} S^\top W_h^{-1}, \tag{17}$$

where $W_h = \mathbb{E}[\hat{e}_T(h)\hat{e}_T^\top(h) \mid \mathcal{I}_T]$ is the variance-covariance matrix of the h-step-ahead base forecast errors, which is different from the coherence errors Σ_h in OLS reconciliation method given in Eq (14). There are various covariance estimators for W_h considered in (Wickramasuriya et al., 2019), the most effective one is the shrinkage estimator with diagonal target, and can be computed by

$$W_h = (1-\alpha)\hat{W}_s + \alpha \hat{W}_d, \quad \hat{W}_s = \frac{1}{T}\sum_{t=1}^{T} \hat{e}_t(1)\hat{e}_t(1)^\top, \tag{18}$$

where $\hat{W}_d = \mathrm{diag}(\hat{W}_s)$ and $\alpha \in (0, 1]$. A simpler substitution is $W_h = k_h \hat{W}_s, \forall h$, where $k_h > 0$. This is a sample covariance estimator for $h = 1$. The above three methods discussed, are implemented as MinT-ols. MinT-shr, and MinT-sam approaches in the main paper.

B.2 Relaxed Unbiasedness Assumption

The above methods are based on the unbiasedness assumption in Eq (12). However, in many situations it is likely that the forecasting models are mis-specified. (Ben Taieb & Koo, 2019) proposed a method to relax this assumption. In particular, the objective function in (15) can be decomposed as

$$\mathbb{E}[\|y_{T+h} - \tilde{y}_T(h)\|_2^2 \mid \mathcal{I}_T] \tag{19}$$

$$= \|SP \cdot (\mathbb{E}[\hat{y}_T(h) \mid \mathcal{I}_T] - \mathbb{E}[y_{T+h} \mid \mathcal{I}_T]) + (S - SPS) \cdot \mathbb{E}[b_{T+h} \mid \mathcal{I}_T]\|_2^2 \tag{20}$$

$$+ \mathrm{Tr}(\mathrm{Var}[y_{T+h} - \tilde{y}_T(h) \mid \mathcal{I}_T]), \tag{21}$$

where (20) and (21) are the bias and variance terms of the revised forecasts $\tilde{y}_T(h)$. Assumption 12 in MinT method renders (20) to 0. Therefore, directly minimize the objective in (19) provides a more general form of reconciliation represented by following empirical risk minimization (ERM) problem:

$$\min_{P \in \mathcal{P}} \frac{1}{(T - T_1 - h + 1)n} \sum_{t=T_1}^{T-h} \|y_{t+h} - SP\hat{y}_t(h)\|_2^2, \tag{22}$$

	Coherent	Matrix Inversion	Probabilistic	Model Agnostic	Unbiased Assumption	Quantile Crossing
Base	No	No	No	Yes	No	N.A.
BU	Yes	No	No	Yes	Yes	N.A.
MinT	Yes	Yes	No	Yes	Yes	N.A.
ERM	Yes	Yes	No	Yes	No	N.A.
SHARQ	Approx.	No	Yes	No	No	Yes
MECATS	Approx.	No	Yes	Yes	No	No

Table 3: Comparison on different properties of reconciliation methods.

where $T_1 < T$ is the number of observations used for model fitting. According to (Ben Taieb & Koo, 2019), this method demonstrates better empirical performance than MinT, particularly when the forecasting models are mis-specified. However, this approach does not perform very well in our implementation when combining with mixture-of-experts. Hence we do not report its results in the main paper due to space constraints. An implementation of this method can also be found in the submitted Python code.

B.3 In-Training Reconciliation

Post-training reconciliation are effective methods in many situations, but they introduce additional computational complexity in the inference step, due to the inversion of S matrix. This is particularly undesirable when reconciling a significant amount of time series in industry forecasting pipelines. In (Han et al., 2021), the authors proposed a method called SHARQ that enables forecasting model to learn across hierarchical structure during model training. Given the same vertex and edge graph representation in the main paper, the core idea is to modify the objective function of forecasting model to incorporate additional information from adjacent aggregation levels as a regularization term:

$$\mathcal{L}_{\text{recon}} = \mathcal{L}_{v_i}(\hat{x}^{v_i}_{t_i+1:T_i}, x^{v_i}_{t_i+1:T_i}) + \lambda_{\mathsf{K}} \cdot \|\hat{x}^{v_i}_{t_i+1:T_i} - \sum_{e_{i,j}\in E} e_{i,j}\, \hat{x}^{v_j}_{t_j+1:T_j}\|^2. \tag{23}$$

Note that this formulation is similar to the gating network objective discussed in the main paper. However, Eq (23) is designed to be an objective function for a single model. Normally, this type of method requires the model to be trained in a gradient based approach, which is not always the case.

Furthermore, SHARQ provides distribution-free probabilistic forecasts through multi-quantile estimation, and calibrate each quantile on top of the coherent point estimation. The multi-quantile estimation is achieved through replacing $\mathcal{L}_{v_i}$ by multiple quantile loss, and the calibration step is done through minimizing another loss function for arbitrary quantile level τ at vertex i:

$$\mathcal{L}_q := \left[f\left(Q_i^\tau - Q_i^{0.5}\right) - \sum_{e_{i,k}\in E} e_{i,k}\, f\left(Q_k^\tau - Q_k^{0.5}\right) + \text{Var}(\epsilon) \right]^2. \tag{24}$$

The idea of calibration in Eq (24) is to make sure the distance measures of each quantile estimation to the point estimation are also coherent across the given hierarchical structure. This can serve as an extra step after obtaining the multi-quantile estimator and coherent point predictor, but there is no strict guarantee about the monotonicity w.r.t. the quantile level τ. Overall, SHARQ provides a controllable trade-off between coherency and forecasting accuracy of a single time series. It reduces the inference time by removing the post-processing step required by aforementioned methods. However, one cannot combine any model with this approach, and it is also unrealistic to use one forecasting model to represent the entire hierarchy of time series. `MECATS` has successfully addressed this problem by employing the mixture-of-experts framework. It has also improved the uncertainty estimation by generating more coherent probabilistic forecast specified by any quantile.

C Comparison with Related Methods

In Table 3, we compare the available reconciliation approaches on different aspects to illustrate the advantage of `MECATS`. Note that the base method refers to regular multi-variate forecast without reconciliation. `MECATS` has therefore addressed major problems in hierarchical time series

forecasting. In addition, previous works by Rangapuram et al. (2021) and Taieb et al. (2017) also addressed the same problem using different methods. In particular, Taieb et al. (2017) proposed to use copula to aggregate bottom-level distributions to higher levels. However, this method only reconciles point forecasts and obtain higher-level distributions in a bottom-up fashion. This causes potential problems such as error accumulation in highly aggregated levels. To avoid this, one needs to require reasonable probabilistic predictions at the bottom-level beforehand. Also, the possible use of high-dimensional copula in real applications is another drawback. Rangapuram et al. (2021) also addresses the reconciliation problem during model training stage. It first formulates a constrained optimization problem as the objective of reconciliation, and then incorporates this as an add-on layer during model training. However, this method requires distributional assumptions and is designed for deep neural network based forecasting models.

D PROCEDURES FOR TRAINING MULTI-QUANTILE GENERATOR

In this section, we discuss more details and backgrounds in training the multiple quantile forecasters of MECATS. Implementation of this method can also be found in the submitted code repository.

D.1 CHEBYSHEV POLYNOMIAL APPROXIMATION

Chebyshev polynomial is an important subject in numerical methods (Dahlquist & Björck, 2008). It is one type of orthogonal polynomials and widely used in numerical integration. The polynomial is defined as a mapping $T_k : [-1, 1] \to \mathbb{R}$ and its recurrent relationship

$$T_0(z) := 1, \quad T_1(z) := z, \quad T_{k+1}(z) := 2zT_k(z) - T_{k-1}(z), \ \ k \geq 1. \tag{25}$$

This recurrence can also be written as $T_k(\cos\theta) = \cos(k\theta)$. As mentioned in the main paper, zeros of the k^{th} Chebyshev polynomial T_k is located in the range of $[-1, 1]$, which can be computed by

$$t_k = \cos\left(\frac{\pi(k+\frac{1}{2})}{d}\right), \quad k \in [0, d). \tag{26}$$

Using trigonometric properties of the cosine function, we can prove the Chebyshev polynomials are orthogonal with each other. Therefore, the approximation using Chebyshev polynomials gives a linearly independent system. Then for arbitrary continuous function $\phi(z) \to \mathbb{R},\ z = 2\tau_s - 1 \in [-1, 1]$, it can be approximated using Chebyshev polynomials by

$$\phi(z) \approx \frac{1}{2}c_0T_0(z) + c_1T_1(z) + \cdots + c_{d-1}T_{d-1}(z), \tag{27}$$

where $c_i = \sum_{k=0}^{d-1} \phi(t_k)\cos\left(\frac{i\pi(k+\frac{1}{2})}{d}\right),\ i \in [0, d)$. We can use this expression to efficiently compute the value of c_i using the DCT method in $\mathcal{O}(d \log d)$ time. According to (Dahlquist & Björck, 2008), Eq (27) is a good approximation when d is sufficiently large. In our experiments, we use $d = 16$.

