GMP-AR: Granularity Message Passing and Adaptive Reconciliation for Temporal Hierarchy Forecasting

Fan Zhou[1], Chen Pan[1], Lintao Ma[1], Yu Liu[1], Siqiao Xue[1] , James Zhang[1], Jun Zhou[1], Hongyuan Mei[2], Weitao Lin[1], Zi Zhuang[1], Wenxin Ning[1], Yunhua Hu[1]

[1]Ant Group, Hangzhou China
[2]TTIC, Chicago USA
{hanlian.zf, bopu.pc, maining.mlt}@antgroup.com

Abstract

Time series forecasts of different temporal granularity are widely used in real-world applications, e.g., sales prediction in days and weeks for making different inventory plans. However, these tasks are usually solved separately without ensuring coherence, which is crucial for aligning downstream decisions. Previous works mainly focus on ensuring coherence with some straightforward methods, e.g., aggregation from the forecasts of fine granularity to the coarse ones, and allocation from the coarse granularity to the fine ones. These methods merely take the temporal hierarchical structure to maintain coherence without improving the forecasting accuracy. In this paper, we propose a novel granularity message-passing mechanism (GMP) that leverages temporal hierarchy information to improve forecasting performance and also utilizes an adaptive reconciliation (AR) strategy to maintain coherence without performance loss. Furthermore, we introduce an optimization module to achieve task-based targets while adhering to more real-world constraints. Experiments on real-world datasets demonstrate that our framework (GMP-AR) achieves superior performances on temporal hierarchical forecasting tasks compared to state-of-the-art methods. In addition, our framework has been successfully applied to a real-world task of payment traffic management in Alipay by integrating with the task-based optimization module.

Introduction

Time series forecasting is widely used in many important real-world tasks, such as supply chains management (Athanasopoulos, Ahmed, and Hyndman 2009; Rostami-Tabar et al. 2013), traffic flow prediction (Laptev et al. 2017; Li et al. 2018). Forecasting of different temporal granularity is essential for informed decision-making in downstream tasks. In management and operational scenarios, forecasting at different time scales is necessary to formulate plans that span from short-term to long-term. For instance, in the retail sales industry, managers need to make decisions about how much inventory to order and how to allocate it to different stores. Daily predictions can be used to plan the daily commodity distribution to each store, while weekly predictions can be used to purchase commodities in bulk. This ensures that the stores have the proper amount of inventory on hand to meet customer demand, while also minimizing the cost of carrying excess inventory.

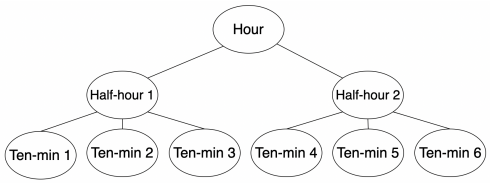

Figure 1: An example of temporal hierarchical time series (HTS) structure for ten-minute frequency as the bottom level, half-hour and hour frequencies as aggregated levels of coarser granularity.

As shown in Fig. 1, a temporal hierarchical structure is formed naturally with different levels of temporal granularity. The time series of the finest granularity makes up the bottom level, and it is aggregated into upper levels of coarser granularity. This type of time series is commonly referred to as a *temporal hierarchical time series* or temporal HTS (Athanasopoulos et al. 2017). Temporal HTS forecasting tasks are challenging because they require accurate prediction results for different granularity while satisfying aggregation (coherence) constraints that time series at upper levels are the aggregation of those at lower levels. It is difficult to produce accurate predictions for all levels with a single model because time series at each level have different granularity and exhibit their own specific dynamics, such as different trends and various seasonalities. In addition, independent forecasts (i.e., *base forecasts*) are unlikely to adhere to the coherence constraints across temporal granularity. The inaccurate and incoherent forecasts would result in inefficient planning of downstream tasks and pose risks to decision-making.

Previous works on temporal hierarchy forecasting can be categorized into two types: vanilla forecasting methods for temporal HTS and multivariate HTS methods applied to temporal hierarchy. Most temporal HTS methods (Athanasopoulos et al. 2017; Theodosiou and Kourentzes 2021) are statistical and theoretically explainable, but forecast performance is limited when applied to complex real-world datasets, implemented in the THIEF package.

The multivariate HTS methods typically follow a two-

stage paradigm. In the first stage, base forecasts are generated independently for each time series in the hierarchy with statistical or deep learning models. In the second stage, base forecasts are adjusted via reconciliation to ensure coherence.

Most methods ignore the temporal hierarchical information among levels of different granularity in forecasting, one exception is COPDeepAR (Rangapuram et al. 2023), which leverages graph neural networks (GNNs) to extract structure information. However, GNNs introduce noise connections between nodes in the tree structure and cause performance loss. As for reconciliation, traditional statistical methods (e.g., MinT (Wickramasuriya, Athanasopoulos, and Hyndman 2019)) derive coherent forecasts relying on strong statistical assumptions. The end-to-end method (Rangapuram et al. 2021) ensures coherence without strong assumptions by using a closed-form projection. However, this may reduce the forecasting performance because the adjustment scale is sometimes either too large or small and could not adapt to node values.

