NHR-TFNet: Forecasting Hierarchical Time Series using Non-Linear Mappings

Shanika L Wickramasuriya[a*], Kasun Bandara[b], Hansika Hewamalage[c], Maneesha Perera[d]

[a] Department of Econometrics and Business Statistics, Monash University, Melbourne, Australia

[b] School of Computing and Information Systems, Melbourne Centre for Data Science, The University of Melbourne, Melbourne, Australia

[c] School of Computer Science & Engineering, University of New South Wales, Sydney, Australia

[d] Department of Mechanical Engineering, School of Electrical, Mechanical and Infrastructure Engineering, The University of Melbourne, Melbourne, Australia

Conceptualization: Shanika L Wickramasuriya, Kasun Bandara and Hansika Hewamalage

Methodology: Shanika L Wickramasuriya, Kasun Bandara and Hansika Hewamalage

Software: Shanika L Wickramasuriya, Kasun Bandara, Hansika Hewamalage and Maneesha Perera

Formal analysis: Shanika L Wickramasuriya, Kasun Bandara, Hansika Hewamalage and Maneesha Perera

Drafting the paper: Shanika L Wickramasuriya, Kasun Bandara, Hansika Hewamalage and Maneesha Perera

[*]Postal address: Department of Econometrics and Business Statistics, 900 Dandenong Road, Caulfield East, VIC 3145, Australia. E-mail address: shanika.wickramasuriya@monash.edu

NHR-TFNet: Forecasting Hierarchical Time Series using Non-linear Mappings

Abstract

Forecast reconciliation is a procedure of forecasting multivariate time series with linear aggregation constraints to ensure that forecasts satisfy the same set of constraints. Most state-of-the-art reconciliation methods reconcile the forecasts using linear mappings to project the base forecasts onto a coherent subspace. They also use assumptions such as the forecasts are unbiased and residuals are jointly covariance stationary. In this study, we propose a non-linear forecast reconciliation approach, NHR-TFNet, using machine learning methods, relaxing the assumptions of traditional linear reconciliation approaches. We introduce a novel loss function incorporating non-linear mappings to obtain a set of coherent forecasts from the individual base forecasts. To obtain the weights of the non-linear mapping between the base forecasts, we train a feed-forward neural network with the proposed loss function using transformed fitted values of the base models as inputs to the network. We evaluate the proposed methodology against state-of-the-art linear reconciliation approaches on multiple benchmark datasets. The results indicate that the proposed method works well when there is a reasonable number of observations in comparison to the number of bottom-level and non-bottom level series.

Keywords: Hierarchical time series, Coherent forecasts, Reconciliation, Non-linear mappings, Feed forward neural networks

1. Introduction

Nowadays, many operational forecasting tasks are associated with huge collections of time series spanning across several operational dimensions such as product categories, geographical areas, etc., which form a hierarchical structure. Often, the forecasts are collated at multiple levels for better decision-making and strategic planning from a business perspective. For instance, the product assortment structure in a retail business often follows a hierarchical structure, where it is required to produce sales forecasts at the product, store, state and country levels [4]. Also, in the electricity domain, generating energy demand forecasts at the smart-meter, grid, and regional levels is important for better demand planning and efficient resource management [27]. The prevalence of hierarchical forecasting problems is further demonstrated by recently held M5 forecasting competition [20] and Wikipedia Web Traffic forecasting competition [26], which were based on hierarchically structured time series data.

Hierarchical time series are often organised with several layers of granularity, where each level of the hierarchy can exhibit diverse patterns such as linear or nonlinear trends,

different seasonal shapes, level shifts, etc. For example, time series at the bottom level of the hierarchy are generally noisy and sparse, whereas time series at the higher aggregation levels of the hierarchy can show strong time series patterns (high signal-to-noise ratio) that are relatively easier to forecast. In a hierarchical forecasting setting, it is also required to generate forecasts that are coherent across the entire aggregation structure, which means the sum of the forecasts must be consistent with the aggregation structure of the hierarchy [30]. Therefore, generating accurate, but also coherent forecasts across the whole time series hierarchy is a challenging task.

The traditional approaches to forecasting hierarchical time series are mostly involved in generating forecasts for one level of aggregation and then either aggregated for higher levels (bottom-up approach), or disaggregated for lower levels (top-down approach), to obtain coherent forecasts across the hierarchy [30, 12]. As these methods produce forecasts considering only a single aggregation level, they are unable to exploit the information available across the rest of the levels in the hierarchy. Alternatively, methods have been introduced to combine forecasts across the entire hierarchy by assigning weights to each individual forecast in the hierarchy, where the weights are obtained through an optimisation process that produces coherent forecasts. Typically, such statistically solid hierarchical forecasting methods pursue a two-step process: 1) *the forecasting step* to produce base forecasts for each time series in the hierarchy; and 2) *the reconciliation step* to ensure the coherency of the forecasts across the hierarchy [11, 15, 30]. During the reconciliation process, these methods are benefited from utilising information across the hierarchy, thus often outperforming bottom-up and top-down approaches in the literature [30]. However, these methods use strong assumptions, e.g., model forecasts being unbiased, model residuals being jointly covariance stationary, and focus only on linear mappings to project the base forecasts onto the coherent subspace.

In parallel to these developments, with the advent of Big Data, many recent advancements in time series forecasting have been shifting to data-driven workhorses such as machine learning. In hierarchical settings, the non-parametric modelling capabilities of these models provide an opportunity to apply them as non-linear mapping functions to base forecasts, obviating the assumptions of traditional linear reconciliation techniques. Nevertheless, the advantages of using machine learning techniques as non-linear mapping functions for forecast reconciliation are yet to be translated into hierarchical forecasting research. Moreover, the time series forecasting techniques used to produce base forecasts have also evolved from traditional univariate models to global models that are trained across sets of many related time series [18]. Thus, the concept of sharing cross-series information can be applied to build competitive global models by clustering all the series in the hierarchy and applying a global model to each cluster allowing better control of the overall model complexity [3, 10].

In the recent literature, the incorporation of structural information for hierarchical forecasting has been explored [24, 21, 8] and has shown that exploiting hierarchical information in a non-linear manner leads to better accuracy than following a conventional linear reconciliation process. For example, [24] uses XGBoost and random forecast as non-linear models to non-linearly map the forecasts in the hierarchy to generate better forecasts at the bottom level of the hierarchy. Whereas [8] uses a regularized loss function to capture the non-linear

relationships among the time series of the hierarchy, which also maintains the forecast coherency of the hierarchy. Also, [21] proposes a framework that simultaneously learns from all the time series in the hierarchy to produce coherent, probabilistic forecasts, removing the additional post-processing reconciliation step used in the traditional reconciliation algorithms. However, the use of non-linear model along with a regularized loss function to minimise the error of the entire hierarchy and a generic hierarchical forecasting framework that can be used with both traditional univariate models and global models as base models has not yet been thoroughly studied. To this end, in this paper, we propose NHR-TFNet, a forecasting framework that attempts to account for the non-linear relationships that exist in hierarchical time series. The proposed NHR-TFNet framework follows a two-step procedure to produce coherent forecasts across the structure. Firstly, the base forecasts for each time series in the hierarchy are generated. NHR-TFNet framework allows both univariate and global models to be employed as base forecasting models. Secondly, to obtain the weights of the non-linear mapping between the base forecasts, we train a feed-forward multi-layer perceptron neural network (MLP) using the transformed fitted values of the base models as inputs to the network. In order to train the MLP, we introduce a novel loss function that accounts for the forecast loss and the loss associated with the coherency error, which acts as a penalty if the fit of the model deviates from the inherent hierarchical structure. As the base forecast models, we use univariate statistical forecasting techniques such as exponential smoothing methods (ETS) [14, 13] and autoregressive integrated moving average (ARIMA) models [6] from the `forecast` package [13] and a global model DeepAR [22] from the `GluonTS` package [1]. The NHR-TFNet framework is evaluated using multiple hierarchically structured time series databases, which contain various seasonal and trend patterns, exhibiting different time series characteristics. The source code relevant to NHR-TFNet framework is available at https://github.com/ManeeshaPerera/hierarchical-reconcilation-ML/

The rest of the paper is organised as follows. In Section 2, we formally define hierarchical time series and discuss the state-of-the-art reconciliation methods. Here, we will also introduce the proposed loss function that is used to train the MLP neural network. Next in Section 3, we explore the proposed NHR-TFNet framework in detail and highlight the key model development strategies used in our architecture. In Section 4, we summarise the benchmark datasets and explain the experimental setup used in this study to evaluate NHR-TFNet against other hierarchical reconciliation methods. Section 5 summarises the results obtained by NHR-TFNet on multiple hierarchical time series datasets and analyse the main observations of this study. Finally, Section 6 concludes the paper.