D.2 COMPUTING CHEBYSHEV COEFFICIENTS

We now discuss how to compute the coefficients for quantile estimations. As mentioned in the main paper, $\phi(t, x^{v_i}_{1:t_i}; \Theta_u)$ is the neural network that approximates the derivative of quantiles. We evaluate $\phi(t, x^{v_i}_{1:t_i}; \Theta_u)$ at its d uniformly distributed roots for d-dimension truncated Chebyshev polynomial approximation, i.e., $\{\phi(t_k, x^{v_i}_{1:t_i}; \Theta_u)\}_{k=0}^{d-1}$. During implementation, we first use the sliding window approach to process the input data $x^{v_i}_{1:t_i}$, where $\mathrm{sw}(x^{v_i}_{1:t_i})$ can be processed by $\phi(\Theta_u)$ in a common supervised learning way. $\{t_k\}_{k=0}^{d-1}$ serves as an additional feature of $\phi(\Theta_u)$ input. The $\phi(\Theta_u) : \mathbb{R}^{\mathrm{bs}\times(\omega+1)} \to \mathbb{R}^{\mathrm{bs}\times 1}$ is implemented as a multilayer perceptron (MLP) with strictly positive output, where bs and ω represent the batch size and window length, respectively. Since the MLP should be evaluated at multiple roots, we combine $\mathrm{sw}(x^{v_i}_{1:t_i}) \in \mathbb{R}^{\mathrm{bs}\times\omega}$ with different $\{t_k\}_{k=0}^{d-1}$ and feed them into d replicates of MLP respectively. The outputs of d MLP are then transformed by the type-II DCT algorithm to obtain the coefficients $\{c_k(x^{v_i}_{1:t_i})\}_{k=0}^{d-1}$. Full procedure in computing the Chebyshev coefficients can be found in Algorithm 1, which returns the set of coefficients $\{C_k\}_{k=0}^{d-1}$ with size $\mathbb{R}^{\mathrm{bs}\times d}$. These coefficients are then used to compute multiple quantile values by iteratively combining with Chebyshev polynomials defined in Eq (25), where $z = 2\tau_s - 1 \in [-1, 1]$ and τ_s can be any specified quantiles. This step is also discussed in Eq (8) of the main paper.

Algorithm 1 Compute Chebyshev Coefficients for Multi-Quantile Estimations

Input: time series data $\text{sw}(x^{v_i}_{1:t_i})$, Chebyshev polynomial degree d, MLP $\phi(\Theta_u)$, point forecast $\hat{x}^{v_i}_{t_i+1:T_i}$.
Process:
$\{t_k\}_{k=0}^{d-1} = \cos\left(\frac{\pi(k+\frac{1}{2})}{d}\right), \quad k \in [0, d)$
$\mathcal{T}_k \leftarrow \texttt{Repeat}(\{t_k\}_{k=0}^{d-1}, \lceil \frac{\text{bs}}{d} \rceil + 1)$ ▷ repeat t_k vector $\lceil \frac{\text{bs}}{d} \rceil + 1$ times
$\text{sw}(\mathcal{T}_k) \in \mathbb{R}^{bs \times d}$ ▷ generate d vectors of roots by rolling window on $\mathcal{T}_k$ with step size 1
for $j = 0, \ldots, d-1$ **do**
  $X_j \leftarrow \texttt{concat}[\text{sw}(\mathcal{T}_k)[:, j], \ \text{sw}(x^{v_i}_{1:t_i})] \in \mathbb{R}^{\text{bs} \times (\omega+1)}$ ▷ combine different root values with time series
  $O_j \leftarrow \phi_j(X_j, \Theta_u)$
  $P_j \leftarrow \texttt{softplus}(O_j + 1e^{-5}) + 1e^{-3}$ ▷ strictly positive outputs
end for
$\{c_k\}_{k=0}^{d-1} \leftarrow \texttt{DCT}(\{P_j\}_{j=0}^{d-1})$
$\{C_k\}_{k=0}^{d-1} \leftarrow C_k = \frac{c_{k-1} - c_{k+1}}{4k}, \ 0 < k < d-1, \quad C_{d-1} = \frac{c_{d-2}}{4(d-1)}$
$C_0 = 2 \cdot \hat{x}^{v_i}_{t_i+1:T_i} - 2\sum_{k=1,k \text{ even}}^{d-1} (-1)^{k/2} C_k$ ▷ assume point predictions are median value
Return: $\{C_k\}_{k=0}^{d-1}$

D.3 Connection to Point Forecast

Although the above framework can simultaneously generate multiple quantile estimations, it is helpful to set the point forecast $\hat{x}^{v_i}_{t_i+1:T_i}$ to be a constraint (e.g., a summary statistic) of the distribution formulated by quantiles. We discuss two cases when $\hat{x}^{v_i}_{t_i+1:T_i}$ is the median or mean value of the predictive distribution. The constraint is reflected on Chebyshev coefficient C_0 which is computed specifically in the algorithm. If $\hat{x}^{v_i}_{t_i+1:T_i}$ is the median estimation, we have $q(0.5, x^{v_i}_{1:t_i}, \hat{x}^{v_i}_{t_i+1:T_i}) = \hat{x}^{v_i}_{t_i+1:T_i}$, since $T_k(0) = 0$ when k is odd and $T_k(0) = (-1)^{k/2}$ when k is even, and

$$q(0.5, x^{v_i}_{1:t_i}, \hat{x}^{v_i}_{t_i+1:T_i}) = \sum_{k=1}^{d-1} C_k(x^{v_i}_{1:t_i}) \cdot T_k(0) + \frac{1}{2} C_0(x^{v_i}_{1:t_i}, \hat{x}^{v_i}_{t_i+1:T_i}), \tag{28}$$

we can therefore obtain the value of C_0 given the point forecasts are median:

$$C_0(x^{v_i}_{1:t_i}, \hat{x}^{v_i}_{t_i+1:T_i}) = 2 \cdot \hat{x}^{v_i}_{t_i+1:T_i} - 2 \sum_{k=1,k \text{ even}}^{d-1} (-1)^{k/2} C_k(x^{v_i}_{1:t_i}). \tag{29}$$

Similarly, if $\hat{x}^{v_i}_{t_i+1:T_i}$ is determined to be the mean prediction, by definition we have

$$\begin{aligned} \int_{-1}^{1} z \cdot q(z, x^{v_i}_{1:t_i}, \hat{x}^{v_i}_{t_i+1:T_i}) \, dz &= \int_{-1}^{1} z \cdot \left[\sum_{k=1}^{d-1} C_k(x^{v_i}_{1:t_i}) \cdot T_k(z) + \frac{1}{2} C_0(x^{v_i}_{1:t_i}, \hat{x}^{v_i}_{t_i+1:T_i}) \right] dz \\ &= \frac{1}{2} C_0(x^{v_i}_{1:t_i}, \hat{x}^{v_i}_{t_i+1:T_i}) + \sum_{k=1}^{d-1} C_k(x^{v_i}_{1:t_i}) \int_{-1}^{1} z \cdot T_k(z) \, dz. \end{aligned} \tag{30}$$

Since Chebyshev polynomials are symmetric, we have

$$\int_{-1}^{1} z \cdot T_k(z) \, dz = [1 + (-1)^{k+1}] \int_{0}^{1} z \cdot T_k(z) \, dz. \tag{31}$$

Then Eq (31) is zero if k is even. Otherwise, according to the recurrence of Chebyshev polynomial

$$\int_{-1}^{1} z \cdot T_k(z) \, dz = \left[\frac{T_{k-2}(z)}{2(k-2)} - \frac{T_{k+2}(z)}{2(k+2)} \right] \Bigg|_0^1 = \frac{2}{k^2 - 4}. \tag{32}$$

Combine Eq (32) with Eq (30) and impose $\int_{-1}^{1} z \cdot q(z, x^{v_i}_{1:t_i}, \hat{x}^{v_i}_{t_i+1:T_i}) \, dz = \hat{x}^{v_i}_{t_i+1:T_i}$, we then have

$$C_0(x^{v_i}_{1:t_i}, \hat{x}^{v_i}_{t_i+1:T_i}) = 2 \cdot \hat{x}^{v_i}_{t_i+1:T_i} - 4 \sum_{k=1,k \text{ odd}}^{d-1} \frac{C_k(x^{v_i}_{1:t_i})}{k^2 - 4}. \tag{33}$$

Algorithm 2 Training MECATS

Input: Graph structure $G = \{V, E\}$, $\mathcal{I}_1 = \{x_{1:t_i}^{v_i}\}_{i=1}^N$, $\mathcal{I}_2 = \{x_{t_i+1:T_i}^{v_i}\}_{i=1}^N$, test set $\mathcal{I}_3 = \{x_{T_i+1:T_i+h}^{v_i}\}_{i=1}^N$, experts $\{\mathcal{M}_i\}_{i=1}^N$, gating networks $\{\mathcal{G}_i\}_{i=1}^N$, uncertainty wrappers $\{\mathcal{U}_i\}_{i=1}^N$, quantile levels $\{\tau_s\}_{s=1}^n$.
Process:
for each vertex v_i at level $[\mathsf{K}, \mathsf{K}-1, \ldots, 1]$ **do**
  pretrain experts $\mathcal{M}_i$ on $\mathcal{I}_1^{v_i}$
  if v_i at bottom aggregation level **then**
    train $\mathcal{G}_i$ on $\mathcal{I}_1^{v_i} \cup \mathcal{I}_2^{v_i}$ using $\mathcal{L}_{v_i}$ in Eq (2)
  else
    train $\mathcal{G}_i$ on $\mathcal{I}_1^{v_i} \cup \mathcal{I}_2^{v_i}$ using $\mathcal{L}_{\text{recon}}$ in Eq (2)
  end if
  $\hat{x}_{t_i+1:T_i}^{v_i} \leftarrow \{\mathcal{M}_i,\ \mathcal{G}_i, \mathcal{I}_1^{v_i}\}$
  train $\mathcal{U}_i$ on $\mathcal{I}_1^{v_i} \cup \mathcal{I}_2^{v_i} \cup \hat{x}_{t_i+1:T_i}^{v_i}$ using $\mathcal{L}_q$ in Eq (3)
end for
Update experts $\{\mathcal{M}_i\}_{i=1}^N$ on $\mathcal{I}_2$
Output: Coherent quantile forecasts starting at $T_i + 1 \leftarrow \{\{\mathcal{M}_i\}_{i=1}^N, \{\mathcal{G}_i\}_{i=1}^N, \{\mathcal{U}_i\}_{i=1}^N, \mathcal{I}_3, \{\tau_s\}_{s=1}^n\}$.