In this paper, we propose a framework (GMP-AR) that efficiently utilizes information among granularity at different levels of temporal hierarchy to improve the forecast performance while maintaining coherence. In summary, our contributions are as follows:

- We propose a granularity message passing mechanism (GMP) to enrich the information of temporal input for each node and integrate the temporal hierarchical features among different granularity to generate base forecasts. This is the first method that employs the temporal hierarchical structure for forecasting tasks to the best of our knowledge.
- We provide an adaptive reconciliation method to produce coherent results without forecasting performance loss by utilizing the node-dependent weighted optimization to precisely control the adjustment scale node by node.
- We integrate our framework with an optimization module to solve real-world problems with task-based targets and realistic constraints.
- Experiments on real-world time series datasets demonstrate that our proposed approach achieves significant improvements over the state-of-the-art baselines and our framework combined with the optimization module has been deployed on the payment traffic management system in Alipay.

Preliminary and Related Work

In this section, we provide background on temporal hierarchical time series and introduce necessary notations here for ease of understanding.

Preliminaries: Temporary Hierarchical Structure

The time series is denoted by $\{y_t; 1 \le t \le T\}$ with a fixed sampling frequency (*base frequency*), which is of the finest granularity. The time series of the coarser granularity is aggregated by non-overlapping and regularly sampled values y_t as Eq. (1) shows. For the time series in Fig. 1, the base frequency is ten-minute. It is aggregated into a half-hour value with three ten-minute sample values, and an hourly frequency with two half-hour sample values.

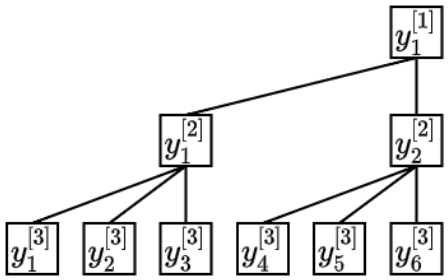

Figure 2: The notation of each node in Fig. 1 observed at timestamp $t = 6$ and $\tau = 1$.

The temporal HTS can be expressed as a tree structure with a linear aggregation constraint called *coherence constraint*, formulated via an aggregation matrix $\mathbf{S} \in \mathbb{R}^{n\times m}$ (see Eq. (2)), where $m = 6$ is the number of leaf nodes and $n = 9$ is the total number of nodes in the tree in Fig. 1.

We consider a *p-level* aggregated temporal hierarchical structure. The number of nodes at each level can be presented as $N_p = \{n_k : k = 1, 2, \ldots, p\}$, where $n_1 = 1$, and $n_p = m$. Take Fig. 1 as an example, $N_3 = \{1, 2, 6\}$. The sample values at timestamps in the time series at level k can be expressed as

$$y^{[k]}_{(\tau-1)n_k+j} = \sum_{t=(\tau-1)m+(j-1)m_k+1}^{(\tau-1)m+jm_k} y_t,\ 1 \le j \le n_k, \quad (1)$$

where $\tau \in \{1, \ldots, \lfloor \frac{T}{m} \rfloor\}$ is the observation index of the most aggregated time series (i.e., the one at the root level), and j is the observation index at level k in the temporal structure, and $m_k = \frac{m}{n_k}$ is the number of leaves aggregated to generate the node values at the k-th level. One can see that the values $\{y^{[k]}_{\tau n_k}, 1 \le k \le p\}$ are observed at the same time point τm.

In the temporal hierarchical structure, the leaf nodes are called the *bottom-level* series: $\mathbf{y}^{[p]}_\tau \in \mathbb{R}^m$, and the remaining nodes are termed *upper-levels* series: $\{y^{[1]}_\tau, \ldots, \mathbf{y}^{[p-1]}_\tau\}$. The number of nodes at upper levels is $r = \sum_{i=1}^{p-1} n_i$. Obviously, the total number of nodes $n = r + m$, and $\mathbf{y}_\tau := [y^{[1]}_\tau, {\mathbf{y}^{[2]}_\tau}^\mathsf{T}, \ldots, {\mathbf{y}^{[p]}_\tau}^\mathsf{T}]^\mathsf{T} \in \mathbb{R}^n$ contains observations at time τ for all levels, which satisfies

$$\mathbf{y}_\tau = \mathbf{S}\mathbf{y}^{[p]}_\tau, \quad (2)$$

where $\mathbf{S} \in \{0, 1\}^{n\times m}$ is an aggregation matrix.

Taking Fig. 1 as an example, the aggregation matrix is

$$\mathbf{S} = \begin{pmatrix} \mathbf{S}_{\text{sum}} \\ \mathbf{I}_6 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ & & \mathbf{I}_6 & & & \end{pmatrix},$$

the total number of nodes in the hierarchy is $n = 1 + 2 + 6$, and the number of notes at upper-levels is $r = 3$. At each time index τ, the coherence constraint of Eq. (2) can be represented as (Rangapuram et al. 2021)

$$\mathbf{A}\mathbf{y}_\tau = \mathbf{0}, \tag{3}$$

where $\mathbf{A} := (\mathbf{I}_r | -\mathbf{S}_{\text{sum}}) \in \{0,1\}^{r\times n}$, $\mathbf{0} \in \mathbb{R}^r$ is the zero vector, and $\mathbf{I}_r \in \mathbb{R}^{r\times r}$ is the identity matrix.

Related Work

Temporal Hierarchical Forecasting. The temporal hierarchical forecasting defined in (Athanasopoulos et al. 2017) requires coherence constraint among temporal granularity, and forecasting tourism data with coherence on both cross-sectional and temporal dimensions is studied in (Kourentzes and Athanasopoulos 2019). They rely on statistical tools for the reconciliation of base forecasts. (Chen, Ma, and Lin 2021) is also aimed at improving the forecasting accuracy of time series with different levels of granularity. However, this work does not address coherence. Recently, an end-to-end model (COPDeepAR) is proposed in (Rangapuram et al. 2023) that focuses on generating coherent probabilistic forecasts for time series with various levels of granularity. This is achieved by utilizing GNNs to extract inter-level information and applying a closed-form projection reconciliation method to maintain coherence. GNNs introduce spurious connections among all nodes but valid information actually only exists between parent and child nodes. Therefore, it fails to capture structure information because of noise connections and results in loss of forecasting performance as pointed out in (Zhou et al. 2023). In our works, we propose a more efficient information extraction mechanism between different granularity in the temporal hierarchy.