2. Problem Statement and Related work

This section mathematically defines hierarchical time series and briefly outlines widely used traditional forecast reconciliation approaches. We then introduce the loss function of the non-linear forecast reconciliation approach proposed in this paper which is then followed by a discussion of closely related relevant work in the literature.

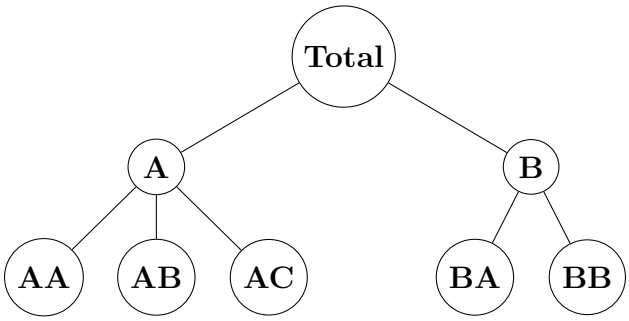

Figure 1: A two-level hierarchical structure.

2.1. Forecasting hierarchical time series

Let $\boldsymbol{b}_t \in \mathbb{R}^n$ be a vector of observations collected from the n series at the most disaggregated level of the structure at time t, $\boldsymbol{y}_t \in \mathbb{R}^m$ be a vector of all observations obtained by aggregating the series at the most disaggregated level at time t, for $t = 1, 2, \ldots, T$, where T is the length of the series. These vectors hold the following relationship:

$$\boldsymbol{y}_t = \boldsymbol{S}\boldsymbol{b}_t,$$

where $\boldsymbol{S}$ is often referred to as the *summing matrix* of order $m \times n$, which consists of a set of linear constraints present in the structure. To elaborate on these notations, let's consider the structure given in Figure 1. For this structure, $m = 8, n = 5, \boldsymbol{b}_t = [y_{AA,t}, y_{AB,t}, y_{AC,t}, y_{BA,t}, y_{BB,t}]^\top, \boldsymbol{y}_t = [y_t, y_{A,t}, y_{B,t}, \boldsymbol{b}_t^\top]^\top$, and

$$\boldsymbol{S} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ & & \boldsymbol{I}_5 & & \end{bmatrix},$$

where $\boldsymbol{I}_k$ denotes an identity matrix of order $k \times k$. In general, a structure with linear aggregation constraints can be decomposed as $\boldsymbol{S} = \begin{bmatrix} \boldsymbol{C} \\ \hdashline \boldsymbol{I}_n \end{bmatrix}$, where $\boldsymbol{C} \in \mathbb{R}^{m^* \times n}$, and $m^* = m - n$.

The point forecasts produced for all m series in the structure (also known as *base forecasts*) may not satisfy the aggregation constraints present in the data unless extremely simple forecasting methods are used. We denote $\hat{\boldsymbol{y}}_{T+h|T} \in \mathbb{R}^m$ as the vector consisting of h-steps-ahead *incoherent base forecasts* made using information available up to and including time T. To ensure that these forecasts satisfy the constraints, the base forecasts need to be adjusted and we called this process *forecast reconciliation*. It involves defining a mapping function ψ such that $\tilde{\boldsymbol{y}}_{T+h|T} = \psi(\hat{\boldsymbol{y}}_{T+h|T})$, where $\tilde{\boldsymbol{y}}_{T+h|T}$ satisfies the constraints and we refer to these as *reconciled forecasts*. The mapping function $\psi(\cdot)$ can be assumed to compose of two mappings $\psi(\hat{\boldsymbol{y}}_{T+h|T}) = s[g(\hat{\boldsymbol{y}}_{T+h|T})]$, where g combines the base forecasts to produce a new set of bottom level forecasts which are then aggregated by s.

When g is assumed to be linear, we can rewrite the mapping function to obtain the reconciled forecasts as $\tilde{\boldsymbol{y}}_{T+h|T} = \boldsymbol{S}\boldsymbol{G}\hat{\boldsymbol{y}}_{T+h|T}$, where $\boldsymbol{G}$ depends on the forecast reconciliation

approach. Many existing point forecast reconciliation approaches in the literature belong to this category and they are summarized in Table 1. Among them, the first five choices satisfy the constraints $\boldsymbol{GS} = \boldsymbol{I}$ which ensures that the reconciled forecasts are unbiased provided that the base forecasts are unbiased. In addition, the estimation of covariance matrices for MinT(Sample) and MinT(Shrink) assumes that the in-sample base forecast errors are jointly covariance stationary.

We could provide an alternative representation for the first five choices of the $\boldsymbol{G}$ matrix where we could interpret the reconciliation process as an adjustment done to the bottom-level base forecasts as

$$\boldsymbol{G}\hat{\boldsymbol{y}}_{T+h|T} = \hat{\boldsymbol{b}}_{T+h|T} - \boldsymbol{J}\boldsymbol{X}\boldsymbol{U}^{\top}\hat{\boldsymbol{y}}_{T+h|T},$$

where $\boldsymbol{X}$ depends on the forecast reconciliation approach and are given in Table 1, $\hat{\boldsymbol{b}}_{T+h|T}$ is the h-steps ahead base forecasts at the bottom level. and $\boldsymbol{U}^{\top} = \left[\boldsymbol{I}_n \vdots -\boldsymbol{C}\right]$. $\boldsymbol{U}^{\top}\hat{\boldsymbol{y}}_{T+h|T}$ quantifies the amount of aggregation inconsistency. For the hierarchy in Figure 1,

$$\boldsymbol{U}^{\top}\hat{\boldsymbol{y}}_{T+h|T} = \begin{bmatrix} \hat{y}_{T+h|T} - \hat{y}_{AA,T+h|T} - \hat{y}_{AB,T+h|T} - \hat{y}_{AC,T+h|T} - \hat{y}_{BA,T+h|T} - \hat{y}_{BB,T+h|T} \\ \hat{y}_{A,T+h|T} - \hat{y}_{AA,T+h|T} - \hat{y}_{AB,T+h|T} - \hat{y}_{AC,T+h|T} \\ \hat{y}_{B,T+h|T} - \hat{y}_{BA,T+h|T} - \hat{y}_{BB,T+h|T} \end{bmatrix}.$$

In theory, the matrix $\boldsymbol{U}$ is the null space of $\boldsymbol{S}^{\top}$ and the above choice is only one possibility. We have chosen this representation here because $\boldsymbol{U}^{\top}\hat{\boldsymbol{y}}_{T+h|T}$ is interpretable as the amount of aggregation inconsistency of the bottom-level forecasts.

Table 1: Traditional choices of the $\boldsymbol{G}$ matrix.