Results in Eq (29) and Eq (33) have shown that our multiple quantile forecasts can be constrained on point forecasts from the modularized ME framework, making the whole distribution more coherent across the hierarchical structure. During our implementation, we use median constraint on the point prediction. Note that we can also derive the value of C_0 following similar procedure for other summary statistics such as maximum or minimum value.

D.4 Procedure for Training MECATS

Algorithm 2 wraps up the overall procedure of expert pretraining, point forecast, and multiple quantile forecasts. Note that in practice, we can parallelize training on time series data at the same level.

E Background in Online Change-point Detection

We now discuss related works (Adams & MacKay, 2007; Knoblauch & Damoulas, 2018) in online change-point detection and how they are integrated into MECATS.

E.1 Bayesian Online Change-point Detection

Define r_t be the run length at time t, $\mathcal{D}$ be a set of N i.i.d. observations with $\{x_n\}_{n=1}^N$, and $x^{(l)}$ is the set of observations with run length l. BOCPD performs prediction of the next data point by

$$P(x_{t+1} \mid x_{1:t}) = \sum_{r_t} P(x_{t+1}, r_t \mid x_{1:t}) = \sum_{r_t} \overbrace{P(x_{t+1} \mid r_t, x^{(l)})}^{\text{UPM predictive}} \cdot \overbrace{P(r_t \mid x_{1:t})}^{\text{RL posterior}}, \tag{34}$$

where UPM predictive is the underlying probabilistic model (UPM) predictive, it can also be written as $P(x_{t+1} \mid r_t = l, x_{(t-l):t})$. Conjugate-exponential models are convenient for integrating with the change-point detection framework. The general form for any member of exponential family (EF) is

$$P(x \mid \eta) = h(x) g(\eta) \exp\{\eta^\top u(x)\}, \tag{35}$$

where η is the natural parameter, $h(x)$ is the underlying measure, $u(x)$ is the sufficient statistic of data and $g(\eta)$ is a normalizer. The conjugate prior with hyper-parameters $\mathcal{V}, \mathcal{X}$ is

$$P(\eta \mid \mathcal{X}, \mathcal{V}) = f(\mathcal{X}, \mathcal{V})\ g(\eta)^{\mathcal{V}} \exp\{\eta^\top, \mathcal{X}\}, \tag{36}$$

the posterior after observing N data points can be written as

$$P(\eta \mid X, \mathcal{X}, \mathcal{V}) \propto g(\eta)^{N+\mathcal{V}} exp\{\eta^\top (\sum_{n=1}^N u(x_n) + \mathcal{X})\}. \tag{37}$$

Algorithm 3 Loss Mitigation for Mixture of Heterogeneous Experts

Input: ME (experts $\mathcal{M}$, gating network $\mathcal{G}$) initialized on data $\mathcal{D} = x_{1:T_i}$; conjugate exponential model $\mathcal{E}$; model initial parameters $\mathcal{V}_0^{(0)} = \mathcal{V}_{\text{prior}}$, $\mathcal{X}_0^{(0)} = \mathcal{X}_{\text{prior}}$; batch size m; constants β_0, γ; step $\mathcal{N} = \infty$.
Process:
for new sample batch $p = 0, 1, \ldots$ **do**
  $\hat{x}_{T_i+pm:T_i+(p+1)m} \leftarrow \texttt{ME}(\mathcal{M}, g_l(\mathcal{D}, \Theta_g))$
  $R_{pm:(p+1)m} = x_{T_i+pm:T_i+(p+1)m} - \hat{x}_{T_i+pm:T_i+(p+1)m}$
  change_point $\leftarrow \texttt{BOCPD}(\mathcal{E}, R_{pm:(p+1)m})$
  if change_point **then**
    $\mathcal{N} = 0$
  end if
  $g_{\text{shr}}(\mathcal{D}, \Theta_g) = (1 - \beta_0\, e^{-\mathcal{N}/\gamma}) \cdot g_l(\mathcal{D}, \Theta_g) + \beta_0\, e^{-\mathcal{N}/\gamma} \cdot \overline{g}$
  if $e^{-\mathcal{N}/\gamma} < 0.1$ **then**
    $\mathcal{N} = \infty$ ▷ remove the weight buffer if the shrinkage factor is sufficiently low
  else
    $\mathcal{N} = \mathcal{N} + 1$
  end if
  online prediction $\leftarrow \texttt{ME}(\mathcal{M}, g_{\text{shr}}(\mathcal{D}, \Theta_g))$
  $\mathcal{D} \leftarrow \mathcal{D} \cup \{x_{T_i+pm:T_i+(p+1)m}\}$
  Update parameter Θ_g of gating network $\mathcal{G}$
  Update the experts $\mathcal{M}$ ▷ this step is optional if updating $\mathcal{M}$ is expensive
end for
Return: online prediction

Therefore, the parameter update for obtaining new data points are:

$$\mathcal{V}' = \mathcal{V}_{\text{prior}} + N, \quad \mathcal{X}' = \mathcal{X}_{\text{prior}} + \sum_{n=1}^{N} u(x_n). \tag{38}$$

We can skip the integration of EF posterior for computing the UPM predictive. Instead, we use the parameter update rules to make prediction at time t by $t-1$. Denote $p(x_t \mid r_{t-1} = l, x^{(l)})$ as $P(x_t \mid \mathcal{V}_{t-1}^{(l)}, \mathcal{X}_{t-1}^{(l)})$. As new datum sequentially arrives, parameters at previous step are prior for the following step, i.e.,

$$\begin{aligned} \mathcal{V}_t^{(0)} &= \mathcal{V}_{\text{prior}}, \quad \mathcal{X}_t^{(0)} = \mathcal{X}_{\text{prior}}, \\ \mathcal{V}_t^{(l)} &= \mathcal{V}_{t-1}^{(l-1)} + 1, \quad \mathcal{X}_t^{(l)} = \mathcal{X}_{t-1}^{(l-1)} + u(x_t). \end{aligned} \tag{39}$$

Eq (39) is an important step in online change-point detection. Our use case is a special case of the conjugate-exponential models, which estimates the unknown mean μ of a Gaussian distribution with fixed variance σ^2 (Murphy, 2007). The conjugate prior has the form

$$P(\mu) \propto \exp\left(-\frac{1}{2\sigma_0^2}(\mu - \mu_0)^2\right) \propto \mathcal{N}(\mu \mid \mu_0, \sigma_0^2). \tag{40}$$

Define $\mathcal{D} = (x_1, \ldots, x_n)$ be the observed data, its likelihood has a similar form

$$P(\mathcal{D} \mid \mu) \propto \exp\left(-\frac{n}{2\sigma^2}(\overline{x} - \mu)^2\right) \propto \mathcal{N}(\overline{x} \mid \mu, \frac{\sigma^2}{n}). \tag{41}$$

Then the posterior can be computed by

$$P(\mu \mid \mathcal{D}) \propto P(\mathcal{D} \mid \mu, \sigma) \cdot P(\mu \mid \mu_0, \sigma_0^2) \tag{42}$$

$$\propto \exp\left[-\frac{1}{2\sigma^2}\sum_i (x_i - \mu)^2\right] \cdot \exp\left[-\frac{1}{2\sigma_0^2}(\mu - \mu_0)^2\right] \tag{43}$$

$$= \exp\left[-\frac{1}{2\sigma^2}\sum_i (x_i^2 + \mu^2 - 2x_i\mu) - \frac{1}{2\sigma_0^2}(\mu^2 + \mu_0^2 - 2\mu_0\mu)\right]. \tag{44}$$

Algorithm 4 Bayesian Online Change-point Detection

Function BOCPD ($\mathcal{E}, R_t$) **:**

Compute UPM predictive probability: $\pi_{t-1}^{(l)} = P(R_t \mid \mathcal{V}_{t-1}^{(l)}, \mathcal{X}_{t-1}^{(l)})$

Compute growth probability: $P(r_t = r_{t-1} + 1, R_{1:t}) = P(r_{t-1}, R_{1:t-1}) \cdot \pi_{t-1}^{(l)} \cdot (1 - H(r_{t-1}))$

Compute change-point probability: $P(r_t = 0, R_{1:t}) = \sum_{r_{t-1}} P(r_{t-1}, R_{1:t-1}) \cdot \pi_{t-1}^{(l)} \cdot H(r_{t-1})$

Calculate evidence: $P(R_{1:t}) = \sum_{r_t} P(r_t, R_{1:t})$

Calculate run length posterior: $P(r_t \mid R_{1:t}) = P(r_t, R_{1:t})/P(R_{1:t})$

if $\underset{r_t}{\text{argmax}}\, P(r_t \mid R_{1:t}) == 1$ and $t \neq 1$ **then**

  change_point $\leftarrow$ True

else

  change_point $\leftarrow$ False

end

Update model parameters using Eq (39)

return change_point

End Function

Since the product of two Gaussian distributions is still a Gaussian distribution, we have

$$P(\mu \mid \mathcal{D}) \propto \exp\left[-\frac{\mu^2}{2}\left(\frac{1}{\sigma_0^2} + \frac{n}{\sigma^2}\right) + \mu\left(\frac{\mu_0}{\sigma_0^2} + \frac{\sum_i x_i}{\sigma^2}\right) - \left(\frac{\mu_0^2}{2\sigma_0^2} + \frac{\sum_i x_i^2}{2\sigma^2}\right)\right] \tag{45}$$

$$\rightarrow \exp\left[-\frac{1}{2\sigma_n^2}(\mu^2 - 2\mu\mu_n + \mu_n^2)\right] \tag{46}$$

$$= \exp\left[-\frac{1}{2\sigma_n^2}(\mu - \mu_n)^2\right] \tag{47}$$

By matching the coefficients of μ^2 between Eq (45) and Eq (47), we can obtain the updating equation for σ_n^2, where σ^2 is the known observation noise variance and σ_0^2 is the prior variance:

$$-\frac{\mu^2}{2\sigma_n^2} = -\frac{\mu^2}{2}\left(\frac{1}{\sigma_0^2} + \frac{n}{\sigma^2}\right) \;\rightarrow\; \frac{1}{\sigma_n^2} = \frac{1}{\sigma_0^2} + \frac{n}{\sigma^2}. \tag{48}$$

