Probabilistic Forecasting with Coherent Aggregation

Anonymous authors
Paper under double-blind review

Abstract

Obtaining accurate probabilistic forecasts is an important operational challenge in many applications, perhaps most obviously in energy management, climate forecast, supply chain planning, and resource allocation. In many of these applications, there is a natural hierarchical structure over the forecasted quantities; and forecasting systems that adhere to this hierarchical structure are said to be coherent. Furthermore, operational planning benefits from accuracy at all levels of the aggregation hierarchy. Building accurate and coherent forecasting systems, however, is challenging: classic multivariate time series tools and neural network methods are still being adapted for this purpose. In this paper, we augment an MQForecaster neural network architecture with a novel deep Gaussian factor forecasting model that achieves coherence by construction, yielding a method we call the *Deep Coherent Factor Model Neural Network* (`DeepCoFactor`) model. `DeepCoFactor` generates samples that can be differentiated with respect to the model parameters, allowing optimization on various sample-based learning objectives that align with the forecasting system's goals, including quantile loss and the scaled Continuous Ranked Probability Score (CRPS). In a comparison to state-of-the-art coherent forecasting methods, `DeepCoFactor` achieves significant improvements in scaled CRPS forecast accuracy, with gains between 4.16 and 54.40%, as measured on three publicly-available hierarchical forecasting datasets.

1 Introduction

Obtaining accurate forecasts is an important step for long-term planning in complex and uncertain environments, with applications ranging from energy to supply chain management, from transportation to climate prediction (Hong et al., 2014; Gneiting & Katzfuss, 2014; Makridakis et al., 2022a). Going beyond point forecasts such as means and medians, probabilistic forecasting provides a key tool for forecasting uncertain future events. This involves, e.g., forecasting that there is a 90% chance of rain on a certain day, or that there is a 99% chance that people will want to buy fewer than 100 items at a certain store on a certain week. Providing more detailed predictions of this form permits finer uncertainty quantification. This in turn permits planners to prepare for different scenarios and to allocate resources depending on their anticipated likelihood and cost structure. This can lead to better resource allocation, improved decision making, and less waste.

In many forecasting applications, there exist natural hierarchies over the quantities one wants to predict, such as energy consumption at various temporal granularities (from monthly to weekly), different geographic levels (from building-level to city-level to state-level), or retail demand for specific items (in a hierarchical product taxonomy). Typically, most or all levels of the hierarchy are important: the bottom levels are key for operational short-term planning, while higher levels of aggregation provide insights into longer-term or broader trends. Moreover, it is often desired that probabilistic forecasts are coherent (or consistent) to ensure efficient decision-making at all levels (Hong et al., 2014; Jeon et al., 2019). Coherence is achieved when the forecast distribution assigns zero probability to forecasts that do not satisfy the hierarchy's constraints (Panagiotelis et al., 2023; Ben Taieb et al., 2017a; Olivares et al., 2023) (see Definition 1.1). Designing an accurate model, capable of leveraging information from all hierarchical levels, while enforcing coherence is a well-known and challenging task (Hyndman et al., 2011).

The hierarchical forecasting literature has been dominated by two-stage reconciliation approaches, where univariate methods are first fitted and later reconciled towards coherence. For many years, most research

Method	End-to-End	Multivariate Forecast Distr.	Leverage Cross Series Information	Arbitrary Learning Objective
`PERMBU` (Ben Taieb et al., 2017b)	✗	✗	✗	✗
`Bootstrap` (Panagiotelis et al., 2023)	✗	✗	✗	✗
`Normality` (Wickramasuriya, 2023)	✗	✓	✗	✗
`DPMN` (Olivares et al., 2023)	✓	✓	✗	✗
`HierE2E` (Rangapuram et al., 2021)	✓	✗	✓	✓
`DeepCoFactor` (ours)	✓	✓	✓	✓

Table 1: Coherent forecast methods' desirable properties.