Method

In this section, we introduce our framework (GMP-AR) that takes advantage of the message-passing to extract valid information among different granularity in the temporal hierarchy to improve forecasting performance and achieve adaptive reconciliation to maintain coherence. Finally, we integrate an efficient optimization module with our framework to solve real-world problems with task-based targets and realistic constraints. The overall architecture is shown in Fig. 3, which consists of three main components:

- *Granularity message passing* module: This module leverages temporal hierarchy information between different granularity to generate base forecasts over the prediction horizon across all levels and adaptive weights;
- *Adaptive reconciliation* module: This module utilizes adaptive weights combined with projection reconciliation to produce node-dependent adjustment to maintain coherency in an end-to-end way;
- *'Plug-and-play' optimization* module: This module adapts to real-world problems with task-based targets and realistic constraints.

Granularity Message Passing

In this section, we introduce our granularity message-passing module that incorporates both temporal and granularity hierarchical information to generate the base forecasts and the weights for adaptive reconciliation. This module consists of four sub-modules: granularity input transformation, temporal feature extractor, and two granularity feature fusion modules.

Granularity Input Transformation This module integrates the input information of different granularity in the hierarchy into temporal inputs by *top-down proportion transformation* and *child distribution modeling* as in Fig. 4.

Top-Down Proportion Transformation. Similar to the top-down proportion models in TDProb (Das et al. 2022), we transform the inputs $\{\mathbf{y}_1, \ldots, \mathbf{y}_\tau\}$ as the fractions/proportions according to root time-series which is disaggregated into its child time series as in Fig. 4. The detail is as follows

$$a_{\tau,j}^{[k]} = \frac{y_{(\tau-1)n_k+j}^{[k]}}{y_\tau^{[1]}}, \quad \tau \in \left\{1, 2, \ldots, \lfloor \frac{T}{m} \rfloor\right\}, \tag{4}$$

where $a_{\tau,j}^{[k]}, 1 \le j \le n_k$ is the fraction input at time τ. Then we sort the fraction inputs from the root node to the current node to build the top-down proportion inputs $\boldsymbol{\alpha}_{\tau,j}^{[k]} = [a_{\tau,l_2}^{[2]}, \ldots, a_{\tau,j}^{[k]}], k \ge 2$, where l_2 means the node index at the second level on the path from the root to the current node, and $l_i = \lceil l_{i+1}/\frac{n_{i+1}}{n_i} \rceil$. We use fraction inputs starting from the second level, as the fraction for the root level is always equal to 1 and does not provide any meaningful information for the temporal extraction module.

The intuition behind this approach is as follows: 1) As pointed out in (Das et al. 2022), the disaggregation proportions of nodes at the bottom level are more predictable compared to raw values of time series; 2) Incorporating the information of nodes at the top level with coarser granularity into the leaf nodes of finer granularity at the bottom level to enhance temporal stability, which is based on the analysis that the seasonality of the top level is clearer and the evolutionary dynamic is 'smoother' (Rostami-Tabar et al. 2013; Taieb, Taylor, and Hyndman 2021).

Child Distribution Modeling. This sub-module integrates the disaggregation proportion distribution of the child nodes with finer granularity into the temporal input of the parent node. This enriches the finer granularity information of the parent node, which help capture dynamic patterns more effectively for parent node with coarser granularity, improving the overall forecasting accuracy.

Our approach utilizes deep models to extract the distribution of child nodes' disaggregation proportions for each parent node. Specifically, we assume that each node at the same level k has the same number of children m_k, so the hierarchical structure is fixed for each upper level. Given the fixed structure, MLP (Gardner and Dorling 1998) is an efficient tool to extract features,

$$\mathbf{b}_{\tau,j}^{[k]} = \text{MLP}\left([a_{\tau,(j-1)m_k+l}^{[k+1]} : 1 \le l \le m_k]\right), k < p, \tag{5}$$

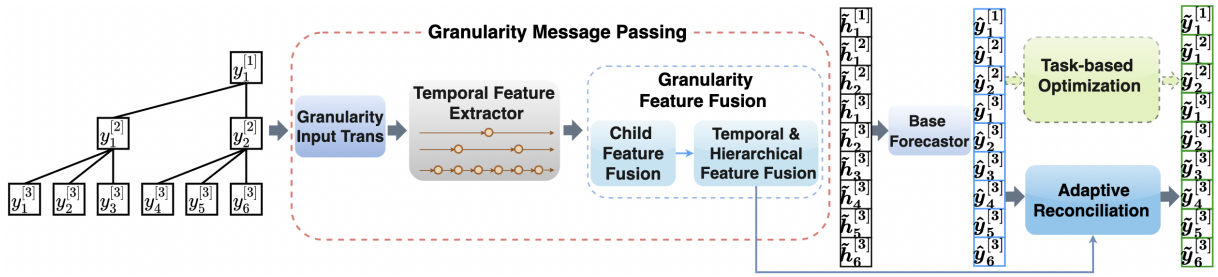

Figure 3: The architecture of GMP-AR: the red dashed box is the granularity message passing component including the granularity input transformation module, temporal feature extractor and two granularity feature fusion modules. This component generates the representation of nodes used to generate base forecasts and adaptive weights. The light green dashed box is the reconciliation module that produces the final results, including the adaptive reconciliation module for forecasting tasks and task-based optimization module to adapt to general real-world tasks.