Then, by matching coefficients of μ we get

$$\frac{\mu\mu_n}{\sigma_n^2} = \mu\left(\frac{\sum_{i=1}^n x_i}{\sigma^2} + \frac{\mu_0}{\sigma_0^2}\right) \;\rightarrow\; \mu_n = \sigma_n^2\left(\frac{\mu_0}{\sigma_0^2} + \frac{n\bar{x}}{\sigma^2}\right). \tag{49}$$

Eq (48) and Eq (49) are therefore the updating equations discussed in the main paper. Furthermore, the run length posterior is proportional to the joint distribution $P(r_t \mid x_{1:t}) = \frac{P(r_t, x_{1:t})}{\sum_{r_t'} P(r_{t'}, x_{1:t})}$. This joint distribution can be computed recursively by

$$P(r_t, x_{1:t}) = \sum_{r_{t-1}} \overbrace{P(x_t \mid r_t, x^{(l)})}^{\text{UPM predictive}} \overbrace{P(r_t \mid r_{t-1})}^{\text{Changepoint prior}} \overbrace{P(r_{t-1}, x_{1:t-1})}^{\text{Recursive term}}, \tag{50}$$

where the change-point prior $P(r_t \mid r_{t-1})$ has nonzero mass at only two outcomes: $r_t = r_{t-1} + 1$ or $r_t = 0$, and can be written as

$$P(r_t \mid r_{t-1}) = \begin{cases} H(r_{t-1}+1) & \text{if } r_t = 0 \\ 1 - H(r_{t-1}+1) & \text{if } r_t = r_{t-1} + 1 \\ 0 & \text{otherwise} \end{cases} \tag{51}$$

The function H is the hazard function that is commonly used in survival analysis. Based on the above derivation, we show how to perform loss mitigation for our mixture of heterogeneous experts method on temporal sequences with change-points. Detailed procedures are shown in Algorithm 3 and 4.

E.2 Change-point Detection with Model Selection

Standard BOCPD method uses single conjugate-exponential model for change-point detection and prediction. It has also been extended to online model selection, where a model distribution is added

to the original change-point detection procedure. The new recursion formula has changed to

$$P(r_t, x_{1:t}, m_t) = \sum_{m_{t-1}, r_{t-1}} P(x_t \mid r_t, x^{(l)}) \times \overbrace{P(m_t \mid r_{t-1}, x_{1:t-1}, m_{t-1})}^{\text{Model dist. term}} \times P(r_t \mid r_{t-1}) \times P(r_{t-1}, x_{1:t-1}, m_{t-1}), \tag{52}$$

and the change-point detection procedure in Algorithm 4 is also changed accordingly by incorporating the model distribution. The resulting algorithm, called BOCPD-MS, performs prediction, model selection and change-point detection online. However, BOCPD-MS restricted its models universe to be Bayesian and also homogenous (e.g., Gaussian process or auto-regressive). Our method, which extends BOCPD into multiple heterogeneous models and adaptively find the best model combination online, has a similar flavor, but demonstrates better empirical performance than BOCPD-MS.

E.3 Integrating MECATS with Online Changepoint Detection

In many real applications, samples are collected and become available sequentially, which requires the model to be updated in an online manner. It is possible that the change of dynamics will occur in such settings. Although the ME framework has proven to be reasonably adaptive when abrupt changes occur (Chaer et al., 1997; Weigend et al., 1995), it still requires a certain amount of time to learn the new dynamics. In the main paper, we propose to alleviate the jump of loss during the transient period in adapting to new dynamics by combining a change-point detection algorithm with ME. Time series change-point detection methods span a range of categories between online and offline (Aminikhanghahi & Cook, 2017; Truong et al., 2020; van den Burg & Williams, 2020). We borrow strength from BOCPD (Adams & MacKay, 2007) that predicts change-points using a Bayesian method.

Consider an ME initialized on a univariate sequential dataset $x_{1:T_i}$, with new data from the same source becoming sequentially available in mini-batches with batch size m. Both gating network and experts are continuously updated by such new samples. Since BOCPD assumes its data is of `i.i.d` nature, we focus on the residuals instead of the original sequence: $R_{pm:(p+1)m} = x_{T_i+pm:T_i+(p+1)m} - \hat{x}_{T_i+pm:T_i+(p+1)m}$, where $p \geq 0$ is the number of mini-batches. We assume the underlying model of the residual R is Gaussian with unknown mean, i.e., $\mu \sim \mathcal{N}(\mu_0, \sigma_0^2),\ R \sim \mathcal{N}(\mu, \sigma^2)$, where σ^2 is the known observational noise variance, and σ_0^2 is the prior variance. This model belongs to the category of conjugate-exponential models, and is convenient for integrating with the change-point detection scheme. Our goal is to estimate the unknown mean μ of R, given the conjugate prior $p(\mu) \propto \mathcal{N}(\mu \mid \mu_0, \sigma_0^2)$ and posterior $p(\mu \mid x_{T_i:T_i+m}) \propto \mathcal{N}(\mu \mid \mu_m, \sigma_m^2)$, from the results in Eq (48) and (49), we have the online updating equation of parameters as: $\frac{1}{\sigma_m^2} = \frac{1}{\sigma_0^2} + \frac{m}{\sigma^2},\ \mu_m = \sigma_m^2 \left(\frac{\mu_0}{\sigma_0^2} + \frac{m\overline{x}}{\sigma^2} \right)$, where $\overline{x}$ defines the mean of m samples. This equation serves as an important step in online change-point detection. When change-point occurs at time stamp t_0. We replace original weights with g_{shr}, which is a combination of weights using a shrinkage factor:

$$g_{\mathsf{shr}}(x_{1:T_i+pm}, \Theta_g) = (1 - \beta_0\, e^{-(t-t_0)/\gamma}) \cdot g_l(x_{1:T_i+pm}, \Theta_g) + \beta_0\, e^{-(t-t_0)/\gamma} \cdot \overline{g}, \tag{53}$$

where $\overline{g} = 1/\mathsf{L}$ is the equally averaged weight; β_0 and γ are constants. By Eq (53), we obtain new weights that are brought closer to the average weight when change-point occurs. The importance of $\overline{g}$ decays exponentially as the experts' weights $\{g_l(x_{1:T_i+pm}, \Theta_g)\}_{l=1}^{\mathsf{L}}$ adapt to new dynamics. We employ this strategy in our experiments at section 4.4. By integrating `MECATS` with BOCPD, it is equivalent to add online model selection to the original change-point detection algorithm.

F Models, Datasets and Evaluation Metrics

F.1 Forecasting Models as Single Experts

Considering diverse situations, the base models we used span a variety of categories: from classical forecasting method to recently prominent deep learning models. All the employed experts have preexisting packages that are public online. We summarize this information below:

- DLM: https://pydlm.github.io/; license: BSD-3-Clause License.

- Auto-ARIMA: https://alkaline-ml.com/pmdarima/0.9.0/index.html; license: Zope Public License.
- Facebook Prophet: https://facebook.github.io/prophet/; license: MIT License.
- LSTNet: https://github.com/laiguokun/LSTNet; license: MIT License.
- DeepAR: https://github.com/zhykoties/TimeSeries; license: Apache-2.0 License.

F.2 DATASETS INFORMATION

We provide detailed information of the real and simulated datasets we used in our experiments. All the data is obtained under the consent from its owner. To the best of our knowledge, it does not contain personally identifiable information or offensive content.

Australian Labour This dataset[1] contains monthly employment information ranging from February 1978 to August 2019 with 500 records for each series. This dataset contains categorical features that can be aggregated across to formulate corresponding hierarchical structures, while we choose three features to obtain a 4-level symmetric structure. Specifically, the 32 bottom level series are hierarchically aggregated using labor force location, gender, and employment status.

M5 Competition This dataset[2] involves the unit sales of various products ranging from January 2011 to June 2016 in Walmart. It involves the unit sales of 3,049 products, classified into 3 product categories (Hobbies, Foods, and Household) and 7 product departments, where these categories are disaggregated. The products are sold across ten stores in three states (CA, TX, and WI). We formulate a 4-level hierarchy in this work, starting from the bottom-level individual item to unit sales of all products aggregated for each store.

Wikipedia Page View This dataset[3] contains daily views of 145k various Wikipedia articles ranging from July 2015 to December 2016. We sample 150 bottom-level series from 145k series and aggregate to obtain upper-level series. The aggregation features include the type of agent, type of access, and country codes. We then obtain a 5-level hierarchical structure with 150 bottom series.

AE-Demand This dataset[4] collected weekly records on the demand of Accident & Emergency (A&E) departments in the UK spanning 7 November 2010 to 7 June 2015. The demands are classified into three types: major A&E, single specialty and other/minor A&E. Forecasting at different time granularity is necessary for better planning and scheduling of resources. We formulate a four-level temporal hierarchical structure, with weekly data at the bottom level, and aggregated to obtain bi-week, month and quarter level. We use the same way to represent temporal hierarchy as in (Athanasopoulos et al., 2017).

Change-point Detection Data The real-data used to evaluate our change-point detection application is general temporal sequence without hierarchical structure. Specifically, the WTI oil price data is obtained from U.S. Energy and Information Administration[5] (EIA). The rest of the data (Snowfall, Nile, and Bee Dance) was experimented by (Knoblauch & Damoulas, 2018) and is available on the Github repository[6].