focused on mean reconciliation (Hyndman et al., 2011; Hyndman & Athanasopoulos, 2018; Vitullo, 2011; Hyndman et al., 2016; Dangerfield & Morris, 1992; Wickramasuriya et al., 2019; Mishchenko et al., 2019). More recent statistical methods consider coherent probabilistic forecasts through variants of the bootstrap reconciliation technique (Ben Taieb et al., 2017a; Panagiotelis et al., 2023) or the clever use of the Gaussian forecast distributions' properties (Wickramasuriya, 2023). Large-scale applications of hierarchical forecasting require one to simplify over the two-stage reconciliation process by favoring *end-to-end* approaches that simultaneously fit all levels of the hierarchy, while still achieving coherence. The end-to-end approach refers to training a model constrained to achieve coherence by optimizing directly for accuracy. End-to-end methods offer advantages such as reduced complexity, improved computational efficiency, and enhanced adaptability by streamlining the entire forecasting pipeline into a single, unified model. More importantly, end-to-end models generally achieve better accuracy compared to two-stage models that are first trained independently for optimized accuracy and then made coherent through various reconciliation approaches (Rangapuram et al., 2021; Olivares et al., 2023).

To the best of our knowledge, only three methods yield coherent probabilistic forecasts and allow models to be trained in an end-to-end manner: Rangapuram et al. (2021), Olivares et al. (2023), and Das et al. (2023). (There is also parallel research on hierarchical forecasts with relaxed constraints (Han et al., 2021; Paria et al., 2021; Kamarthi et al., 2022), but this line of work is less relevant since our focus is on strictly coherent forecasting methods.) In particular, Olivares et al. (2023) considers a finite mixture of Poisson distributions that captures correlations implicitly through latent variables and does not leverage cross-time series information. On the other hand, Rangapuram et al. (2021) leverages the multivariate time series information, but it achieves coherence through a differentiable projection layer, which could degrade forecast accuracy: it does not directly model correlations between the multivariate outputs, but rather it couples them through its projection layer. Dedicated effort is still necessary to capture these hierarchical relationships to improve forecast accuracy. Such probabilistic methods can benefit from the ability to optimize for arbitrary loss functions through making samples differentiable, as demonstrated by Rangapuram et al. (2021). The capacity to optimize any loss computed from forecast samples can help align the forecasting system's goals with the neural network's learning objective.

More generally, an ideal hierarchical forecasting method should satisfy several desiderata: 1) be end-to-end coherent; 2) model a joint multivariate probability distribution, capturing the intricate relationships between series within the hierarchy; 3) leverage cross time series information to accurately reflect these relationships; and 4) generate differentiable samples, to enable the method to optimize for arbitrary learning objectives that align with the forecasting system's goals. In this paper, we present a method which satisfies all of these ideal properties; see Table 1 for a summary.

Our **main contributions** are the following.

1. We introduce the *Deep Coherent Factor Model Neural Network* (`DeepCoFactor`) model, a method for probabilistic coherent forecasting that satisfies all the desired properties stated above. `DeepCoFactor` achieves forecast coherence exactly by construction, producing a joint forecast distribution over the bottom-level quantities in the hierarchy, and simply aggregating its samples up. Our approach is

generic and can be applied to most univariate neural forecasting models with minimal modifications to achieve vector autoregressive-like (VAR-like) behavior. The framework can be based on any generic deep learning univariate forecasting model. In this paper, we test the modifications on the well-performing `MQCNN` architecture (Wen et al., 2017; Olivares et al., 2022b), an MQForecaster neural network architecture which specializes in mult-step probabilistic forecast.

2. `DeepCoFactor` uses a multivariate factor model to capture the hierarchy's complex correlation structure. Unlike existing joint distribution models that typically optimize for likelihood, we directly optimize this neural network to achieve high marginal forecast accuracy, measured using the Continuous Ranked Probability Score (CRPS) (Matheson & Winkler, 1976), through sample differentiability and the reparametrization trick. This new learning objective enhances the robustness of the probabilistic model to mis-specification, leading to improved forecast accuracy. Furthermore, our model is versatile and can be tailored to optimize various forecast metrics, such as quantile losses, CRPS, mean squared error, or combinations thereof, depending on the specific use case.