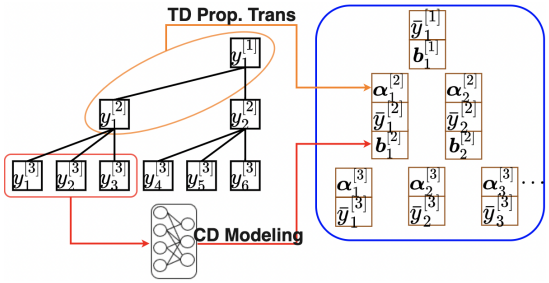

Figure 4: The process of granularity input transformation: the orange cycle is the top-down proportion transformation module that produces the disaggregation proportions for child nodes, the red rectangle is the child distribution modeling module which extracts valid information of finer granularity for parent nodes; the blue box concatenate these processed inputs with normalized value and put them into temporal feature extractor to extract dynamic patterns.

where $\mathbf{b}_{\tau,j}^{[k]}$ represents the hidden features of child node's distribution for the parent node $y_{(\tau-1)n_k+j}^{[k]}$ and $\{a_{\tau,(j-1)m_k+l}^{[k+1]} : 1 \leq l \leq m_k\}$ is the proportion input of its child set.

Temporal Feature Extraction. This module extracts temporal features for nodes at the same level (same granularity) as follows:

$$h_{\tau,j}^{[k]} = \text{UPDATE}^{[k]}\left(h_{\tau,j-1}^{[k]}, [\boldsymbol{\alpha}_{\tau,j}^{[k]}, \mathbf{b}_{\tau,j}^{[k]}, \bar{y}_{\tau,j}^{[k]}]; \theta^{[k]}\right),$$
$$\mathbf{H}_\tau = [h_{\tau,1}^{[1]}, h_{\tau,1}^{[2]}, \ldots, h_{\tau,m}^{[p]}],$$

where $\boldsymbol{\alpha}_{\tau,j}^{[k]}$ is the top-down proportion inputs of node j at level k and time point τ, $\mathbf{b}_{\tau,j}^{[k]}$ is the distribution feature of the child set, and $\bar{y}_{\tau,j}^{[k]}$ is the scaled value of the sample at the same level over the whole past observations as follows:

$$\bar{y}_{\tau,j}^{[k]} = \frac{y_{(\tau-1)n_k+j}^{[k]}}{\max\{y_1^{[k]}, \ldots, y_{\tau n_k}^{[k]}\}}, 1 \leq j \leq n_k. \quad (6)$$

Since the root node has no ancestors and the leaf nodes have no child, we remove $\boldsymbol{\alpha}_\tau^{[1]}$ for the coarsest granularity and $\mathbf{b}_{\tau,j}^{[p]}$ for the finest granularity. The module UPDATE$^{[k]}$ is a temporal pattern extraction function at level k, and $\theta^{[k]}$ is the parameters of the function. Any type of recurrent-type neural networks can be adopted as the temporal feature extraction module, such as the RNN variants, TCN (Bai, Kolter, and Koltun 2018), and NBeats (Oreshkin et al. 2019). In our experiments, we use GRU (Chung et al. 2014) for simplicity.

It is worth emphasizing that temporal HTS differs from multivariate HTS in that different levels do not share the same temporal features. In multivariate HTS, all nodes are of the same temporal granularity and may have similar trends or seasonality. However, in temporal HTS, nodes at different levels have different granularity and different context lengths. Nodes at the top levels, which have coarser granularity, have smaller context lengths (Rangapuram et al. 2023). In other words, different levels have their own dynamic patterns (trend, seasonality, etc.), which requires different specific parameters $\theta^{[k]}$ of temporal updating function for each level.

Child Granularity Feature Fusion This module integrates temporal features from the finer granularity of child nodes to their parent at top levels, enhancing the framework's ability to adapt to the variation of dynamic patterns, such as sudden trend changes due to some special events, e.g., Double 11, black Friday. This because the value aggregation relationship between the parent node and child nodes determines the changes in dynamic patterns at the finer granularity would influence the ones at the coarser granularity.

Unlike directly modeling the disaggregation proportion distribution across child nodes, this module extracts valid temporal features from child nodes, which may have different contributions to their parent. CNN and attention mechanism (Vaswani et al. 2017) are effective tools to extract valid patterns from various features. However, we assume that all nodes at the same level have the same number of children, which indicates the tree structure at the same level is fixed. For a fixed structure, CNN is more appropriate because it has lower computation cost (see Appendix E) and better performance as demonstrated in the ablation study (see Appendix

F). The detail is as follows:

$$\begin{aligned}\hat{h}_{\tau,j}^{[k]} &= \text{Conv}^{[k]}\left([h_{\tau,(j-1)m_k+l}^{[k+1]} : 1 \le l \le m_k]\right) \\ &= \sum_{l=1}^{m_k} w_l^{[k]} h_{\tau,(j-1)m_k+l}^{[k+1]},\end{aligned}$$

where $\text{Conv}^{[k]}$ is the CNN kernel for level k, and the $w_l^{[k]}$ is the kernel weight of child l at level k.

Temporal and Hierarchical Granularity Feature Fusion. This module integrates the temporal and hierarchical features among granularity for each node to enrich the dynamic and structure information of node representation, which is used to generate the base forecasts and the weights for adaptive reconciliation.