Reconciliation method	$\boldsymbol{G}$	$\boldsymbol{X}$
BU (bottom-up)	$\left[\boldsymbol{0}_{n\times m^*} \vdots \boldsymbol{I}_n\right]$	$\boldsymbol{0}$
OLS (ordinary least squares) [11]	$(\boldsymbol{S}^{\top}\boldsymbol{S})^{-1}\boldsymbol{S}^{\top}$	$\boldsymbol{U}(\boldsymbol{U}^{\top}\boldsymbol{U})^{-1}$
WLS (weighted least squares) [15]	$(\boldsymbol{S}^{\top}\hat{\boldsymbol{\Lambda}}_1^{-1}\boldsymbol{S})^{-1}\boldsymbol{S}^{\top}\hat{\boldsymbol{\Lambda}}_1^{-1}$	$\hat{\boldsymbol{\Lambda}}_1\boldsymbol{U}(\boldsymbol{U}^{\top}\hat{\boldsymbol{\Lambda}}_1\boldsymbol{U})^{-1}$
MinT(Sample) [30]	$(\boldsymbol{S}^{\top}\hat{\boldsymbol{W}}_{1,\text{sam}}^{-1}\boldsymbol{S})^{-1}\boldsymbol{S}^{\top}\hat{\boldsymbol{W}}_{1,\text{sam}}^{-1}$	$\hat{\boldsymbol{W}}_{1,\text{sam}}\boldsymbol{U}(\boldsymbol{U}^{\top}\hat{\boldsymbol{W}}_{1,\text{sam}}\boldsymbol{U})^{-1}$
MinT(Shrink) [30]	$(\boldsymbol{S}^{\top}\hat{\boldsymbol{W}}_{1,\text{shr}}^{-1}\boldsymbol{S})^{-1}\boldsymbol{S}^{\top}\hat{\boldsymbol{W}}_{1,\text{shr}}^{-1}$	$\hat{\boldsymbol{W}}_{1,\text{shr}}\boldsymbol{U}(\boldsymbol{U}^{\top}\hat{\boldsymbol{W}}_{1,\text{shr}}\boldsymbol{U})^{-1}$
EMinT-U [28]	$\boldsymbol{B}_h^{\top}\hat{\boldsymbol{Y}}_h(\hat{\boldsymbol{Y}}_h^{\top}\hat{\boldsymbol{Y}}_h)^{-1}$	

$\hat{\boldsymbol{W}}_{1,\text{sam}}$ and $\hat{\boldsymbol{W}}_{1,\text{shr}}$ are the sample, and shrinkage covariance matrix, respectively of 1-step-ahead in-sample base forecast errors. $\hat{\boldsymbol{\Lambda}}_1 = \text{diag}(\hat{\boldsymbol{W}}_{1,\text{sam}})$, $\boldsymbol{Y}_h = [\boldsymbol{y}_h, \boldsymbol{y}_{h+1}, \dots, \boldsymbol{y}_T]^{\top} \in \mathbb{R}^{(T-h+1)\times m}$, $\boldsymbol{B}_h = [\boldsymbol{b}_h, \boldsymbol{b}_{h+1}, \dots, \boldsymbol{b}_T]^{\top} \in \mathbb{R}^{(T-h+1)\times n}$, $\hat{\boldsymbol{Y}}_h = [\hat{\boldsymbol{y}}_{h|0}, \hat{\boldsymbol{y}}_{h+1|1}, \dots, \hat{\boldsymbol{y}}_{T|T-h}]^{\top} \in \mathbb{R}^{(T-h+1)\times m}$. $\text{diag}(\boldsymbol{A})$ constructs a diagonal matrix using the diagonal elements of the square matrix $\boldsymbol{A}$.

2.2. Proposed loss function

Motivated by this second interpretation of forecast reconciliation, we consider a non-linear mapping for g and are interested in minimise the following loss function:

$$\min_{\boldsymbol{\theta}} \frac{1}{T} \left[\sum_{t=1}^{T} \left\| \boldsymbol{b}_t - \tilde{\boldsymbol{b}}_{t|t-1} \right\|_2^2 + \lambda \left\| \boldsymbol{a}_t - \boldsymbol{C}\tilde{\boldsymbol{b}}_{t|t-1} \right\|_2^2 \right], \tag{1}$$

where $\tilde{\boldsymbol{b}}_{t|t-1} = \hat{\boldsymbol{b}}_{t|t-1} + \boldsymbol{g}\left(\boldsymbol{U}^{\top}\hat{\boldsymbol{y}}_{t|t-1}, \boldsymbol{\theta}\right)$, $\lambda > 0$ is the penalty parameter, $\boldsymbol{g}\left(\cdot, \boldsymbol{\theta}\right) = [g_1(\cdot, \boldsymbol{\theta}_1), g_2(\cdot, \boldsymbol{\theta}_2), \dots g_n(\cdot,$ $g_j(\cdot, \boldsymbol{\theta}_j)$ is a non-linear mapping function with parameter vector $\boldsymbol{\theta}_j$ for $j = 1, 2, \dots, n$, $\boldsymbol{\theta} = \left[\boldsymbol{\theta}_1^{\top}, \boldsymbol{\theta}_2^{\top}, \dots, \boldsymbol{\theta}_n^{\top}\right]^{\top}$, $\boldsymbol{y}_t = \left[\boldsymbol{a}_t^{\top}, \boldsymbol{b}_t^{\top}\right]^{\top}$, and $\| \cdot \|_2$ is the l_2-norm. The intuition behind this objective function is to non-linearly map a given set of base forecasts so that the mean squared forecast error of the bottom level is minimized while incorporating a penalty for not being able to minimize the mean squared forecast error of the aggregated series. We could rearrange the terms of the loss function and rewrite it as

$$\min_{\boldsymbol{\theta}} \frac{1}{T} \left[\sum_{t=1}^{T} \left\| \boldsymbol{\Lambda}^{1/2}(\boldsymbol{y}_t - \boldsymbol{S}\tilde{\boldsymbol{b}}_{t|t-1}) \right\|_2^2 \right], \tag{2}$$

where $\boldsymbol{\Lambda} = \begin{bmatrix} \lambda\boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{I} \end{bmatrix}$ and it can be viewed as a nonlinear weighted least squares approach where the weights are given by $\boldsymbol{\Lambda}$. It is interesting to note that MinT(Sample) can be viewed as a special case of this approach. If restricted to linear mappings with no hidden layers in the MLP network and include conditions for unbiasedness, then the solution reduces to MinT(Sample) for any choice of λ.

After estimating the best non-linear mapping by minimizing the in-sample 1-step-ahead forecast errors as shown in Eq. (1), we obtain the coherent forecasts for the structure as

$$\tilde{\boldsymbol{y}}_{T+h|T} = \boldsymbol{S}\left[\hat{\boldsymbol{b}}_{T+h|T} + \boldsymbol{g}\left(\boldsymbol{U}^{\top}\hat{\boldsymbol{y}}_{T+h|T}, \hat{\boldsymbol{\theta}}\right)\right],$$

where $\hat{\boldsymbol{\theta}}$ is the solution of Eq. (1). For simplicity, we could use the same non-linear mapping for each forecast horizon. Alternatively, we could modify the proposed objective function to accommodate h-step-ahead in-sample forecast errors which leads to different non-linear mappings for each h.

2.3. Related literature

Burba and Chen [7] introduced a non-linear approach based on an encoder-decoder neural network, where the encoder is a trainable feed-forward neural network that takes base forecasts as the input and outputs the bottom-level reconciled forecasts, and the decoder is the summing matrix which obtains forecasts for all the levels using encoded bottom level forecasts. They have evaluated the performance of this approach using two loss functions: mean absolute scaled error and mean logarithm of absolute error and showed superior performances over traditional approaches when the hierarchies are complex. Shiratori et al. [23]

considered a loss function similar to Eq. (1) but it was used to estimate a prediction model for the bottom-level series while incorporating upper-level forecasts through the penalty function. They have used a different penalty parameter for each level in the hierarchy which can be computationally costly for large structures. Using 2-level hierarchies constructed from synthetic and real data, they have demonstrated that their method yields comparable performances to BU and MinT(Shrink). Spiliotis et al. [24] proposed a non-linear reconciliation approach based on decision tree-based machine learning models: XGBoost and random forest. Their approach models each of the bottom-level series separately on 1-step ahead base forecasts obtained for each series in the structure for a holdout set. The loss function used in this study is interested in minimizing the mean squared combination forecast error for each bottom-level series and has not taken reconciliation error into account.