3. The `DeepCoFactor` model possesses the adaptability to fit complex temporal relationships between known data and future predictions, while also modeling correlations between elements in the hierarchy. We demonstrate `DeepCoFactor`'s flexibility by achieving state-of-the-art results on three public datasets. Our findings show that `DeepCoFactor` improves accuracy by **4.16** to **54.40**%, depending on the dataset. Additionally, we evaluate our mean forecasts using the relative squared error and find that our method surpasses previous methods by **14.56** to **95.98**% across all three datasets.

Hierarchical forecasting notations. We denote a hierarchical multivariate time series vector by $\mathbf{y}_{[i],t} = \left[\mathbf{y}^{\top}_{[a],t} \mid \mathbf{y}^{\top}_{[b],t}\right] \in \mathbb{R}^{N_a+N_b}$, where $[i], [a]$, and $[b]$ denote the set of full, aggregate and bottom indices of the time series, respectively. There are $|[i]| = N_a + N_b$ time series in total, with N_a aggregates from the N_b bottom time series, at the finest level of granularity. We use t as a time index. In our notations, we keep track of shape of tensors using square brackets in subscripts. Since each aggregated time series is a linear transformation of the multivariate bottom series, we write the hierarchical aggregation constraint as

$$\mathbf{y}_{[i],t} = \mathbf{S}_{[i][b]}\mathbf{y}_{[b],t} \iff \begin{bmatrix}\mathbf{y}_{[a],t}\\ \mathbf{y}_{[b],t}\end{bmatrix} = \begin{bmatrix}\mathbf{A}_{[a][b]}\\ \mathbf{I}_{[b][b]}\end{bmatrix}\mathbf{y}_{[b],t}. \tag{1}$$

The aggregation matrix $\mathbf{A}_{[a][b]}$ represents the collection of linear transformations for deriving the aggregates, and sums the bottom series to the aggregate levels. The hierarchical aggregation constraints matrix $\mathbf{S}_{[i][b]}$ obtained by stacking $\mathbf{A}_{[a][b]}$ and the $N_b \times N_b$ identity matrix $\mathbf{I}_{[b][b]}$.

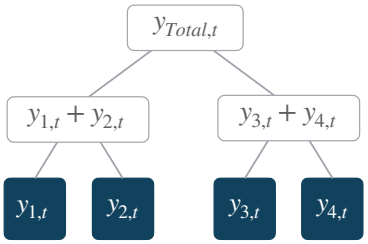
For a simple example, consider $N_b = 4$ bottom-series, so $[b] = \{1, 2, 3, 4\}$ and $y_{Total,t} = \sum_{i=1}^{4} y_{i,t}$. Figure 1 shows an example of such hierarchical structure, where the multivariate hierarchical time series is defined

$$\mathbf{y}_{[a],t} = \left[y_{Total,t},\ y_{1,t} + y_{2,t}\ y_{3,t} + y_{4,t}\right]^{\top}, \qquad \mathbf{y}_{[b],t} = \left[y_{1,t},\ y_{2,t},\ y_{3,t},\ y_{4,t}\right]^{\top}. \tag{2}$$

Consider historical temporal features $\mathbf{x}^{(h)}_{[b][:t]}$, known future information $\mathbf{x}^{(f)}_{[b][t+1:t+N_h]}$, and static data $\mathbf{x}^{(s)}_{[b]}$, forecast creation date t and forecast horizons in $[t+1 : t+N_h]$. A multi-step multivariate forecasting task aims to estimate the following conditional probability:

$$\mathbb{P}\left(\mathbf{Y}_{[i],t+\eta} \mid \mathbf{x}^{(h)}_{[b][:t]},\ \mathbf{x}^{(f)}_{[b][t+1:t+N_h]},\ \mathbf{x}^{(s)}_{[b]}\right) \qquad \text{for } \eta = 1, \cdots, N_h. \tag{3}$$

A hierarchical forecasting task augments the forecast probability with coherence constraints (Ben Taieb et al., 2020; Panagiotelis et al., 2023; Olivares et al., 2022b), by restricting the probabilistic forecast space to assign zero probability to non-coherent forecasts. We can formalize this through Definition 1.1, which essentially tells us that the distribution of a given aggregate random variable is exactly the distribution defined as the aggregates of the bottom-series distributions through the summation matrix $\mathbf{S}_{[i][b]}$.