We adopt CNNs to extract valid information across the temporal and granularity hierarchy domains as follows:

$$\begin{aligned}\tilde{h}_{\tau,j}^{[k]} &= \text{Conv}(\hat{\mathbf{H}}_{\tau,j}^{[k]}; \theta), \\ \hat{\mathbf{H}}_{\tau,j}^{[k]} &= \begin{pmatrix} h_{\tau,1}^{[k]} & \hat{h}_{\tau,1}^{[k]} \\ \vdots & \vdots \\ h_{\tau,j}^{[k]} & \hat{h}_{\tau,j}^{[k]} \end{pmatrix},\end{aligned}$$

where all nodes share the same extraction parameters, $\tilde{h}_{\tau,j}^{[k]}$ are used as inputs of MLPs (the module can be replaced with other neural layers, such as seq2seq (Sutskever, Vinyals, and Le 2014), transformers (Vaswani et al. 2017), and CNNs) to generate the base forecasts $\hat{\mathbf{y}}_\tau$ and reconciliation weights $\mathbf{w}$.

Adaptive Reconciliation

In this section, we introduce our adaptive reconciliation module by utilizing node-dependent weights that incorporate temporal and hierarchical information. These weights are used to control the adjustment scale node by node in reconciliation, improving the prediction performance and ensuring coherence.

Under the assumption that base forecasts are highly accurate, the closed-form projection reconciliation only minimizes the distance between the coherent result with the base forecast (Rangapuram et al. 2021). However, the optimization target here is to minimize the mean distance among all nodes, which may result in a performance loss for several reasons. 1) The scale of values among different nodes can be diverse, and mean distance minimization may cause a relatively large adjustment to the node with a small value, while only slight adjusting to the nodes with large values. Since the values at higher levels of the hierarchy have a coarser granularity that are larger than those at lower levels, the reconciliation mainly provides efficient adjustments to the upper level values. 2) Since different nodes have distinct levels of forecasting accuracy, the reconciliation adjustment scale should also be insignificant for nodes with high accuracy. In order to achieve adaptive adjustment node by node, node-dependent weights are imposed to the optimization target to resolve these issues as follows

$$\begin{aligned}\tilde{\mathbf{y}}_\tau = &\underset{\mathbf{y}\in\mathbb{R}^n}{\arg\min} \|\mathbf{w}(\mathbf{y} - \hat{\mathbf{y}}_\tau)\|_2 \\ \text{s.t.} \quad &\mathbf{A}\mathbf{y} = \mathbf{0},\end{aligned} \tag{7}$$

where $\mathbf{w} \in \mathbb{R}^{n\times n}$ is a diagonal matrix with weights on the diagonal. Then we apply the method of Lagrange multiplier to derive the closed-form solution (see the proof in Appendix A):

$$\tilde{\mathbf{y}}_\tau = \mathbf{M}\hat{\mathbf{y}}_\tau, \tag{8}$$

where $\mathbf{M} = (\mathbf{I} - \tilde{\mathbf{w}}^{-1}\mathbf{A}^\mathsf{T}(\mathbf{A}\tilde{\mathbf{w}}^{-1}\mathbf{A}^\mathsf{T})^{-1}\mathbf{A})\hat{\mathbf{y}}_\tau$, $\tilde{\mathbf{w}} = \mathbf{w}^\mathsf{T}\mathbf{w}$.

The node-dependent weights could be hyperparameters for simpler structures or realistic tasks, which is explainable and easy to implement. However, it requires prior domain knowledge of each scenario to set proper weights, which causes inflexibility to new scenarios and the improvement of performance is limited. Our framework utilizes temporal and hierarchical granularity to generate the node-dependent weights, which has superior performance improvement and adaptability than human settings as shown in Appendix F.

Task-based Optimization in Real-world Scenarios

This module introduces an optimization module, which can be applied in our forecasting framework (GMP-AR) to solve real-world problems with task-based targets and realistic constraints. The detailed formulation is as follows:

$$\begin{aligned}&\mathcal{J}(\hat{\mathbf{y}}) = \min_{\mathbf{y}} f(\hat{\mathbf{y}}, \mathbf{y}) \\ &\text{s.t.} \quad \begin{cases}\mathbf{A}\mathbf{y} = \mathbf{0}, e_j = 0,\ j = 1, \ldots, n_{\text{eq}}, \\ g_i(\mathbf{y}, \hat{\mathbf{y}}) \le 0,\ i = 1, \ldots, n_{\text{ineq}}.\end{cases}\end{aligned} \tag{9}$$

The target functional f can take various forms of functions besides Euclidean distance (such as cosine similarity and L^d-norm ($d \ge 1$)). The e_j terms represent task-based equality constraints other than coherence constraint. The n_{eq} term is the number of equality constraints, the g_i term is an inequality constraint, and n_{ineq} term is the number of task-based inequality constraints. As shown in Eq. (9), our task is transformed into an optimization task, where legacy methods such as L-BFGS or Powell (Nocedal and Wright 2006) can be applied. Many useful optimization methods have been encapsulated in *Scipy.optmize*, which is used in our experiment in Alipay with finance targets and supervised constraints.

Experiments

In this section, we empirically evaluate our proposed method on public time series datasets (details are in Appendix B) and representative SOTA methods (details are shown in Appendix C). We then present the successful application of our method with the task-based optimization module in the real-world payment traffic management system of Alipay. The experiment setup is described in Appendix G and the implementation code is included in the supplementary files.