3. NHR-TFNet Framework

In this section, we describe the main components of the NHR-TFNet framework. NHR-TFNet is composed of three components, namely: 1) the base-forecasting layer, 2) the MLP training layer, and 3) the post-processing layer. In the following, we first discuss the base-forecasting layer that generates base forecasts for each time series in the hierarchy and provide fitted values of the base models. Next, we explain the MLP training layer, which is the main learning component of NHR-TFNet. Then, we discuss the post-processing layer of NHR-TFNet and describe the hyper-parameter selection method used in NHR-TFNet. Finally, we demonstrate the forecast reconciliation stage of this trained network.

3.1. Base-forecasting layer

The base-forecasting layer produces base forecasts and model fits for each time series in the hierarchy. In our experiments, we use both univariate and global models to generate base forecasts. As the univariate models, we apply ETS and ARIMA implementations from the `forecast` package [13], whereas for the global model, we use DeepAR [22] implementation from the `GluonTS` package [1]. As global models are trained across a set of many time series, prior to applying the DeepAR model, we first perform k-means clustering on the set of time series features extracted from all the time series in the hierarchy. We use the `tsfeatures` package developed by Hyndman et al. [16] to extract features from each time series. Based on the time series features, we next perform the k-means clustering with a cluster size of 20 (i.e., the number of clusters). Although we have fixed the number of clusters here to reduce the complexity of finding the optimal number of clusters and having a large number of possible clusters, we can apply techniques such as the Elbow method to determine the optimal number of clusters.

3.2. MLP training layer

For adjusting the base forecasts according to the aggregation constraints in the hierarchy, we use an MLP based model. Neural Network based models have the support for producing multiple outputs as a function of the same set of inputs. At time step t, the inputs $\bar{\boldsymbol{x}}_{t|t-1}$ to our network are the transformed fitted values from the base forecasting method. Given

a simple hierarchical structure with two levels as shown in Figure 1, the transformation of the fitted values can be done as per Eq. (3). In Eq. (3), $\bar{y}_{t|t-1}, \bar{y}_{A,t|t-1}$ and $\bar{y}_{B,t|t-1}$ for $t = 1, 2, \ldots, T$ refer to the transformed fitted values. If the residuals of the base forecasting method are jointly covariance stationary, considering the transformed fitted values in this way allows us to feed reasonably stationary data as inputs to the MLP model. The corresponding outputs of the network $\tilde{\boldsymbol{b}}_{t|t-1}$ are the adjusted fitted values at the bottom-level as shown in Eq. (4).

$$
\begin{aligned}
\bar{y}_{t|t-1} &= \hat{y}_{t|t-1} - \left[\hat{y}_{AA,t|t-1} + \hat{y}_{AB,t|t-1} + \hat{y}_{AC,t|t-1} + \hat{y}_{BA,t|t-1} + \hat{y}_{BB,t|t-1}\right] \\
\bar{y}_{A,t|t-1} &= \hat{y}_{A,t|t-1} - \left[\hat{y}_{AA,t|t-1} + \hat{y}_{AB,t|t-1} + \hat{y}_{AC,t|t-1}\right] \\
\bar{y}_{B,t|t-1} &= \hat{y}_{B,t|t-1} - \left[\hat{y}_{BA,t|t-1} + \hat{y}_{BB,t|t-1}\right] \\
\bar{\boldsymbol{x}}_{t|t-1} &= \left[\bar{y}_{t|t-1}, \bar{y}_{A,t|t-1}, \bar{y}_{B,t|t-1}, \hat{y}_{AA,t|t-1}, \hat{y}_{AB,t|t-1}, \hat{y}_{AC,t|t-1}, \hat{y}_{BA,t|t-1}, \hat{y}_{BB,t|t-1}\right]^{\top} \qquad (3) \\
\tilde{\boldsymbol{b}}_{t|t-1} &= \left[\tilde{y}_{AA,t|t-1}, \tilde{y}_{AB,t|t-1}, \tilde{y}_{AC,t|t-1}, \tilde{y}_{BA,t|t-1}, \tilde{y}_{BB,t|t-1}\right]^{\top} \qquad (4)
\end{aligned}
$$

Figure 2 shows the overall architecture of the MLP model used for the reconciliation in this work. The model contains a number of stacks each of which consists of a dense layer, a batch normalisation layer, rectified linear unit (ReLU) activation followed by a dropout layer. The batch normalisation step helps to speed up the neural network model training by standardizing the inputs to layers of the network while also handling internal covariate shift, which is the change of data distribution from one layer of the network to the next [17]. Also, both batch normalisation and the dropout layer help to reduce overfitting in the network. The dropout layer randomly drops nodes from the dense layers during the model training to ensure model generalisation and to reduce node inter-dependence [25]. A final dense layer, placed after the final stack of the network, having the same size as the number of time series in the bottom-most level of the hierarchy, reshapes the model outputs to the required final output shape.

The MLP model also uses a skip connection to the outputs which directly adds up the original fitted values of the bottom-level series to the outputs of the MLP model. Hence, the skip connection helps the model to learn only the adjustment that needs to be performed on top of the fitted values of the bottom-level series provided by the base forecasting method. Theoretically, this makes the learning process of the MLP model easier [9]. On top of the aforementioned input transformation, inputs are further transformed using min-max normalisation as well.

3.3. Post-processing layer

The outputs of the model, which are the adjusted fitted values of the bottom-most level in the hierarchy, are in the original scale. Once these values are retrieved, they are hierarchically aggregated in a bottom-up fashion to construct the values for all the levels of aggregation.

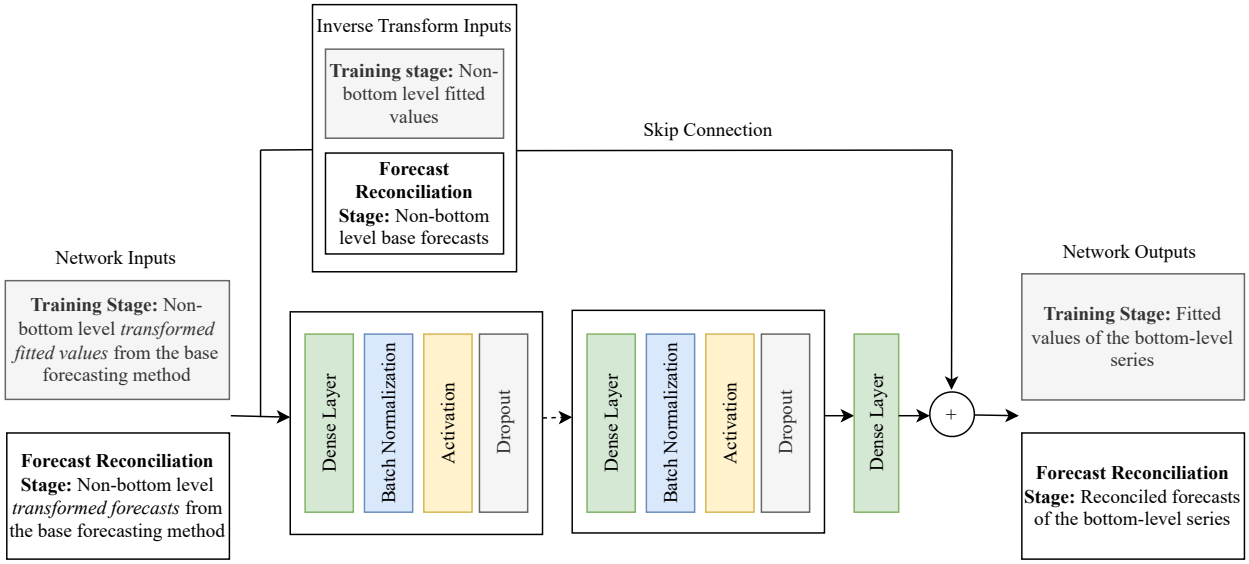

Figure 2: An overview of the proposed architecture for the NHR-TFNet framework, which consists of a base-forecasting layer, MLP training layer, and a post-processing layer.