(a) Graph representation

$$\mathbf{S}_{[i][b]} = \begin{bmatrix} \mathbf{A}_{[a][b]} \\ \\ \mathbf{I}_{[b][b]} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ \hline 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ \hline 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

(b) Matrix representation

Figure 1: A simple time series hierarchical structure with $N_a = 3$ aggregates over $N_b = 4$ bottom time series. Figure 1a shows the disaggregated bottom variables with blue background. Figure 1b (right) shows the corresponding hierarchical aggregation constraints matrix with horizontal lines to separate levels of the hierarchy. We decompose our evaluation throughout the levels.

Definition 1.1. Let $\Omega_{[b]}, \mathcal{F}_{[b]}, \mathbb{P}_{[b]}$ be a probabilistic forecast space (on the bottom-level series), with sample space $\Omega_{[b]}$, $\mathcal{F}_{[b]}$ its σ-algebra, and $\mathbb{P}_{[b]}$ a forecast probability. A **coherent forecast** space $\Omega_{[i]}, \mathcal{F}_{[i]}, \mathbb{P}_{[i]}$ satisfies

$$\mathbb{P}_{[i]}\left(\mathbf{S}_{[i][b]}(\mathcal{B})\right) = \mathbb{P}_{[b]}(\mathcal{B}), \tag{4}$$

for any set $\mathcal{B} \in \mathbb{F}_{[b]}$ and set's image $\mathbf{S}_{[i][b]}(\mathcal{B}) \in \mathbb{F}_{[i]}$.

Definition 1.2. Any bottom-level multivariate distribution can be transformed into a coherent distribution through **coherent aggregation**.[1] Given a sample $\hat{\mathbf{y}}_{[b]} \sim \mathbb{P}_{[b]}$, a coherent $\mathbb{P}_{[i]}$ distribution can be constructed with the following sample transformation

$$\tilde{\mathbf{y}}_{[i]} = \mathbf{S}_{[i][b]}\left(\hat{\mathbf{y}}_{[b]}\right). \tag{5}$$

In other words, it is enough to aggregate the bottom-level forecasts in a bottom-up manner. We include in Appendix A a proof of the approach's coherence property.

2 Methodology

In this section, we describe our main method, the *Deep Coherent Factor Model Neural Network* (`DeepCoFactor`) model. It consists of a multivariate probabilistic model, an underlying neural network structure, and an end-to-end model estimation procedure.

2.1 Multivariate Probabilistic Model

Our predicted probabilistic forecasts at all hierarchical levels are jointly represented by a Gaussian factor model. Our neural network maps the known information (past, static and known future) to the location, scale and shared factor parameters, and the forecasted factor model parameters are designed to model correlations between the bottom-level series, while conditioning on all known information. Our factor model[2] combined with the coherent aggregation in Eqn. 8 directly estimates the multivariate probability of bottom-level series $\mathbf{y}_{[b][t+1:t+N_h]}$ conditioning on historical, known-future, and static covariates $\mathbf{x}^{(h)}_{[b][:t]}$, $\mathbf{x}^{(f)}_{[b][t+1:t+N_h]}, \mathbf{x}^{(s)}_{[b]}$, i.e.,

$$\mathbb{P}\left(\tilde{\boldsymbol{Y}}_{[i][t+1:t+N_h]} \mid \mathbf{x}^{(h)}_{[b][:t]}, \mathbf{x}^{(f)}_{[b][t+1:t+N_h]}, \mathbf{x}^{(s)}_{[b]}\right) = \mathbb{P}\left(\mathbf{S}_{[i][b]}\hat{\boldsymbol{Y}}_{[b][t+1:t+N_h]} \mid \hat{\boldsymbol{\mu}}_{[b][h],t},\ \hat{\boldsymbol{\sigma}}_{[b][h],t}, \hat{\mathbf{F}}_{[b][k][h],t}\right). \tag{6}$$