Result Analysis

In this section, we evaluate the forecasting performance of our proposed method on three public datasets. We use the mean absolute percentage error to measure the performance of each method, both at the bottom level (b-MAPE) that has the finest granularity and across all nodes (MAPE).

The experimental results are shown in Table 1. The top section shows the results of the traditional statistic method,

Dataset	Electricity		Traffic		Exchange Rate	
Metrics	b-MAPE	MAPE	b-MAPE	MAPE	b-MAPE	MAPE
THIEF-ARIMA-BU	0.1895	**0.1319**	0.7501	0.5518	0.0095	0.0092
THIEF-ETS-BU	0.3712	0.3111	0.8815	0.8600	0.0100	0.0096)
THIEF-THETA-BU	0.2071	0.1434	0.7064	0.7065	0.0102	0.0099
THIEF-ARIMA-OLS	0.1896	0.1319	0.8293	0.5711	0.0118	0.0113
THIEF-ETS-OLS	0.2208	0.1438	1.0477	0.6871	0.0133	0.0102
THIEF-THETA-OLS	0.2256	0.1520	1.6886	0.8952	0.0099	0.0094
THIEF-ARIMA-MSE	0.1906	0.1333	0.7000	0.5141	0.0097	0.0093
THIEF-ETS-MSE	0.2142	0.1473	0.7315	0.6872	0.0103	0.0100
THIEF-THETA-MSE	0.2036	0.1373	0.9467	0.7035	0.0094	0.0090
DeepAR-BU	0.4168(0.0182)	0.4210(0.0154)	1.3439(0.0172)	0.8885(0.0219)	0.0138(0.0027)	0.0136(0.0030)
NBeats-BU	0.2683(0.0383)	0.3000(0.0318)	0.5601(0.0570)	0.4292(0.0377)	0.0196(0.0062)	0.0192(0.0064)
Autoformer-BU	0.4059(0.1022)	0.4049(0.0859)	1.6798(0.2757)	0.9505(0.1362)	0.0092(0.0006)	0.0087(0.0006)
DeepAR-Proj	0.4167(0.0042)	0.3889(0.0023)	3.3717(0.0290)	1.6043(0.0138)	0.0114(0.0019)	0.0111(0.0019)
NBeats-Proj	0.2432(0.0173)	0.2799(0.0203)	0.7422(0.0676)	0.5308(0.0297)	0.0211(0.0072)	0.0207(0.0075)
Autoformer-Proj	0.2754(0.0047)	0.3024(0.0064)	1.1376(0.0758)	0.6768(0.0382)	0.0111(0.0007)	0.0107(0.0007)
DeepAR-TDProb	0.1761(0.0055)	0.2370(0.0035)	1.0536(0.0296)	0.6455(0.0126)	0.0096(0.0008)	0.0094(0.0008)
NBeats-TDProb	0.1830(0.0081)	0.2395(0.0098)	1.0744(0.0339)	0.6662(0.0142)	0.0161(0.0060)	0.0159(0.0061)
Autoformer-TDProb	0.2052(0.0243)	0.2704(0.0268)	0.9894(0.0395)	0.6244(0.0198)	0.0120(0.0035)	0.0118(0.0035)
HierE2E	0.2346(0.0148)	0.2980(0.0139)	0.4728(0.0180)	0.4768(0.0128)	0.0145(0.0017)	0.0136(0.0016)
SHARQ	0.2101(0.0058)	0.2730(0.0051)	0.5041(0.0086)	0.5041(0.0128)	0.0156(0.0043)	0.0138(0.0033)
COPDeepAR	0.6088(0.0531)	0.7309(0.0542)	0.8345(0.0058)	0.8202(0.0060)	0.1247(0.0274)	0.1246(0.0275)
SLOTH	0.1765(0.0022)	0.2424(0.0027)	0.4621(0.0026)	0.4616(0.0019)	0.0114(0.0003)	0.0110(0.0003)
GMP-BU	0.1711(0.0074)	0.2288(0.0079)	0.4484(0.021)	0.3932(0.0145)	0.0107(0.0014)	0.0102(0.0015)
GMP-TDProb	0.1731(0.0026)	0.2331(0.0044)	0.9989(0.0301)	0.6047(0.0112)	0.0088(0.0009)	0.0085(0.0009)
GMP-Proj	0.1937(0.0143)	0.2481(0.0131)	0.4868(0.0243)	0.4083(0.0094)	0.0089(0.0024)	0.0086(0.0025)
GMP-AR	**0.1499(0.0021)**	0.2158(0.003)	**0.4289(0.0202)**	**0.3798(0.0066)**	**0.0085(0.0021)**	**0.0082(0.0021)**

Table 1: B-MAPE (bottom MAPE) and MAPE metric values over five independent runs for baselines such as traditional reconciliation methods, deep learning methods, and popular multivariate HTS methods, as well as our approach. The values in brackets are the variances over five runs.

Thief; The second section shows results from deep neural networks methods with bottom-up (BU), closed-formed projection (Proj) (Rangapuram et al. 2021) and top down proportion (TDProb) (Das et al. 2022) reconciliation; The third section shows the results of popular multivariate hierarchical time series forecasting methods (HierE2E, SHARQ, COPDeepAR, SLOTH); The bottom section shows the results of our forecasting approach (GMP) and the combination of our forecasting mechanism with popular reconciliation methods (BU, Proj, TDProb), as well as our adaptive reconciliation with weighted projection (AR) method.