Table 2: Minimum and maximum values for each hyper-parameter.

Hyper-parameter	Minimum	Maximum
Number of layers	1	5
Number of epochs	10	200
Dropout rate	0	0.5
Max normalization constraint	0	10
Learning rate	0.0001	0.1
Batch size	1	size of input data
λ (proposed loss function)	0.01	5

3.4. Hyper-parameter automation and optimisation

Several hyper-parameters need to be tuned for the MLP model. Table 2 shows the hyper-parameters, minimum and maximum values we use for each parameter. To find the best values for the hyper-parameters we use the Bayesian optimization algorithm [5] from `hyperopt` Python package. Once the optimal parameter values are found, we train the MLP using the optimal parameters five times with different seeds to account for the random weight initialization of the network. The average across the adjusted bottom-level fitted values from the five iterations is taken as the final adjusted bottom-level fitted values.

3.5. Forecast reconciliation stage

Once the network is trained, the base forecasts and transformed base forecasts (by following similar equations as transformed fitted values) are fed into the network to compute the reconciled forecasts at the bottom level. These forecasts are then summed appropriately to obtain the forecasts for other series in the structure.

Table 3: Description of the datasets.

Dataset	Frequency	Total number of Levels	Series	Bottom-level series	Non-bottom level series
Prison	4 (quarterly)	5	121	64	57
Labour	4 (quarterly)	4	57	32	25
Tourism	12 (monthly)	3	85	77	8
Wikipedia	7 (weekly)	6	1095	913	182

4. Experiments

4.1. Datasets

We use four state-of-the-art hierarchical time series datasets for evaluation. We first pre-process the datasets using the `tsclean` function from the `forecast` package [13] to remove any outliers that maybe present in the data. Figure 3 shows the time series datasets after pre-processing. Table 3 shows the frequency of the time series, the number of levels, the total number of time series in the hierarchy, bottom-level and non-bottom level for the four datasets. We can observe that these datasets show varying levels of non-linear trends and seasonality. For the Wikipedia dataset, we can observe that some series at the bottom level are intermittent.

Prison Quarterly prison population in Australia from Q1 2005 to Q4 2016 [12]. The levels of the hierarchy are Australia (total), state, gender, legal status and indigenous status.

Labour Quarterly employed individuals in Australia from Q1 1987 to Q4 2018 [2]. The levels of the hierarchy are Australia (total), occupation category, employment status and gender.

Tourism Monthly domestic visitor nights in Australia from January 1998 to December 2019 [29]. The levels of the hierarchy are Australia (total), state and region.

Wikipedia Daily pageviews for the most popular social network articles on Wikipedia from 01-01-2016 to 29-06-2017 [19]. The levels of the hierarchy are access, agent, language, purpose and article.

4.2. Evaluation

4.2.1. Benchmark Methods

We evaluate the proposed NHR-TFNet with six state-of-the-art hierarchical forecasting benchmarks: BU, OLS, WLS, MinT(Sample), MinT(Shrink) and EMinT-U. For the prison and Wikipedia datasets, MinT(Sample) cannot be calculated as the number of time series in the hierarchy is greater than the number of observations. Therefore the comparisons for these two datasets are done using the other five benchmarks.

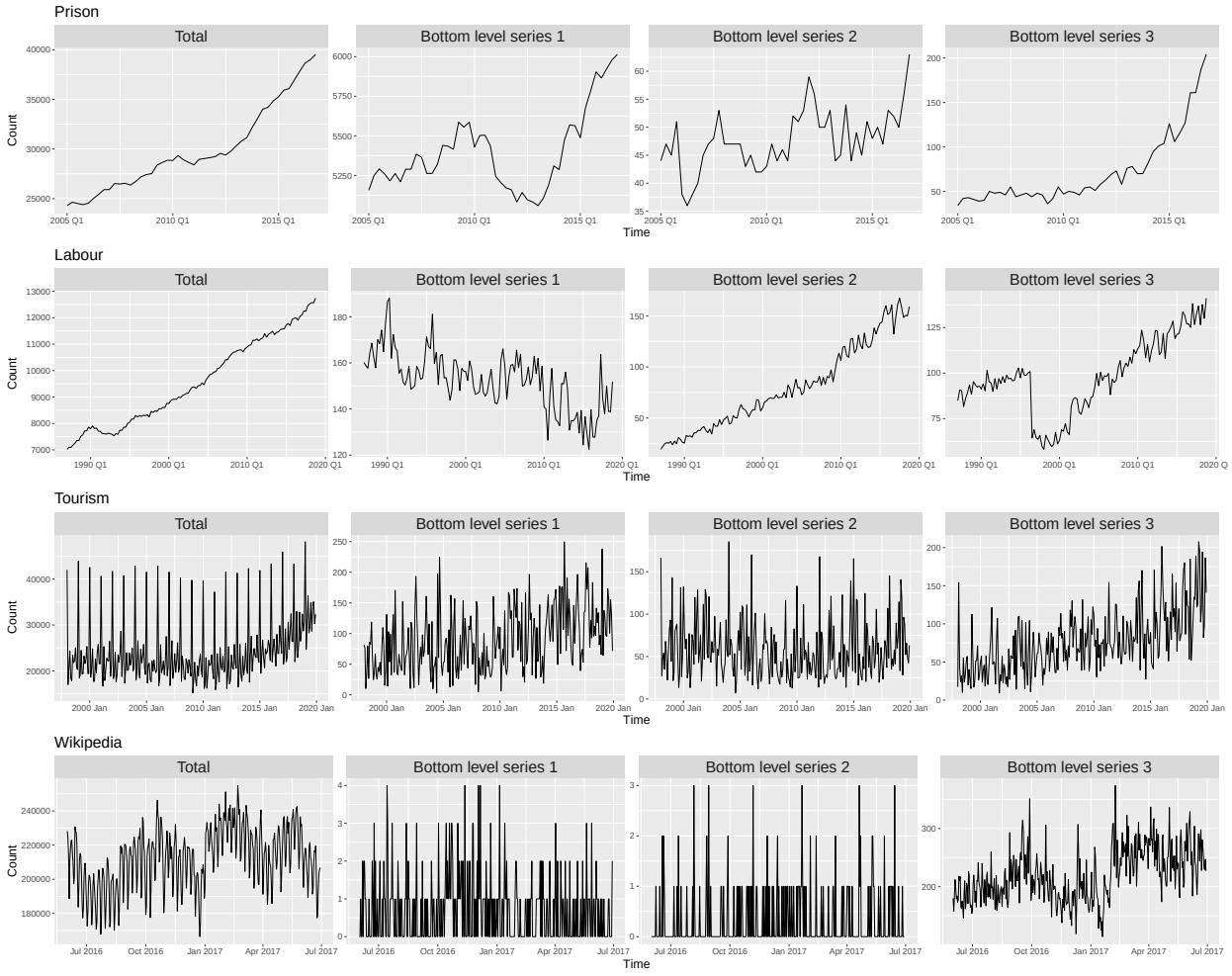

Figure 3: A few selected series at the top and bottom level of the hierarchy for each dataset used in this study.