At a given forecast creation date t, the model uses the location $\hat{\boldsymbol{\mu}}_{[b][h],t} \in \mathbb{R}^{N_b \times N_h}$, scale $\hat{\boldsymbol{\sigma}}_{[b][h],t} \in \mathbb{R}^{N_b \times N_h}$ and shared factor $\hat{\mathbf{F}}_{[b][k][h],t} \in \mathbb{R}^{N_b \times N_k \times N_h}$ parameters, along with samples from standard normal variables

[1] Coherent aggregation can be though of a special case of bootstrap reconciliation (Panagiotelis et al., 2023) that only relies on a bottom-level forecast distribution.

[2] Early work on factor forecast models augmenting neural networks done by Wang et al. (2019) does not ensure coherence.

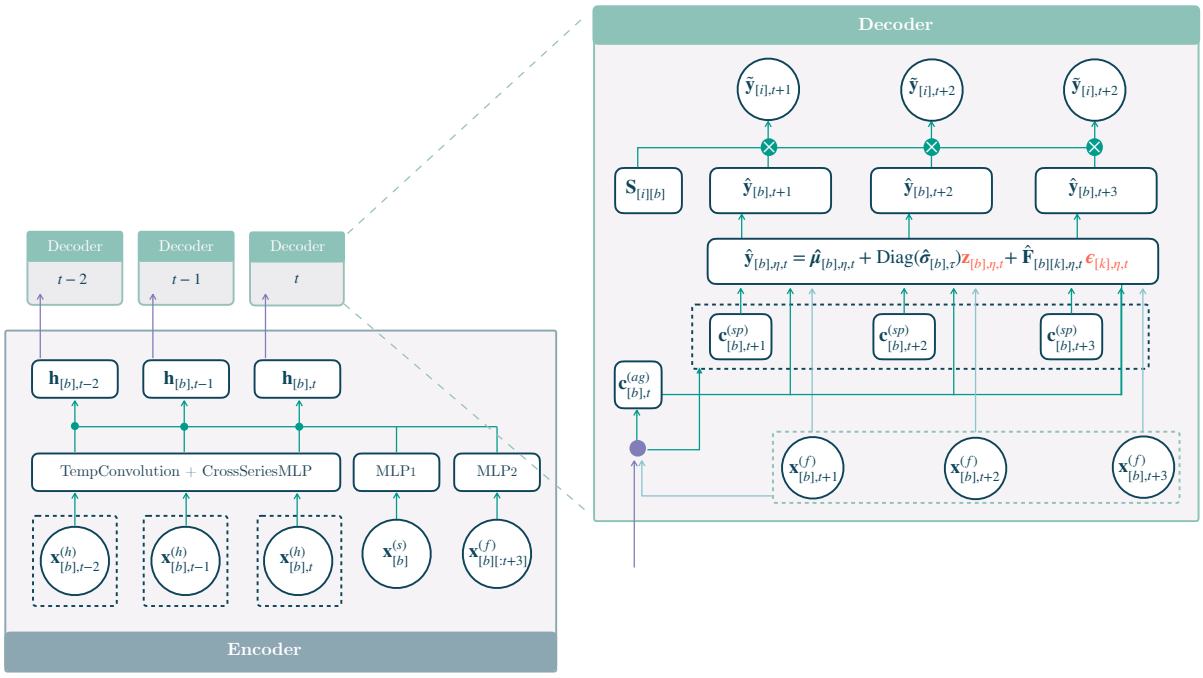


Figure 2: The *Deep Coherent Factor Model Neural Network* is a Sequence-to-Sequence with Context network that uses dilated temporal convolutions as the primary encoder and multilayer perceptron based decoders for the creation of the multi-step forecast. `DeepCoFactor` leverages coherently aggregates the samples of the factor model $\tilde{\mathbf{y}}_{[i],\eta,t} = \mathbf{S}_{[i][b]}\hat{\mathbf{y}}_{[b],\eta,t}$. We mark in red the standard normal samples that are parameter-free, the reparametrization trick allows to apply backpropagation through the factor model outputs.