For the bottom-level metrics (b-MAPE), DeepAR (Salinas et al. 2020) with TDProb reconciliation performs the best on the Electricity and Exchange Rate datasets, and NBeats with BU reconciliation performs the best on the Traffic datasets. In general, the performance of statistical methods in Thief is more stable than the deep learning models, but deep models achieve the best performances across all baselines. One can observe that models of best performance on b-MAPE do not necessarily perform as well on MAPE, which is caused by different level contributions to the overall performance. For example, the temporal patterns and scales of different levels can be diverse. For overall metrics, statistic methods in Thief outperform other baselines and our methods on Electricity datasets, and multivariate HTS methods have superior performance than deep learning methods. This because HTS methods take all nodes into consideration while modeling.

Our proposed approach, GMP-AR, delivers an average performance increase of over $2\% \sim 3\%$ compared to other methods on both b-MAPE and MAPE metrics in most scenarios on the Electricity and Traffic datasets. Specifically, GMP with AR Projection achieves the best performance for b-MAPE among all models on all three datasets, and it also performs the best on MAPE on Traffic and Exchange Rate datasets. In addition to adaptive reconciliation, we also combine our forecasting mechanism (GMP) with the aforementioned popular reconciliation methods (BU, Proj TDProb). These combinations also achieve higher accuracy than the baselines. We also assess the performance across all levels (Appendix D) and running time (Appendix E), and the ablation study for each component in the GMP forecasting framework (Appendix F).

In conclusion, our GMP-AR mechanism achieves the best forecasting accuracy by utilizing both temporal and granularity hierarchical information. The adaptive reconciliation produces coherent results while also improving the forecasting performance.

Alipay Payment Traffic Management

Alipay is a world-leading third-party online payment service platform that provides billions of users with online and mobile payment services. These services support amount to billions of transactions and trillions of dollars daily, which has

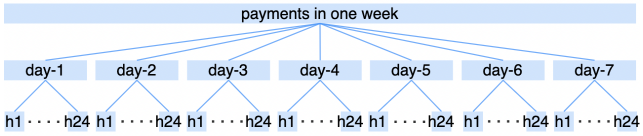

Figure 5: The temporal hierarchical structure of mobile payment service of Alipay, with weekly traffic forecast as the root node, daily forecast as the second level, and hourly forecast as leaf nodes ($N_3 = \{1, 7, 168\}$).

been widely recognized as a significant contribution to the orderly development and digital upgrading of society and economy. To ensure a stable and reliable payment experience for users, it is necessary to forecast future payment traffic in advance to ensure that the massive number of payment transactions and amounts can be effectively supported at the infrastructure and operational levels. In the temporal dimension, the manager needs to reschedule the underlying system resource under different granularity, i.e., hourly, daily, and longer-term like weekly. This way could help the system to support billions of payments efficiently and energy-savingly. Therefore, it is necessary to forecast payment traffic at hourly, daily, and multi-day granularity. This creates a natural three-level tree structure (see Fig. 5). In this temporal hierarchical structure, the coherence of different granularity forecasts needs to be satisfied.

The real-world task of hierarchical time series (HTS) forecasting differs from theoretical temporal HTS forecasting for its more task-based targets and realistic constraints, which are listed as follows: 1) Target - not only must forecast accuracy be considered but also the trend of the forecast, which is vital for business corporations that interact with external banks. 2) Constraint - the root node, i.e., the base daily forecast, cannot be altered due to the historical requirements of the regulatory authority. 3) Constraint - the child nodes, i.e., the scale of adjustment must not be more than 20 percent due to the risk regulations set by the authority. These requirements can be formulated as

$$\mathcal{J}(\hat{\mathbf{y}}) = \arg\min_{\mathbf{y}} \|\hat{\mathbf{y}} - \mathbf{y}\|_2 + \beta \left(1 - \frac{\hat{\mathbf{y}}\mathbf{y}}{\|\hat{\mathbf{y}}\|\|\mathbf{y}\|}\right)$$
$$\text{s.t.} \quad \begin{cases} \mathbf{A}\mathbf{y} = \mathbf{0}, \\ \mathbf{y}[0] - \hat{\mathbf{y}}[0] = 0, \\ abs(\mathbf{y} - \hat{\mathbf{y}}) - 0.2\hat{\mathbf{y}} \leq \mathbf{0}, \end{cases} \tag{10}$$

where we append cosine similarity with a hyperparameter multiplier β to optimization targets to improve trend forecasting performance, and also take additional two real-world constraints into consideration.

The results in Fig. 6 show that vanilla GMP-AR performs the best on prediction accuracy by 5%-7% without considering trend targets and the other two task-based constraints compared to the online baseline model, DeepAR. However, the trend accuracy is lower by 5%. Our GMP-AR combined with the optimization module achieves 3%-5% improvements on MAPE and remains similar trend accuracy as the baseline. In other words, this method achieves a compromise between forecast and trend accuracy. Besides, the

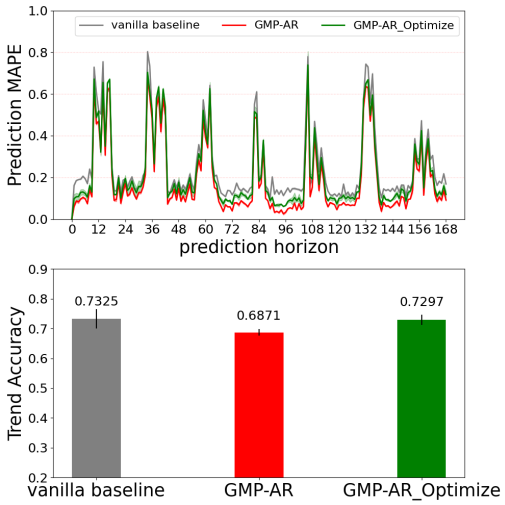

Figure 6: Results of five independent runs on Alipay scenarios of three methods, i.e., DeepAR (online baseline), GMP-AR, and GMP-AR with optimization module (lower MAPE and higher trend accuracy are better). Our GMP-AR with optimization module improves MAPE by 3.8% over baselines without performance loss on trend accuracy.

results of the optimization module adhere to all task-based constraints, while vanilla GMP-AR only satisfies the coherence constraint.