4.2.2. Rolling Origin Evaluation

In this work to evaluate the forecasting performance, we conduct a rolling origin evaluation where the forecasting origin is updated at each rolling window and the forecasts are produced from each origin. An example of this evaluation is illustrated in Figure 4. To start the rolling origin evaluation, for each dataset we pick a reasonable training window size considering the trade-off between the number of rolling windows and the number of observations available to train the base forecasting methods and derive the fitted values to train the proposed MLP model. Table 4 shows the number of rolling windows and the number of observations available for the first rolling window (i.e., training window size of the first rolling window).

For each rolling window, we fit the base forecasting methods (ARIMA, ETS, or DeepAR) and compute the fitted values and 1-step-ahead forecasts. We next apply the hierarchical forecasting benchmarks and the proposed MLP using the fitted values and forecasts for a

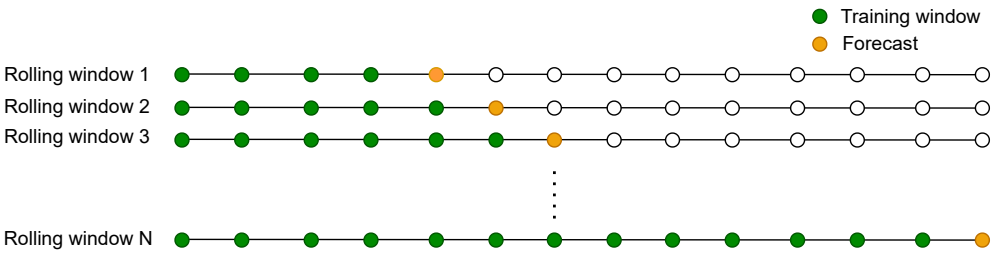

Figure 4: An illustration of the rolling origin evaluation.

Table 4: Number of rolling windows for each dataset and observations present for training in the first rolling window.

Dataset	Number of rolling windows	Min. window size
Prison	24	24
Labour	60	68
Tourism	120	144
Wikipedia	70	324

given rolling window. The same steps are conducted for all rolling windows of a dataset. For each rolling window, the mean squared error (MSE) is calculated and the average across series in different levels and average across all the series in the structure for a given dataset is obtained. Finally, the percentage relative improvement in average loss (PRIAL) is calculated as shown in Eq. (5) for each level in the structure and for the entire hierarchy (i.e., overall) using the average MSEs of the forecasts from the base forecasting methods (i.e., base forecasts) and the average MSEs after applying the reconciliation methods (i.e., reconciled forecasts). A positive value of PRIAL indicates the accuracy of the reconciled forecasts has increased while a negative value indicates the accuracy has decreased.

$$\text{PRIAL} = \frac{\text{MSE(base forecasts)} - \text{MSE(reconciled forecasts)}}{\text{MSE(base forecasts)}} \times 100\%. \tag{5}$$

5. Discussion

Table 5 reports the forecasting performance of the various reconciliation methods for the four datasets. The row labeled *Base MSE* reports the MSEs of the base forecasting methods and *Rank* reports the ranking of the reconciliation methods where rank one is given to the best-performing method whereas rank 6 or 7 is given to the least performing method. For the Prison, Labour and Tourism datasets, we set the λ value to take [0.1, 0.9] whereas for the Wikipedia dataset, we set a slightly higher range i.e., [0.01, 5] because some series at the bottom-level are intermittent so it may be better to give a high importance to the non-bottom level series.

According to Table 5, we observe that the proposed NHR-TFNet framework obtains the best PRIAL rank on the Prison, Labour, and Tourism datasets when DeepAR, ETS, and

Table 5: Average MSE of the base forecasts and PRIAL from reconciliation methods for the four datasets.

Prison		BU	OLS	WLS	MinT(Sample)	MinT(Shrink)	EMinT-U	NHR-TFNet
					ARIMA			
	Base MSE ($\times 10^3$)							2.8
	PRIAL	−48.5	7.5	−2.3		−0.2	−83.5	−3.1
	Rank	*5*	*1*	*3*		*2*	*6*	*4*
					ETS			
	Base MSE ($\times 10^3$)							3.0
	PRIAL	−46.6	6.8	3.8		10.3	−139.2	−45.5
	Rank	*5*	*2*	*3*		*1*	*6*	*4*
					DeepAR			
	Base MSE ($\times 10^3$)							4.3
	PRIAL	23.6	4.7	21.9		23.0	−32.4	24.1
	Rank	*2*	*5*	*4*		*3*	*6*	*1*
Labour		BU	OLS	WLS	MinT(Sample)	MinT(Shrink)	EMinT-U	NHR-TFNet
					ARIMA			
	Base MSE ($\times 10^2$)							3.9
	PRIAL	−6.8	3.8	2.2	−33.1	5.8	−141.6	−2.7
	Rank	*5*	*2*	*3*	*6*	*1*	*7*	*4*
					ETS			
	Base MSE ($\times 10^2$)							3.8
	PRIAL	−2.8	2.2	2.0	−30.3	3.2	−198.1	3.5
	Rank	*5*	*3*	*4*	*6*	*2*	*7*	*1*
					DeepAR			
	Base MSE ($\times 10^2$)							5.3
	PRIAL	15.0	5.0	12.4	−1.3	17.0	−116.9	13.4
	Rank	*2*	*5*	*4*	*6*	*1*	*7*	*3*
Tourism		BU	OLS	WLS	MinT(Sample)	MinT(Shrink)	EMinT-U	NHR-TFNet
					ARIMA			
	Base MSE ($\times 10^4$)							5.6
	PRIAL	−71.7	2.5	−27.7	0.3	−11.9	−35.9	−3.1
	Rank	*7*	*1*	*5*	*2*	*4*	*6*	*3*
					ETS			
	Base MSE ($\times 10^4$)							5.3
	PRIAL	−38.7	0.8	−17.7	2.1	−13.1	−116.8	1.0
	Rank	*6*	*3*	*5*	*1*	*4*	*7*	*2*
					DeepAR			
	Base MSE ($\times 10^4$)							11.8
	PRIAL	−23.2	1.8	−9.8	16.1	12.6	−6.5	23.0
	Rank	*7*	*4*	*6*	*2*	*3*	*5*	*1*
Wikipedia		BU	OLS	WLS	MinT(Sample)	MinT(Shrink)	EMinT-U	NHR-TFNet
					ARIMA			
	Base MSE ($\times 10^5$)							1.6
	PRIAL	4.4	2.1	8.8		13.4	−294.3	6.5
	Rank	*4*	*5*	*2*		*1*	*6*	*3*
					ETS			
	Base MSE ($\times 10^5$)							1.2
	PRIAL	1.6	0.6	2.4		2.7	−604.4	−5.2
	Rank	*3*	*4*	*2*		*1*	*6*	*5*
					DeepAR			
	Base MSE ($\times 10^5$)							5.0
	PRIAL	19.9	3.3	19.9		47.4	−49.0	15.5
	Rank	*3*	*5*	*2*		*1*	*6*	*4*

DeepAR models are used as the respective base forecasting models. For the Prison, Labour and Wikipedia datasets, in general, the best-performing methods use a linear mapping function such as MinT(Shrink). Notably, the proposed NHR-TFNet framework performs among the top four reconciliation methods. This observation can be attributed to these three datasets having a limited number of observations in comparison to the number of bottom-level and non-bottom level series so learning a less complex linear mapping function is beneficial. For the Tourism dataset, MinT(Sample) and NHR-TFNet show competitive performances as this dataset has a reasonable number of observations in comparison to the number of bottom-level and non-bottom level series. For all the datasets, the worst-performing method is EMinT-U. This result is expected as this method works well when the series are jointly covariance stationary which is not valid for these datasets. With respect to base forecast models, we see that MSE is highest in DeepAR. This result is not surprising given that all the datasets have a limited number of time series, thus learning a global model such as DeepAR can be challenging in practice as they require a sufficient amount of time series to estimate their numerous model parameters.