$\mathbf{z}_{[b],\eta} \sim \mathcal{N}(\mathbf{0}_{[b]}, \mathbf{I}_{[b][b]})$, and $\boldsymbol{\epsilon}_{[k],\eta} \sim \mathcal{N}(\mathbf{0}_{[k]}, \mathbf{I}_{[k][k]})$ to compose the following multivariate variables for each horizon:

$$\hat{\mathbf{y}}_{[b],\eta,t} = \hat{\boldsymbol{\mu}}_{[b],\eta,t} + \mathrm{Diag}(\hat{\boldsymbol{\sigma}}_{[b],\eta,t})\mathbf{z}_{[b],\eta,t} + \hat{\mathbf{F}}_{[b][k],\eta,t}\boldsymbol{\epsilon}_{[k],\eta,t}, \qquad \eta = 1, \cdots, N_h. \tag{7}$$

After sampling from the multivariate factor we *coherently aggregate* the clipped outputs of the network,

$$\tilde{\mathbf{y}}_{[i],\eta,\tau} = \mathbf{S}_{[i][b]}\left(\hat{\mathbf{y}}_{[b],\eta,\tau}\right)_{+}, \tag{8}$$

where $(\cdot)_{+} = \max(\cdot, 0)$ returns the nonnegative part of its argument.

The shared factors enable the factor model to capture the relationships across the disaggregated series, and the covariance structure of the disaggregated series follows:

$$\mathrm{Cov}\left(\hat{\mathbf{y}}_{[b],\eta,t}\right) = \mathrm{Diag}(\hat{\boldsymbol{\sigma}}^{2}_{[b],\eta,t}) + \hat{\mathbf{F}}_{[b][k],\eta,t}\hat{\mathbf{F}}^{\top}_{[b][k],\eta,t}. \tag{9}$$

We include in Appendix A a proof of the covariance structure our multivariate factor model.

2.2 Neural Network Architecture

Our framework can adapt to any univariate deep learning forecasting models, so long as they can be made to output factor model parameters. In this paper, we focus on the architecture implementation based on `MQCNN` (Wen et al., 2017; Olivares et al., 2022b), because of its outstanding performance in multi-step forecasting problems. Our `MQCNN`-based architecture has a main encoder which consists of a stack of dilated temporal convolutions, and it is applied to historical information for all series. In addition, it uses a global multi-layer perceptron (MLP) to encode the static and future information. The encoder at time t is described

in Eqn. 10 below, and is applied to each disaggregated series:

$$\begin{aligned}\mathbf{h}^{(h)}_{[i],t} &= \text{TempConvolution}\left([\mathbf{S}_{[i][b]}\mathbf{x}^{(h)}_{[b][:t]}]\right) \ {}^{(3)}\\ \mathbf{h}^{(s)}_{[b]} &= \text{MLP}_1\left(\mathbf{x}^{(s)}_{[b]}\right)\\ \mathbf{h}^{(f)}_{[b],t} &= \text{MLP}_2\left(\mathbf{x}^{(f)}_{[b][t+1:t+N_h]}\right).\end{aligned} \tag{10}$$

We use a residual cross series MLP to capture vector autorregressive relationships in the hierarchy with minimal modifications to the architecture:

$$\mathbf{h}^{(h)}_{[b],t} = \text{CrossSeriesMLP}\left(\mathbf{h}^{(h)}_{[i],t}\right). \tag{11}$$