Simulation Experiment Below is the function used to generate simulated sequential data in section 4.3 of the main paper, assume $x \in [0, 1]$ and f is a deterministic function:

$$f(x) = \begin{cases} 3 + 3x + 0.1 \cdot \sin(60x) & \text{if } x < 0.8 \\ 3 + 3x + 0.1x^2 \cdot \sin(60x) & \text{if } x \geq 0.8 \text{ and } x < 0.9 \\ 10 + 5x^4 + \sin(60x) & \text{otherwise.} \end{cases} \tag{54}$$

[1] https://www.abs.gov.au/AUSSTATS/abs@.nsf/DetailsPage/6202.0Dec%202019?OpenDocument
[2] https://mofc.unic.ac.cy/wp-content/uploads/2020/03/M5-Competitors-Guide-Final-10-March-2020.docx
[3] https://www.kaggle.com/c/web-traffic-time-series-forecasting
[4] https://cran.r-project.org/web/packages/thief/index.html
[5] https://www.eia.gov/
[6] https://github.com/alan-turing-institute/bocpdms/tree/master/Data

We randomly generated 2000 samples using this function and applied Gaussian noise sample-wise with zero mean and 0.05 variance. The first 80% of data is used to initialize the ME framework offline. The rest 20% is used in online updating, and the change-point occurs at 90%.

F.3 Evaluation Metrics

Mean Absolute Scaled Error (MASE) Denote $\hat{Y}_T(h)$ and $Y_T(h)$ as the h-step ahead forecast at time T and its ground truth, respectively. The MASE is commonly used to evaluate the point forecasting performance. It is defined by

$$\text{MASE} = \frac{\frac{100}{H} \times \sum_{h=1}^{H} |Y_T(h) - \hat{Y}_T(h)|}{\frac{1}{T-1}\sum_{t=2}^{T} |Y_T(t) - Y_T(t-1)|}. \tag{55}$$

Similarly, normalized root mean squared error (NRMSE) is another commonly used scaled error to evaluate time series forecasting.

Continuous Ranked Probability Score (CRPS) CRPS measures the compatibility of a cumulative distribution function F with an observation x as:

$$\text{CRPS}(F, x) = \int_{\mathbb{R}} (F(z) - \mathbb{I}\{x \leq z\})^2 \, dz \tag{56}$$

where $\mathbb{I}\{x \leq z\}$ is the indicator function. CRPS attains its minimum when the distribution F and the distribution formulated by x (can be either scalar or vector) are equal. We used `properscoring` to compute CRPS. For simplicity, we compute the CRPS results of the one-step ahead forecasts.

Coherent Loss Given m vertices in the hierarchy at time T, we compute the coherent loss by

$$\frac{1}{H}\sum_{h=1}^{H}\sum_{i=1}^{m} \|\hat{Y}_T^i(h) - \sum_{e_{i,k}\in E} \hat{Y}_T^k(h)\|_1. \tag{57}$$

G Hyper-parameters and Experiment Settings

We list all the hyper-parameters and experiment settings in our empirical evaluations. We have also submitted a Python implementation of all the experiments along with this Appendix. Details of how to run these experiments can be find in the README file under the submitted code. Note that as for single models' parameters, we leave most of the original parameters unchanged (parameters that the authors claimed in their papers or used in code), while we only modify model training related parameters (e.g., learning rate, batch size, epoch) to generate better predictions on a specific dataset. We run all experiments on a NVIDIA GeForce RTX 3090 4-GPU machine. The hyper-parameters are determined by grid searching on the range specified by our applications.

Gating Networks NN_g
LSTM recurrent layer: $\mathbb{R}^{\text{bs}\times\omega\times 1} \to \mathbb{R}^{\text{bs}\times 1}$
fc layer with 60-dim and `Tanh`: $\mathbb{R}^{\text{bs}\times 1} \to \mathbb{R}^{60\times 1}$
fc layer with 5-dim and `Softmax`: $\mathbb{R}^{60\times 1} \to \mathbb{R}^{5\times 1}$
recurrent hidden dimension: 1
recurrent layer dimension: 3
batch size: 16
fc hidden dimension: 60
learning rate: $1e$-4
number of epochs: 1200
reconciliation parameter λ_{K}: 0.1

MLP $\phi(\Theta_u)$
fc layer with 120 dims and `ReLU`: $\mathbb{R}^{\text{bs}\times(\omega+1)} \to \mathbb{R}^{\text{bs}\times 120}$
fc layer with 120 dims and `ReLU`: $\mathbb{R}^{\text{bs}\times 120} \to \mathbb{R}^{\text{bs}\times 120}$
fc layer with 60 dims and `ReLU`: $\mathbb{R}^{\text{bs}\times 120} \to \mathbb{R}^{\text{bs}\times 60}$
fc layer with 60 dims and `ReLU`: $\mathbb{R}^{\text{bs}\times 60} \to \mathbb{R}^{\text{bs}\times 60}$
fc layer with 10 dims and `ReLU`: $\mathbb{R}^{\text{bs}\times 60} \to \mathbb{R}^{\text{bs}\times 10}$
fc layer with 1 dims and `Softplus`: $\mathbb{R}^{\text{bs}\times 10} \to \mathbb{R}^{\text{bs}\times 1}$
d: 16
learning rate: $1e$-4
number of epochs: 600
quantiles τ_s: [0.95, 0.7, 0.3, 0.05]
point forecast: median

Single Models' Parameters	
DeepAR	batch size: 16 learning rate: $1e$-4 number of epochs: 50 LSTM layers: 3 number of class: 1 co-variance dimension: 1 LSTM hidden dimension: 40 embedding dimension: 20 sample times: 200 LSTM dropout: 0.1 predict batch: 256 stride: 1
LSTNet	batch size: 16 learning rate: $1e$-3 number of epochs: 1000 optimizer: adam
RNN-GRU	batch size: 16 learning rate: $5e$-4 number of epochs: 1000 hidden dimension: 5 recurrent layer dimension: 3
DLM	trend degree: 2 discount factor: 0.95 prior co-variance: $1e7$

General Hyper-parameters
forecast recursive step: 1
train/validation/test: 0.6/0.2/0.2
online update batch size: 1
online update epoch: 5
hazard function initial value: $1e$-3
prior mean μ_0: 0
prior variance σ_0^2: 2
observation variance σ^2: 1
weight decay factor τ: 2

H Additional Results

H.1 Hierarchical Time Series Forecast

Table 4 - 20 shows hierarchical time series forecasting results on Australian Labour, M5, AE-Demand and Wikipedia datasets by each aggregation level. Similar to the main paper, the results are in the form of $\mathrm{MASE} \pm \mathrm{error}(\mathrm{CRPS})$. We have included simple baseline methods such as bottom-up (BU) aggregation, as well as ensemble averaging and model selection used in Bhatnagar et al. (2021). In addition, we intended to compare with Rangapuram et al. (2021); Taieb et al. (2017), but their state-of-the-art implementations are not available. We will add comparison to these methods once the implementations are made public. The following results are averaged over 5 random experiments.

Recon \ Model	Level 0			
	AR	RNN-GRU	LSTNet	DeepAR
BU	104.31 ±2.89 (**.463**)	86.89 ±0.44 (**.379**)	84.51 ±0.02 (**.384**)	80.12 ±0.31 (**.321**)
Base	66.74 ±0.32 (**.157**)	54.21 ±0.28 (**.161**)	54.63 ±0.07 (**.173**)	52.33 ±0.42 (**.054**)
MinT-sam	85.42 ±0.74 (**.175**)	63.41 ±0.04 (**.091**)	61.89 ±0.36 (**.101**)	57.62 ±0.69 (**.100**)
MinT-shr	81.42 ±0.19 (**.175**)	59.47 ±0.48 (**.094**)	58.61 ±0.28 (**.098**)	51.37 ±0.87 (**.091**)
MinT-ols	86.02 ±1.37 (**.155**)	64.11 ±1.21 (**.112**)	60.72 ±0.97 (**.101**)	56.39 ±0.44 (**.121**)
SHARQ	61.44 ±0.62 (**.190**)	52.07 ±0.45 (**.085**)	52.75 ±0.95 (**.071**)	51.84 ±0.22 (**.045**)
Average	49.34 ±0.65 (**.075**)			
MS	45.12 ±0.23 (**.085**)			
MECATS	**38.84 ±0.04** (**.035**)			

Table 4: Australian Labour dataset, level 0

Recon \ Model	Level 1			
	AR	RNN-GRU	LSTNet	DeepAR
BU	91.46 ±1.63 (**.357**)	85.36 ±0.87 (**.341**)	85.12 ±0.49 (**.307**)	79.97 ±0.13 (**.297**)
Base	82.61 ±0.15 (**.227**)	69.78 ±0.85 (**.142**)	67.48 ±0.92 (**.108**)	59.68 ±0.68 (**.104**)
MinT-sam	84.72 ±1.89 (**.186**)	71.35 ±0.17 (**.134**)	70.63 ±0.02 (**.121**)	65.23 ±0.52 (**.127**)
MinT-shr	78.52 ±0.83 (**.186**)	65.23 ±1.08 (**.137**)	67.45 ±0.21 (**.123**)	60.43 ±0.14 (**.132**)
MinT-ols	82.69 ±0.24 (**.192**)	71.22 ±0.06 (**.123**)	70.46 ±0.59 (**.124**)	64.72 ±0.37 (**.134**)
SHARQ	74.91 ±0.71 (**.176**)	58.69 ±0.41 (**.120**)	57.86 ±0.03 (**.104**)	57.63 ±0.68 (**.112**)
Average	60.87 ±0.33 (**.119**)			
MS	55.61 ±0.74 (**.107**)			
MECATS	**48.64 ±0.78** (**.082**)			