Table 6 summarizes the performance across various levels in the hierarchy for the Prison dataset. The bold entries highlight the best-performing method and the underline numbers highlight the second-best method. On average, for ARIMA base forecasts, OLS performs the best, and MinT(Shrink) is the second best in most of the levels. For ETS and DeepAR base forecasts, MinT(Shrink) is the best in most of the levels except the top-level. This observed pattern is aligned with the rankings noted in Table 5. To save space we report PRIAL for each level in Appendix A for the remaining datasets.

6. Conclusions and Future Work

Widely used traditional methods for forecast reconciliation use a linear function to map all the base forecasts into the bottom-level series which are then added up by a summing matrix to produce coherent forecasts for the whole hierarchy. In this paper, we explored the forecasting performance when a non-linear mapping function is used instead. We trained a feed-forward neural network with a skip connection where an adjustment estimated using transformed fitted values is added to the original base forecasts at the bottom-level series which are then summed to produce forecasts for the upper-level series. We evaluated the proposed methodology using four benchmark datasets. The results indicated that NHR-TFNet showed competitive results for the datasets having a reasonable number of observations compared to the number of bottom-level and non-bottom level series.

An important perspective of this study is to explore a slightly modified version of the proposed loss function. It attempts to find a non-linear mapping by minimizing the in-sample 1-step-ahead mean squared error of the whole structure while penalizing for incoherence in the forecasts. However, this approach will produce near coherent forecasts, and the deviation from coherence depends on the tuning parameter.

Table 6: Average MSEs of base forecasts and PRIAL from reconciliation methods for the Prison dataset.

Method	Australia	State	Gender	Legal	Indigenous
			ARIMA		
BU	−125.3	−46.6	−37.4	−4.4	0.0
OLS	**5.7**	**3.3**	**4.0**	**13.2**	11.2
WLS	−26.0	−3.5	2.8	11.0	13.3
MinT(Shrink)	−19.7	−2.6	2.3	12.1	**14.5**
EMinT-U	−101.4	−79.1	−79.4	−66.8	−85.9
NHR-TFNet	−9.1	−9.3	−4.5	3.9	5.4
Base MSE ($\times 10^3$)	*88.1*	*7.9*	*3.5*	*2.3*	*0.9*
			ETS		
BU	−105.8	−40.6	−30.4	−6.4	0.0
OLS	4.7	4.5	8.1	9.0	9.8
WLS	−4.7	3.0	9.3	9.3	9.6
MinT(Shrink)	**6.5**	**8.2**	**13.3**	**13.4**	**13.7**
EMinT-U	−154.6	−122.4	−133.1	−140.2	−133.5
NHR-TFNet	−83.5	−43.0	−30.8	−25.1	−11.6
Base MSE ($\times 10^3$)	*113.7*	*8.6*	*3.8*	*2.2*	*0.8*
			DeepAR		
BU	43.1	12.5	9.0	0.9	0.0
OLS	11.4	−1.3	0.6	−2.4	−2.2
WLS	39.1	11.6	9.4	2.2	1.1
MinT(Shrink)	39.1	**11.8**	**9.8**	**6.4**	**5.9**
EMinT-U	−9.4	−34.8	−45.2	−63.9	−77.9
NHR-TFNet	**46.9**	8.7	5.7	0.0	−0.4
Base MSE ($\times 10^3$)	*241.9*	*10.7*	*4.4*	*2.1*	*0.8*

7. Acknowledgment

This research was funded by an IIF-SAS research award and by an Infrastructure Data Science research award at Meta. The research was undertaken using the LIEF HPC-GPGPU Facility hosted at the University of Melbourne. This Facility was established with the assistance of LIEF Grant LE170100200

References

[1] Alexandrov, A., Benidis, K., Bohlke-Schneider, M., Flunkert, V., Gasthaus, J., Januschowski, T., Maddix, D.C., Rangapuram, S., Salinas, D., Schulz, J., Stella, L., Türkmen, A.C., Wang, Y., 2020. GluonTS: Probabilistic and neural time series modeling in Python. Journal of Machine Learning Research 21, 1–6. URL: http://jmlr.org/papers/v21/19-820.html.

[2] Australian Bureau of Statistics, 2022. Employed persons by age, occupation sub-major group of main job (ANZSCO) and sex. Accessed on 12-06-2022.

[3] Bandara, K., Bergmeir, C., Smyl, S., 2020. Forecasting across time series databases using recurrent neural networks on groups of similar series: A clustering approach. Expert Systems with Applications 140, 112896.

[4] Bandara, K., Hewamalage, H., Godahewa, R., Gamakumara, P., 2022. A fast and scalable ensemble of global models with long memory and data partitioning for the M5 forecasting competition. International Journal of Forecasting 38, 1400–1404.

[5] Bergstra, J., Yamins, D., Cox, D., 2013. Making a science of model search: hyperparameter optimization in hundreds of dimensions for vision architectures, in: International Conference on Machine Learning, PMLR. pp. 115–123. URL: http://hyperopt.github.io/hyperopt.

[6] Box, G.E.P., Jenkins, G.M., Reinsel, G.C., Ljung, G.M., 2015. Time Series Analysis: Forecasting and Control. John Wiley & Sons.

[7] Burba, D., Chen, T., 2021. A trainable reconciliation method for hierarchical time-series. arXiv:2101.01329.

[8] Han, X., Dasgupta, S., Ghosh, J., 2021. Simultaneously reconciled quantile forecasting of hierarchically related time series, in: Proceedings of the 24th International Conference on Artificial Intelligence and Statistics (AISTATS), San Diego, CA, USA.

[9] He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770–778. doi:10.1109/CVPR.2016.90.

[10] Hewamalage, H., Bergmeir, C., Bandara, K., 2021. Recurrent neural networks for time series forecasting: current status and future directions. International Journal of Forecasting 37, 388–427.

[11] Hyndman, R.J., Ahmed, R.A., Athanasopoulos, G., Shang, H.L., 2011. Optimal combination forecasts for hierarchical time series. Computational Statistics & Data Analysis 55, 2579–2589.

[12] Hyndman, R.J., Athanasopoulos, G., 2021. Forecasting: Principles and Practice. OTexts: Melbourne, Australia. OTexts.com/fpp3. Accessed on 12-06-2022.

[13] Hyndman, R.J., Khandakar, Y., 2008. Automatic time series forecasting: the forecast package for R. Journal of Statistical Software 26, 1–22.

[14] Hyndman, R.J., Koehler, A.B., Ord, K.J., Snyder, R.D., 2008. Forecasting with Exponential Smoothing: The State Space Approach. Springer Science & Business Media.

[15] Hyndman, R.J., Lee, A.J., Wang, E., 2016. Fast computation of reconciled forecasts for hierarchical and grouped time series. Computational Statistics & Data Analysis 97, 16–32.

[16] Hyndman, R.J., Wang, E., Kang, Y., Talagala, T., 2021. tsfeatures: Time series feature extraction. URL: https://github.com/robjhyndman/tsfeatures/. r package version 0.1.

[17] Ioffe, S., Szegedy, C., 2015. Batch normalization: accelerating deep network training by reducing internal covariate shift, in: Proceedings of the 32nd International Conference on International Conference on Machine Learning, JMLR.org. pp. 448–456.

[18] Januschowski, T., Gasthaus, J., Wang, Y., Salinas, D., Flunkert, V., Bohlke-Schneider, M., Callot, L., 2020. Criteria for classifying forecasting methods. International Journal of Forecasting 36, 167–177.

[19] Mahsa Ashouri, R.J.H., Shmueli, G., 2022. Fast forecast reconciliation using linear models. Journal of Computational and Graphical Statistics 31, 263–282. doi:10.1080/10618600.2021.1939038.