The `DeepCoFactor` uses a two-stage MLP decoder: the first decoder summarizes information into the horizon agnostic context $\mathbf{c}^{(ag)}_{[b],t}$ and the horizon specific context $\mathbf{c}^{(sp)}_{[b][h],t}$; and the second stage decoder transforms the contexts into the Factor model parameters $\left(\hat{\boldsymbol{\mu}}_{[b][h],t},\ \hat{\boldsymbol{\sigma}}_{[b][h],t}, \hat{\mathbf{F}}_{[b][k][h],t}\right)$. Eqn. 12 describes the operations:

$$\begin{aligned}\mathbf{h}_{[b],t} &= \left[\mathbf{h}^{(h)}_{[b],t}, \mathbf{h}^{(s)}_{[b]}, \mathbf{h}^{(f)}_{[b],t}\right]\\ \mathbf{c}^{(ag)}_{[b],t} &= \text{MLP}_3\left(\mathbf{h}_{[b],t}\right)\\ \mathbf{c}^{(sp)}_{[b][h],t} &= \text{MLP}_4\left(\mathbf{h}_{[b],t}\right)\\ \left(\hat{\boldsymbol{\mu}}_{[b][h],t},\ \hat{\boldsymbol{\sigma}}_{[b][h],t}, \hat{\mathbf{F}}_{[b][k][h],t}\right) &= \text{MLP}_5\left(\left[\mathbf{c}^{(sp)}_{[b][h],t}, \mathbf{c}^{(ag)}_{[b],t}, \mathbf{x}^{(f)}_{[b][t+1:t+N_h]}\right]\right).\end{aligned} \tag{12}$$

For the last step, the network composes the factor model samples, using Eqn. 7 and aggregates them (equivalent to bottom-up reconciliation) in Eqn. 8.

We design our method so that it can provide differentiable samples: we can differentiate sample-based losses with respect to our distributional parameters, allowing to differentiate with respect to the neural network weights. The forecast representation (i.e., Gaussian linear latent variable models) leverages the simple reparametrization trick for Gaussian random variables (Kingma & Welling, 2013). However, recent work (Figurnov et al., 2018; Ruiz et al., 2016; Jankowiak & Obermeyer, 2018) has shown that one can sample in a differentiable manner from almost any continuous distribution. Our method exploits these results. If we can compute differentiable samples from the factor distributions and from the bottom-level distributions, we can compute differentiable samples for our forecasts, at any level of aggregation. This then allows us to optimize any metric which can be estimated as a differentiable function of samples.

Differentiable sampling is implemented for many distributions of interest in the Pytorch open source machine learning framework PyTorch[4] (Paszke et al., 2019; Bradbury et al., 2018; Abadi et al., 2015). The differentiable sampling approach is fairly easy to implement for many different distributions. We demonstrate this in the PyTorch code snippet in Figure 4 in Appendix B. We only need to change a single line of code to change distribution assumptions.

2.3 Learning Objective

Let θ be a model that resides in the class of models Θ defined by the model architecture. Here θ can be thought of a non-linear function mapping from the model feature space to the parameter set for all target horizons, we have

$$(\hat{\boldsymbol{\mu}}_{[b][h],t},\ \hat{\boldsymbol{\sigma}}_{[b][h],t}, \hat{\mathbf{F}}_{[b][k][h],t}) = \theta(\mathbf{x}^{(h)}_{[b][:t]},\ \mathbf{x}^{(f)}_{[b][t+1:t+N_h]}, \mathbf{x}^{(s)}_{[b]}). \tag{13}$$

Let $\hat{Y}_{i,\eta,t}(\theta)$ be the random variable parameterized by θ. In some problems, multi-step coherent forecasts for multiple items are needed (e.g., in retail business, coherent regional demand forecasts are required for each

[3]Temporal exogenous data only aggregates the target signal, other features (e.g. calendar) are maintained without aggregation.

[4]`https://pytorch.org/docs/stable/distributions.html`, and also in Jax and Tensorflow. See `rsample` methods in PyTorch.