Conclusion

In this paper, we introduced a novel structure-learning framework for temporal hierarchical time series (GMP-AR). Our framework incorporated temporal and hierarchical information with a message passing mechanism to improve the performance of base forecasts, which is consisted of several modules, i.e., the granularity input transformation module that enriches the information of the temporal input of each node and the other two granularity feature fusion modules integrating the temporal and hierarchical features. For reconciliation, we proposed an adaptive reconciliation method to ensure coherence without any loss in forecasting performance by utilizing the node-dependent weighted optimization target to precisely control the adjustment scale for each node. We also provided a 'plug-and-play' optimization module that incorporates task-based targets and realistic constraints to solve real-world temporal HTS problems. We conducted extensive empirical evaluations on real-world datasets to validate our method, demonstrating the competitiveness of our approach compared with other state-of-the-art methods. Furthermore, our ablation studies verify the efficacy of each designed component. Our framework combining with the optimization module has been applied in the payment traffic management system in Alipay successfully.

References

Athanasopoulos, G.; Ahmed, R. A.; and Hyndman, R. J. 2009. Hierarchical forecasts for Australian domestic tourism. *International Journal of Forecasting*, 25(1): 146–166.

Athanasopoulos, G.; Hyndman, R. J.; Kourentzes, N.; and Petropoulos, F. 2017. Forecasting with temporal hierarchies. *European Journal of Operational Research*, 262(1): 60–74.

Bai, S.; Kolter, J. Z.; and Koltun, V. 2018. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. arXiv:1803.01271.

Chen, Z.; Ma, Q.; and Lin, Z. 2021. Time-Aware Multi-Scale RNNs for Time Series Modeling. In Zhou, Z.-H., ed., *International Joint Conference on Artificial Intelligence, IJCAI-21*, 2285–2291.

Chung, J.; Gulcehre, C.; Cho, K.; and Bengio, Y. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.

Das, A.; Kong, W.; Paria, B.; and Sen, R. 2022. A deep top-down approach to hierarchically coherent probabilistic forecasting.

Gardner, M. W.; and Dorling, S. 1998. Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences. *Atmospheric environment*, 32(14-15): 2627–2636.

Kourentzes, N.; and Athanasopoulos, G. 2019. Cross-temporal coherent forecasts for Australian tourism. *Annals of Tourism Research*, 75.

Laptev, N.; Yosinski, J.; Li, L. E.; and Smyl, S. 2017. Time-series Extreme Event Forecasting with Neural Networks at Uber. In *International Conference on Machine Learning*, 1–5. PMLR.

Li, Y.; Yu, R.; Shahabi, C.; and Liu, Y. 2018. Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting. In *International Conference on Learning Representations*.

Nocedal, J.; and Wright, S. J. 2006. *Numerical Optimization*. Springer New Yrok, NY.

Oreshkin, B. N.; Carpov, D.; Chapados, N.; and Bengio, Y. 2019. N-BEATS: Neural basis expansion analysis for interpretable time series forecasting. arXiv:1905.10437.

Rangapuram, S. S.; Kapoor, S.; Nirwan, R. S.; Mercado, P.; Januschowski, T.; Wang, Y.; and Bohlke-Schneider, M. 2023. Coherent Probabilistic Forecasting of Temporal Hierarchies. In Ruiz, F.; Dy, J.; and van de Meent, J.-W., eds., *International Conference on Artificial Intelligence and Statistics*, volume 206 of *Proceedings of Machine Learning Research*, 9362–9376. PMLR.

Rangapuram, S. S.; Werner, L. D.; Benidis, K.; Mercado, P.; Gasthaus, J.; and Januschowski, T. 2021. End-to-End Learning of Coherent Probabilistic Forecasts for Hierarchical Time Series. In *International Conference on Machine Learning*, 8832–8843. PMLR.

Rostami-Tabar, B.; Babai, M. Z.; Syntetos, A.; and Ducq3, Y. 2013. Demand Forecasting by Temporal Aggregation. *Naval Research Logistics*, 60(6): 479–498.

Salinas, D.; Flunkert, V.; Gasthaus, J.; and Januschowski, T. 2020. DeepAR: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting*, 36(3): 1181–1191.

Sutskever, I.; Vinyals, O.; and Le, Q. V. 2014. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27.

Taieb, S. B.; Taylor, J. W.; and Hyndman, R. J. 2021. Hierarchical probabilistic forecasting of electricity demand with smart meter data. *Journal of the American Statistical Association*, 116(533): 27–43.

Theodosiou, F.; and Kourentzes, N. 2021. Forecasting with deep temporal hierarchies. *Available at SSRN 3918315*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *Advances in neural information processing systems*, 5998–6008.

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

Zhou, F.; Pan, C.; Ma, L.; Liu, Y.; Wang, S.; Zhang, J.; Zhu, X.; Hu, X.; Hu, Y.; Zheng, Y.; Lei, L.; and Yun, H. 2023. SLOTH: Structured Learning and Task-Based Optimization for Time Series Forecasting on Hierarchies. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 11417–11425.