[20] Makridakis, S., Spiliotis, E., Assimakopoulos, V., 2022. M5 accuracy competition: results, findings, and conclusions. International Journal of Forecasting 38, 1346–1364.

[21] Rangapuram, S.S., Werner, L.D., Benidis, K., Mercado, P., Gasthaus, J., Januschowski, T., 2021. End-to-End learning of coherent probabilistic forecasts for hierarchical time series, in: Meila, M., Zhang, T. (Eds.), Proceedings of the 38th International Conference on Machine Learning, PMLR. pp. 8832–8843.

[22] Salinas, D., Flunkert, V., Gasthaus, J., Januschowski, T., 2020. DeepAR: probabilistic forecasting with autoregressive recurrent networks. International Journal of Forecasting 36, 1181–1191.

[23] Shiratori, T., Kobayashi, K., Takano, Y., 2020. Prediction of hierarchical time series using structured regularization and its application to artificial neural networks. PLos ONE 15, 1–23.

[24] Spiliotis, E., Abolghasemi, M., Hyndman, R.J., Petropoulos, F., Assimakopoulos, V., 2021. Hierarchical forecast reconciliation with machine learning. Applied Soft Computing 112, 107756.

[25] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R., 2014. Dropout: a simple way to prevent neural networks from overfitting. Journal of Machine Learning Research 15, 1929–1958.

[26] Suilin, A., 2018. Kaggle web traffic. https://github.com/Arturus/kaggle-web-traffic. Accessed: 2020-02-10.

[27] Taieb, S.B., Taylor, J.W., Hyndman, R.J., 2021. Hierarchical probabilistic forecasting of electricity demand with smart meter data. Journal of the American Statistical Association 116, 27–43.

[28] Wickramasuriya, S.L., 2021. Properties of point forecast reconciliation approaches. arXiv:2103.11129.
[29] Wickramasuriya, S.L., 2023. Probabilistic forecast reconciliation under the gaussian framework. Journal of Business & Economic Statistics , 1–14.
[30] Wickramasuriya, S.L., Athanasopoulos, G., Hyndman, R.J., 2019. Optimal forecast reconciliation for hierarchical and grouped time series through trace minimization. Journal of the American Statistical Association 114, 804–819.

Appendix A. Level-wise forecasting performances of the datasets

Table A.1: Average MSEs of the base forecasts and PRIAL from reconciliation methods for the Labour dataset.

Method	Total Employment	Occupation	Employment Status	Gender
	ARIMA			
BU	−36.1	−3.3	0.7	0.0
OLS	3.5	4.6	4.6	2.3
WLS	−7.0	3.6	5.3	2.9
MinT(Sample)	−46.8	−30.7	−28.7	−32.1
MinT(Shrink)	**4.6**	**5.3**	**7.8**	**4.7**
EMinT-U	−145.4	−143.2	−140.4	−138.7
NHR-TFNet	−18.3	−2.3	2.5	1.2
Base MSE ($\times 10^2$)	*37.6*	*7.8*	*4.2*	*1.8*
	ETS			
BU	−14.5	−3.5	2.5	0.0
OLS	3.9	**1.6**	3.2	0.4
WLS	3.1	1.0	3.7	0.1
MinT(Sample)	−11.3	−39.4	−28.7	−35.1
MinT(Shrink)	7.7	1.0	**4.6**	0.9
EMinT-U	−194.0	−201.2	−198.6	−196.7
NHR-TFNet	**12.1**	−0.8	4.5	**1.3**
Base MSE ($\times 10^2$)	*37.6*	*7.6*	*4.0*	*1.7*
	DeepAR			
BU	**27.9**	21.5	2.8	0.0
OLS	7.4	14.0	−2.4	−3.1
WLS	20.9	20.5	1.8	−0.4
MinT(Sample)	9.7	7.2	−12.4	−17.5
MinT(Shrink)	26.0	**24.4**	**7.1**	**4.1**
EMinT-U	−144.2	−87.4	−117.7	−118.3
NHR-TFNet	14.9	23.8	**7.1**	3.1
Base MSE ($\times 10^2$)	*88.6*	*10.8*	*4.3*	*1.8*

Table A.2: Average MSEs of the base forecasts and PRIAL from reconciliation methods for the Tourism dataset.

Method	Australia	States	Regions
	ARIMA		
BU	−122.2	−31.0	0.0
OLS	**−2.5**	9.1	5.1
WLS	−54.8	−3.1	6.4
MinT(Sample)	−9.5	**11.8**	**8.1**
MinT(Shrink)	−29.7	5.7	**8.1**
EMinT-U	−44.4	−24.5	−31.4
NHR-TFNet	−12.6	7.6	5.1
Base MSE ($\times 10^4$)	*242.2*	*21.2*	*1.1*
	ETS		
BU	−59.2	−20.4	0.0
OLS	−1.2	**3.4**	3.3
WLS	−29.4	−6.5	3.4
MinT(Sample)	**2.0**	2.2	2.4
MinT(Shrink)	−22.6	−3.8	**3.8**
EMinT-U	−127.7	−103.5	−102.4
NHR-TFNet	0.3	1.3	2.9
Base MSE ($\times 10^4$)	*249.4*	*18.1*	*0.9*
	DeepAR		
BU	−36.6	0.3	0.0
OLS	1.0	3.7	1.2
WLS	−19.0	7.6	2.5
MinT(Sample)	13.2	25.8	9.1
MinT(Shrink)	8.6	22.9	**10.2**
EMinT-U	−4.4	−2.5	−30.6
NHR-TFNet	**22.5**	**30.5**	6.2
Base MSE ($\times 10^4$)	*635.7*	*37.7*	*1.3*

Table A.3: Average MSEs of the base forecasts and PRIAL from reconciliation methods for the Wikipedia dataset.

Method	Total	Access	Agent	Language	Purpose	Article
			ARIMA			
BU	7.6	8.6	−2.7	3.9	6.3	0.0
OLS	4.4	6.3	−2.5	1.6	0.1	0.0
WLS	11.9	14.2	3.7	8.9	7.1	1.5
MinT(Shrink)	**15.2**	**19.2**	**9.2**	**14.8**	**11.2**	**4.7**
EMinT-U	−316.0	−258.0	−285.0	−293.5	−313.0	−323.0
NHR-TFNet	13.0	10.9	−1.0	6.7	3.5	0.0
Base MSE ($\times 10^4$)	*3811.9*	*1239.1*	*670.2*	*137.1*	*14.1*	*1.7*
			ETS			
BU	−0.4	4.9	4.6	0.8	−3.3	0.0
OLS	−1.9	2.8	2.3	−0.2	**−1.1**	0.5
WLS	0.4	5.2	5.5	1.5	−2.0	1.3
MinT(Shrink)	**0.8**	**5.5**	**5.8**	**1.9**	−1.7	**1.7**
EMinT-U	−694.0	−576.5	−558.6	−621.1	−631.7	−512.0
NHR-TFNet	−8.1	−4.4	−4.0	−4.6	−8.7	−0.2
Base MSE ($\times 10^4$)	*2799.7*	*897.6*	*552.0*	*100.8*	*10.6*	*1.5*
			DeepAR			
BU	33.6	15.8	18.2	21.0	8.6	0.0
OLS	17.4	−2.0	0.5	1.5	−2.2	−2.2
WLS	40.0	14.9	16.8	17.9	6.1	−3.0
MinT(Shrink)	**62.3**	**46.3**	**47.9**	**45.3**	**30.8**	**8.1**
EMinT-U	−52.9	−30.3	−27.0	−48.6	−107.9	−183.3
NHR-TFNet	−8.2	22.4	25.7	25.0	13.3	1.3
Base MSE ($\times 10^4$)	*11921.9*	*4766.0*	*2841.5*	*404.1*	*30.9*	*2.1*