Table 5: Australian Labour dataset, level 1

Recon \ Model	Level 2			
	AR	RNN-GRU	LSTNet	DeepAR
BU	87.78 ±2.41 (**.336**)	79.62 ±0.96 (**.312**)	81.23 ±0.66 (**.324**)	77.14 ±0.06 (**.275**)
Base	75.15 ±0.21 (**.231**)	70.42 ±0.51 (**.158**)	69.84 ±0.96 (**.113**)	67.16 ±0.25 (**.185**)
MinT-sam	92.57 ±0.49 (**.205**)	75.52 ±0.76 (**.148**)	74.81 ±0.45 (**.142**)	69.99 ±0.12 (**.152**)
MinT-shr	83.47 ±1.14 (**.201**)	68.74 ±0.16 (**.144**)	68.12 ±0.47 (**.135**)	67.44 ±0.16 (**.149**)
MinT-ols	89.61 ±1.24 (**.212**)	74.89 ±0.18 (**.142**)	75.05 ±0.93 (**.137**)	72.44 ±1.76 (**.158**)
SHARQ	79.56 ±0.04 (**.197**)	64.02 ±0.09 (**.132**)	62.89 ±0.88 (**.124**)	60.03 ±0.26 (**.134**)
Average	69.29 ±0.42 (**.138**)			
MS	60.03 ±0.26 (**.134**)			
MECATS	**49.17 ±0.36** (**.097**)			

Table 6: Australian Labour dataset, level 2

Recon \ Model	Level 3			
	AR	RNN-GRU	LSTNet	DeepAR
BU	86.44 ±0.63 (**.213**)	72.13 ±0.34 (**.167**)	71.38 ±0.15 (**.154**)	72.05 ±0.18 (**.193**)
Base	86.44 ±0.63 (**.213**)	72.13 ±0.34 (**.167**)	71.38 ±0.15 (**.154**)	72.05 ±0.18 (**.193**)
MinT-sam	112.36 ±2.18 (**.234**)	86.97 ±0.28 (**.159**)	84.21 ±1.12 (**.156**)	81.56 ±1.43 (**.185**)
MinT-shr	97.86 ±0.23 (**.226**)	79.68 ±1.34 (**.153**)	77.59 ±1.18 (**.144**)	78.19 ±0.55 (**.168**)
MinT-ols	108.56 ±2.42 (**.249**)	87.42 ±1.78 (**.167**)	83.96 ±0.76 (**.158**)	82.95 ±0.69 (**.175**)
SHARQ	86.44 ±0.63 (**.213**)	72.13 ±0.34 (**.167**)	71.38 ±0.15 (**.154**)	72.05 ±0.18 (**.193**)
Average	75.56 ±0.94 (**.156**)			
MS	71.38 ±0.15 (**.154**)			
MECATS	**61.22 ±0.14** (**.110**)			

Table 7: Australian Labour dataset, level 3

Recon \ Model	Level 0			
	AR	RNN-GRU	LSTNet	DeepAR
BU	97.62 ±1.21 (**.372**)	84.37 ±0.12 (**.293**)	82.13 ±0.28 (**.289**)	79.98 ±0.11 (**.301**)
Base	64.24 ±0.42 (**.277**)	59.87 ±0.46 (**.198**)	57.63 ±0.31 (**.178**)	54.29 ±0.34 (**.071**)
MinT-sam	88.56 ±0.54 (**.126**)	73.62 ±0.34 (**.150**)	74.12 ±0.29 (**.113**)	68.22 ±0.51 (**.097**)
MinT-shr	78.56 ±0.76 (**.169**)	68.21 ±1.13 (**.126**)	66.78 ±0.06 (**.096**)	65.04 ±0.23 (**.087**)
MinT-ols	83.34 ±0.03 (**.194**)	72.16 ±0.24 (**.144**)	72.46 ±0.17 (**.122**)	69.64 ±0.74 (**.096**)
SHARQ	62.74 ±0.23 (**.262**)	56.31 ±0.17 (**.054**)	54.49 ±0.28 (**.058**)	51.69 ±0.05 (**.039**)
Average	59.61 ±0.38 (**.104**)			
MS	51.69 ±0.05 (**.070**)			
MECATS	**42.61 ±0.14** (**.046**)			

Table 8: M5 dataset, level 0

Recon \ Model	Level 1			
	AR	RNN-GRU	LSTNet	DeepAR
BU	85.62 ±0.34 (**.345**)	82.16 ±0.13 (**.256**)	80.87 ±0.23 (**.249**)	78.12 ±0.24 (**.261**)
Base	69.78 ±0.86 (**.244**)	63.22 ±0.32 (**.167**)	62.19 ±0.17 (**.138**)	59.36 ±0.28 (**.127**)
MinT-sam	84.31 ±0.72 (**.249**)	76.21 ±0.13 (**.155**)	75.33 ±0.05 (**.153**)	71.39 ±0.75 (**.122**)
MinT-shr	81.39 ±0.68 (**.245**)	71.62 ±0.35 (**.154**)	70.34 ±0.19 (**.134**)	69.12 ±0.34 (**.132**)
MinT-ols	84.69 ±0.48 (**.201**)	75.11 ±0.85 (**.157**)	76.21 ±0.04 (**.148**)	72.16 ±0.43 (**.124**)
SHARQ	68.92 ±0.47 (**.212**)	62.16 ±0.27 (**.079**)	61.64 ±0.87 (**.089**)	58.74 ±0.12 (**.103**)
Average	60.48 ±0.58 (**.133**)			
MS	54.72 ±0.63 (**.116**)			
MECATS	**49.75 ±0.22** (**.084**)			

Table 9: M5 dataset, level 1

Recon \ Model	Level 2			
	AR	RNN-GRU	LSTNet	DeepAR
BU	81.69 ±0.13 (**.304**)	76.42 ±0.08 (**.221**)	76.32 ±0.26 (**.233**)	75.39 ±0.42 (**.219**)
Base	74.26 ±0.03 (**.259**)	69.46 ±0.74 (**.178**)	72.76 ±0.93 (**.164**)	67.18 ±0.22 (**.142**)
MinT-sam	82.29 ±0.49 (**.308**)	77.69 ±0.37 (**.173**)	77.43 ±0.21 (**.168**)	75.12 ±0.04 (**.154**)
MinT-shr	82.77 ±0.16 (**.297**)	75.44 ±0.26 (**.167**)	73.42 ±0.43 (**.148**)	72.16 ±0.82 (**.147**)
MinT-ols	86.21 ±0.72 (**.312**)	78.13 ±0.56 (**.172**)	76.93 ±0.17 (**.164**)	74.98 ±0.15 (**.147**)
SHARQ	73.53 ±0.59 (**.246**)	65.37 ±0.63 (**.134**)	72.03 ±0.41 (**.113**)	66.17 ±0.26 (**.142**)
Average	68.29 ±0.25 (**.143**)			
MS	65.02 ±0.24 (**.142**)			
MECATS	**53.61 ±0.42** (**.101**)			

Table 10: M5 dataset, level 2

Recon \ Model	Level 3			
	AR	RNN-GRU	LSTNet	DeepAR
BU	75.46 ±0.57 (**.272**)	72.86 ±0.27 (**.189**)	73.56 ±0.41 (**.156**)	72.04 ±0.36 (**.164**)
Base	75.46 ±0.57 (**.272**)	72.86 ±0.27 (**.189**)	73.56 ±0.41 (**.156**)	72.04 ±0.36 (**.164**)
MinT-sam	87.59 ±0.61 (**.385**)	83.76 ±0.24 (**.194**)	82.06 ±0.46 (**.182**)	82.67 ±0.35 (**.179**)
MinT-shr	85.49 ±0.42 (**.349**)	79.41 ±0.01 (**.189**)	80.46 ±0.93 (**.162**)	78.54 ±0.29 (**.158**)
MinT-ols	88.14 ±0.64 (**.369**)	83.62 ±0.15 (**.199**)	84.18 ±0.57 (**.178**)	81.59 ±0.38 (**.161**)
SHARQ	75.46 ±0.57 (**.272**)	72.86 ±0.27 (**.189**)	73.56 ±0.41 (**.156**)	72.04 ±0.36 (**.164**)
Average	70.29 ±0.34 (**.168**)			
MS	72.04 ±0.36 (**.164**)			
MECATS	**57.89 ±0.47** (**.109**)			

Table 11: M5 dataset, level 3

Recon \ Model	Level 0			
	AR	RNN-GRU	LSTNet	DeepAR
BU	103.41 ±2.03 (**.396**)	96.32 ±0.89 (**.337**)	92.33 ±0.41 (**.312**)	93.43 ±0.56 (**.281**)
Base	79.65 ±0.83 (**.189**)	71.24 ±0.75 (**.163**)	67.32 ±0.48 (**.273**)	74.35 ±0.76 (**.232**)
MinT-sam	91.31 ±0.97 (**.139**)	72.23 ±0.67 (**.169**)	69.13 ±0.36 (**.282**)	76.88 ±0.56 (**.243**)
MinT-shr	88.41 ±0.79 (**.136**)	71.31 ±0.14 (**.167**)	65.83 ±0.27 (**.269**)	73.28 ±0.19 (**.236**)
MinT-ols	90.37 ±0.45 (**.142**)	74.28 ±0.26 (**.174**)	68.93 ±0.25 (**.278**)	79.21 ±0.82 (**.256**)
SHARQ	77.23 ±0.35 (**.123**)	68.19 ±0.29 (**.113**)	64.45 ±0.48 (**.213**)	71.21 ±0.45 (**.211**)
Average	67.32 ±0.29 (**.164**)			
MS	64.45 ±0.48 (**.213**)			
MECATS	**59.89 ±0.32** (**.145**)			

Table 12: AE-Demand dataset, level 0

Recon \ Model	Level 1			
	AR	RNN-GRU	LSTNet	DeepAR
BU	97.12 ±0.31 (**.324**)	88.62 ±0.36 (**.274**)	86.81 ±0.33 (**.189**)	90.13 ±0.07 (**.222**)
Base	83.74 ±0.82 (**.264**)	69.24 ±0.41 (**.207**)	66.28 ±0.12 (**.155**)	79.12 ±0.46 (**.172**)
MinT-sam	86.79 ±0.42 (**.252**)	73.24 ±0.63 (**.215**)	68.38 ±0.15 (**.167**)	80.31 ±0.73 (**.180**)
MinT-shr	85.24 ±0.65 (**.241**)	70.22 ±0.32 (**.201**)	63.67 ±0.28 (**.151**)	76.47 ±0.26 (**.162**)
MinT-ols	87.79 ±0.44 (**.231**)	74.23 ±0.61 (**.232**)	69.23 ±0.41 (**.171**)	82.56 ±1.25 (**.187**)
SHARQ	75.91 ±0.42 (**.203**)	66.57 ±0.24 (**.199**)	65.11 ±0.31 (**.159**)	74.09 ±0.32 (**.133**)
Average	63.58 ±0.72 (**.129**)			
MS	63.72 ±0.36(**.131**)			
MECATS	**55.72 ±0.73** (**.122**)			

Table 13: AE-Demand dataset, level 1

Recon \ Model	Level 2			
	AR	RNN-GRU	LSTNet	DeepAR
BU	94.23 ±0.45 (**.302**)	85.37 ±0.27 (**.214**)	83.72 ±0.55 (**.198**)	87.62 ±0.48 (**.223**)
Base	87.41 ±0.76 (**.247**)	72.34 ±0.26 (**.164**)	69.88 ±0.35 (**.147**)	80.38 ±0.78 (**.184**)
MinT-sam	87.97 ±0.42 (**.258**)	74.22 ±0.24 (**.171**)	72.46 ±0.38 (**.158**)	82.45 ±0.18 (**.204**)
MinT-shr	86.32 ±0.36 (**.235**)	71.49 ±0.46 (**.143**)	67.12 ±0.25 (**.129**)	79.48 ±0.39 (**.178**)
MinT-ols	88.72 ±0.37 (**.242**)	73.52 ±0.85 (**.166**)	71.98 ±0.37 (**.153**)	84.29 ±0.73 (**.207**)
SHARQ	78.97 ±0.45 (**.203**)	68.25 ±0.47 (**.131**)	68.01 ±0.22 (**.126**)	76.05 ±0.53 (**.191**)
Average	70.44 ±0.09 (**.124**)			
MS	68.01 ±0.22 (**.126**)			
MECATS	**62.55 ±0.14** (**.111**)			

Table 14: AE-Demand dataset, level 2

Recon \ Model	Level 3			
	AR	RNN-GRU	LSTNet	DeepAR
BU	90.33 ±0.49 (**.231**)	87.35 ±0.69 (**.225**)	82.47 ±0.28 (**.192**)	84.58 ±0.63 (**.236**)
Base	90.33 ±0.49 (**.231**)	87.35 ±0.69 (**.225**)	82.47 ±0.28 (**.192**)	84.58 ±0.63 (**.236**)
MinT-sam	91.23 ±0.43 (**.244**)	88.42 ±0.74 (**.229**)	85.63 ±0.79 (**.234**)	86.43 ±0.75 (**.256**)
MinT-shr	89.58 ±0.89 (**.241**)	86.17 ±0.73 (**.217**)	83.14 ±0.12 (**.202**)	84.92 ±0.37 (**.227**)
MinT-ols	92.24 ±0.53 (**.252**)	87.98 ±0.44 (**.229**)	85.77 ±0.87 (**.228**)	87.74 ±0.46 (**.254**)
SHARQ	90.33 ±0.49 (**.231**)	87.35 ±0.69 (**.225**)	82.47 ±0.28 (**.192**)	84.58 ±0.63 (**.236**)
Average	73.22 ±0.37 (**.135**)			
MS	82.47 ±0.28 (**.192**)			
MECATS	**71.45 ±0.43** (**.125**)			

Table 15: AE-Demand dataset, level 3

Recon \ Model	Level 0			
	AR	RNN-GRU	LSTNet	DeepAR
BU	94.23 ±0.72 (**.276**)	91.14 ±0.23 (**.238**)	91.24 ±0.37 (**.226**)	92.34 ±0.47 (**.244**)
Base	75.12 ±0.25 (**.182**)	74.01 ±0.23 (**.174**)	73.92 ±0.16 (**.162**)	73.98 ±0.22 (**.171**)
MinT-sam	87.41 ±0.33 (**.209**)	78.32 ±0.48 (**.196**)	77.14 ±0.49 (**.172**)	77.54 ±0.69 (**.181**)
MinT-shr	86.12 ±0.24 (**.196**)	77.64 ±0.25 (**.187**)	76.53 ±0.36 (**.169**)	75.88 ±0.25 (**.176**)
MinT-ols	88.03 ±0.32 (**.217**)	80.93 ±0.31 (**.205**)	79.32 ±0.75 (**.185**)	77.45 ±0.34 (**.184**)
SHARQ	74.25 ±0.31 (**.181**)	70.36 ±0.24 (**.147**)	70.55 ±0.42 (**.167**)	69.67 ±0.58 (**.168**)
Average	66.42 ±0.16 (**.128**)			
MS	69.67 ±0.58 (**.168**)			
MECATS	**63.27 ±0.73** (**.117**)			

Table 16: Wikipedia dataset, level 0

Recon \ Model	Level 1			
	AR	RNN-GRU	LSTNet	DeepAR
BU	89.76 ±0.27 (**.282**)	89.71 ±0.56 (**.212**)	90.02 ±0.24 (**.221**)	90.86 ±0.23 (**.216**)
Base	79.86 ±0.39 (**.212**)	76.33 ±0.65 (**.177**)	75.38 ±0.44 (**.165**)	74.54 ±0.47 (**.167**)
MinT-sam	88.27 ±0.38 (**.293**)	79.34 ±0.49 (**.195**)	80.23 ±0.32 (**.193**)	78.46 ±0.29 (**.191**)
MinT-shr	84.79 ±0.76 (**.263**)	75.58 ±0.27 (**.167**)	77.49 ±0.34 (**.183**)	77.45 ±0.34 (**.187**)
MinT-ols	87.82 ±0.28 (**.244**)	82.11 ±0.72 (**.204**)	82.69 ±0.48 (**.202**)	80.36 ±0.67 (**.204**)
SHARQ	72.63 ±0.36 (**.173**)	73.06 ±0.42 (**.159**)	72.34 ±0.26 (**.161**)	71.34 ±0.58 (**.179**)
Average	72.01 ±0.52 (**.157**)			
MS	68.24 ±0.33 (**.151**)			
MECATS	**65.14 ±0.46** (**.143**)			

Table 17: Wikipedia dataset, level 1

Recon \ Model	Level 2			
	AR	RNN-GRU	LSTNet	DeepAR
BU	88.16 ±0.26 (**.267**)	88.21 ±0.42 (**.237**)	86.49 ±0.43 (**.224**)	89.64 ±0.37 (**.212**)
Base	81.32 ±0.14 (**.188**)	81.29 ±0.34 (**.192**)	78.65 ±0.31 (**.188**)	77.42 ±0.36 (**.179**)
MinT-sam	86.24 ±0.34 (**.242**)	83.64 ±0.48 (**.219**)	83.67 ±0.46 (**.221**)	84.14 ±0.27 (**.187**)
MinT-shr	85.41 ±0.77 (**.239**)	80.09 ±0.12 (**.189**)	82.17 ±0.95 (**.206**)	82.13 ±0.88 (**.176**)
MinT-ols	90.31 ±0.42 (**.284**)	83.31 ±0.24 (**.214**)	84.68 ±0.74 (**.238**)	84.54 ±0.44 (**.193**)
SHARQ	74.63 ±0.35 (**.175**)	76.15 ±0.34 (**.135**)	75.56 ±0.41 (**.179**)	74.62 ±0.19 (**.155**)
Average	74.37 ±0.83 (**.147**)			
MS	74.62 ±0.19 (**.155**)			
MECATS	**69.48 ±0.33** (**.142**)			

Table 18: Wikipedia dataset, level 2

Recon \ Model	Level 3			
	AR	RNN-GRU	LSTNet	DeepAR
BU	87.62 ±0.57 (**.269**)	86.45 ±0.78 (**.243**)	85.57 ±0.12 (**.279**)	86.79 ±0.32 (**.349**)
Base	83.26 ±0.44 (**.232**)	82.23 ±0.73 (**.224**)	80.33 ±0.21 (**.207**)	79.12 ±0.23 (**.298**)
MinT-sam	86.57 ±0.63 (**.256**)	85.36 ±0.85 (**.240**)	86.02 ±0.56 (**.284**)	84.57 ±0.36 (**.314**)
MinT-shr	86.22 ±0.31 (**.244**)	81.69 ±0.49 (**.221**)	84.57 ±0.19 (**.251**)	82.89 ±0.61 (**.304**)
MinT-ols	88.23 ±0.37 (**.289**)	84.78 ±0.57 (**.216**)	86.44 ±0.34 (**.286**)	83.24 ±0.43 (**.310**)
SHARQ	78.95 ±0.25 (**.198**)	78.42 ±0.34 (**.201**)	79.63 ±0.41 (**.191**)	80.37 ±0.22 (**.301**)
Average	81.38 ±0.65 (**.278**)			
MS	79.63 ±0.41 (**.191**)			
MECATS	**75.69 ±0.76** (**.189**)			

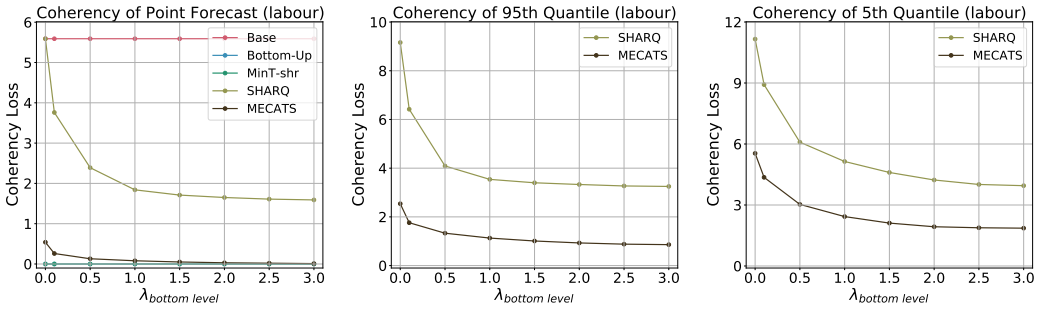
Table 19: Wikipedia dataset, level 3

Recon \ Model	Level 4			
	AR	RNN-GRU	LSTNet	DeepAR
BU	86.44 ±0.64 (**.321**)	85.12 ±0.62 (**.345**)	82.41 ±0.82 (**.179**)	84.77 ±0.49 (**.241**)
Base	86.44 ±0.64 (**.321**)	85.12 ±0.62 (**.345**)	82.41 ±0.82 (**.179**)	84.77 ±0.49 (**.241**)
MinT-sam	98.19 ±0.85 (**.588**)	87.44 ±0.22 (**.359**)	87.39 ±0.11 (**.224**)	92.21 ±1.32 (**.368**)
MinT-shr	95.89 ±0.27 (**.423**)	85.26 ±0.38 (**.332**)	86.01 ±0.37 (**.211**)	91.47 ±0.66 (**.337**)
MinT-ols	99.41 ±0.19 (**.602**)	88.41 ±0.37 (**.384**)	86.89 ±0.35 (**.220**)	93.16 ±0.93 (**.384**)
SHARQ	86.44 ±0.64 (**.321**)	85.12 ±0.62 (**.345**)	82.41 ±0.82 (**.179**)	84.77 ±0.49 (**.241**)
Average	84.68 ±0.42 (**.221**)			
MS	79.65 ±0.24 (**.186**)			
MECATS	**76.88 ±0.72** (**.163**)			

Table 20: Wikipedia dataset, level 4

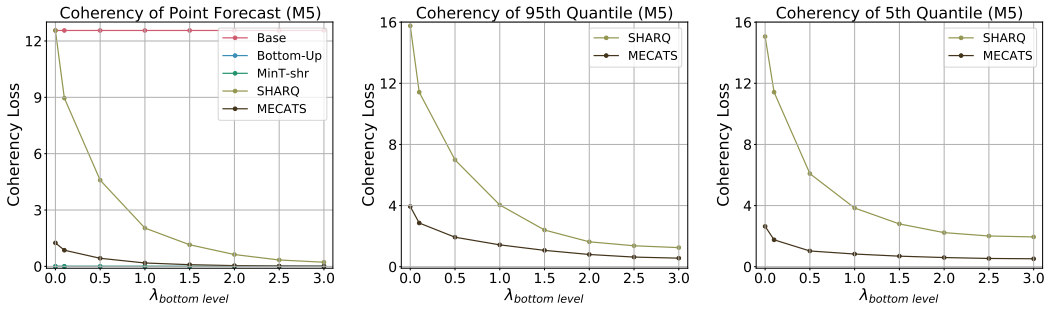
H.2 COHERENCY ANALYSIS

The following results are coherent loss computed using Eq (57), which is averaged across 5 random experiments. Note that base forecasts refer to single-model base forecast. $\lambda_{\text{bottom level}}$ is the parameter that controls the strength of regularization from the bottom level forecasts.



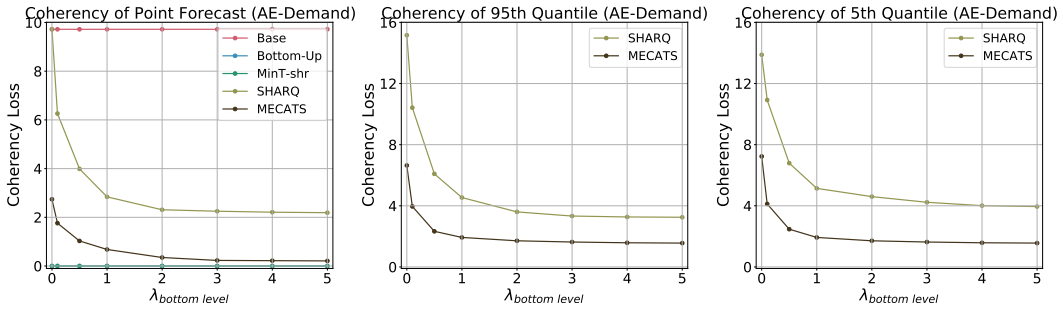
(a) (b) (c)

Figure 7: Coherency analysis, Australian Labour dataset.



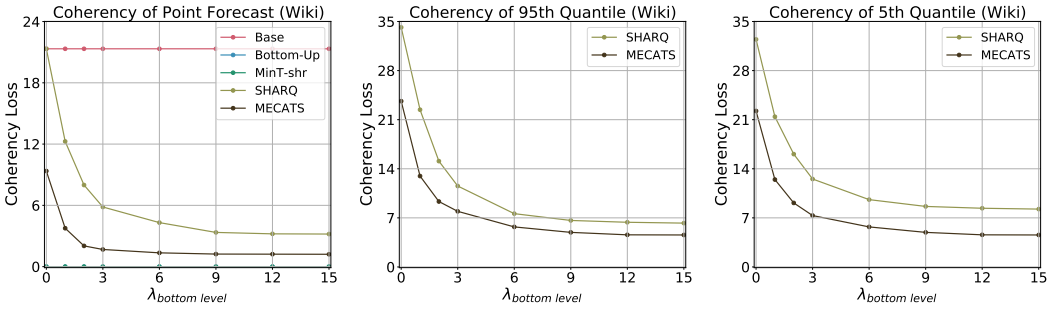
(a) (b) (c)

Figure 8: Coherency analysis, M5 dataset.



(a) (b) (c)

Figure 9: Coherency analysis, AE-Demand dataset.

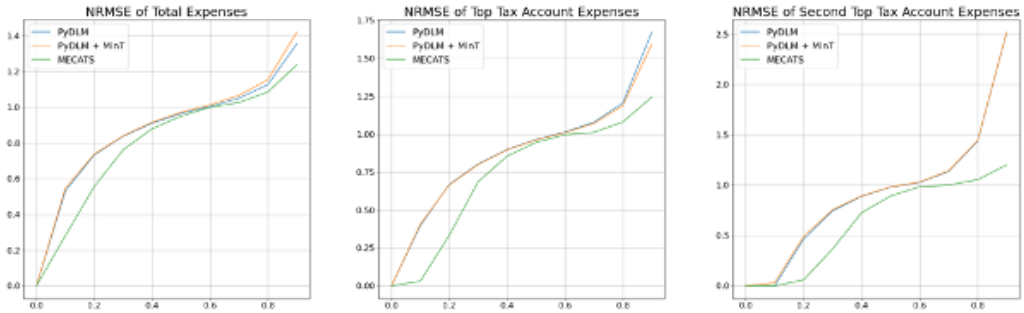


(a) (b) (c)

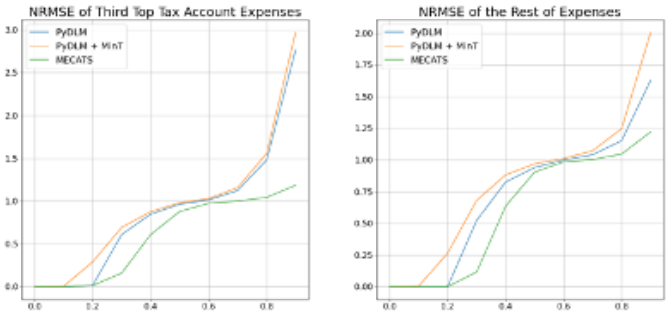
Figure 10: Coherency analysis, Wikipedia dataset.

H.3 MECATS IN DEPLOYMENT

MECATS has also been tested within a large financial software company for cash flow forecasting. The hierarchy consists of 5 vertices, where the top level vertex represents the total expense of one user, the four bottom level vertices are the first top, second top, third top and the rest of expenses of the corresponding user. We conducted experiments on 12,000 users' data, which has the same hierarchical structure, and compared with already deployed baseline methods (PyDLM, PyDLM + MinT-shr) using normalised root mean squared error (NRMSE). Figure 11 (a-e) shows the NRMSE results on corresponding vertex, where each figure represents the cumulative value of NRMSE distribution over 12,000 users at each percentile. It can be seen from the results that MECATS outperforms the baselines at every percentile. In addition, given the time series between each hierarchy are short sequences with sparsity issues at the bottom level, we did not employ deep neural network based models that are more prone to overfitting. The four experts include Auto-ARIMA, FB-Prophet, PyDLM and simple average. Figure 12 and 13 show the percentage of these experts with the highest and lowest weights over 12,000 hierarchies. The advantage of combining experts is obvious given no expert dominates the forecasts.



(a) (b) (c)



(d) (e)

Figure 11: NRMSE results on 12,000 users

Figure 12: Percentage of experts with the highest weight by vertex.

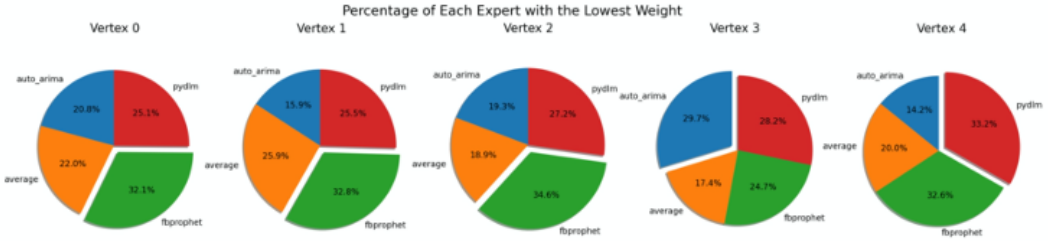


Figure 13: Percentage of experts with the lowest weight by vertex.