product). Let u be the index of such item within an index set $\{1, \cdots, N_u\}$ of interest, and let $\tilde{Y}_{u,i,\eta,t}(\theta)$ be the coherent forecast random variable for target $y_{u,i,t+\eta}$. Using the reparametrization strategy (Kingma & Welling, 2013), within the class of parameters Θ defined by the neural network architecture, we optimize for

$$\min_{\theta \in \Theta} \sum_{u,i,\eta,t} \text{CRPS}\left(y_{u,i,t+\eta},\ \tilde{Y}_{u,i,\eta,t}(\theta)\right), \tag{14}$$

where the CRPS between a target y and distributional forecast $\tilde{Y}$ (Matheson & Winkler, 1976; Gneiting & Raftery, 2007) is defined as

$$\text{CRPS}\left(y, \tilde{Y}\right) = \mathbb{E}_{\tilde{Y}}\left[\,|\tilde{Y} - y|\,\right] - \frac{1}{2}\mathbb{E}_{\tilde{Y},\tilde{Y}'}\left[\,|\tilde{Y} - \tilde{Y}'|\,\right], \tag{15}$$

where Y' is distributed as Y, but is independent of it. We optimize the model using adaptive moments stochastic gradient descent (`Adam` (Kingma & Ba, 2014)) with early stopping (Yao et al., 2007). Additional details on the neural network optimization and hyperparameter selection are available in Appendix C.

2.4 Discussion

Here, we discuss differences between `DeepCoFactor` (our method) with two end-to-end coherent probabilistic forecasting baselines in `HierE2E` (Rangapuram et al., 2021) and `DPMN` (Olivares et al., 2023).

The `HierE2E` method of Rangapuram et al. (2021) is "too general." It consists of an augmented `DeepVAR` neural network model (Flunkert et al., 2017) that produces probabilistic forecasts for all time-series in the hierarchy. `HierE2E` claims to be more general than hierarchical forecasting, since it is designed to enforce any convex constraint satisfied by the forecasts; due to the constraining operation in the method, it has to revise the optimized forecasts. It does not leverage specifics of the hierarchical constraints, which are more structured than a general convex constraint. `HierE2E` produces forecast samples from Gaussian distributions for each time-series in the hierarchy, assuming independence; since the samples are not guaranteed to be hierarchically coherent, `HierE2E` couples samples by projecting them on the space of coherent probabilistic forecasts. Both the sampling operation (Kingma & Welling, 2013) and the projection are differentiable, allowing the method to be trained end-to-end. `HierE2E` allows different distribution choices, although they are not explored in the initial paper, since Gaussians can be replaced by any distribution which can be sampled in a differentiable way, i.e., almost any continuous distribution (Ruiz et al., 2016; Figurnov et al., 2018; Jankowiak & Obermeyer, 2018). In Rangapuram et al. (2021), the projection operator ensures coherence, and correlations between bottom-levels are learned only by optimizing the neural network. In contrast, `DeepCoFactor` produces forecasts for bottom-level series only, while relying on common factors to encode correlations. This removes the need to forecast at all levels simultaneously, therefore reducing computational requirements if we are only interested in a subset of the aggregates.

On the other hand, the `DPMN` baseline (Olivares et al., 2023) is "too restrictive," in particular as a Poisson Mixture can be prone to distribution mis-specification problems. It is known that when a probability model is mis-specified, optimizing log likelihood is equivalent to minimizing Kullback–Leibler (KL) divergence with respect to the true probabilistic distribution, KL divergence measures change in probability space, while optimizing CRPS is equivalent to minimizing the Cramer-von Mises criterion (Gneiting & Raftery, 2007), which quantifies the distance with respect to the probability model in the sample space. The `DeepCoFactor` learning objective for the probabilistic model is resilient to distributional mis-specification (Bellemare et al., 2017). Moreover `DeepCoFactor` can be optimized to adapt for other evaluation metrics of interest.

Finally `DPMN` estimates the covariance among time series, but it does not leverage this when encoding the historical time series. Similar to other `ARIMA` based baselines, on specific hierarchical benchmark datasets such as `Traffic`, `DPMN` produces sub-optimal bottom-series forecasts. We improve the encoder for historical time series by adding a CrossSeriesMLP after the Temporal convolution encoder, which bridges the accuracy gap between `HierE2E` and our `MQCNN` based approach.

Dataset	# Items (N_u)	Bottom (N_b)	Levels	Aggregated ($N_a + N_b$)	Time range	Frequency	Horizon (N_h)
`Favorita`	4036	54	4	93	1/2013 - 8/2017	Daily	34
`Tourism-L`	1	304	4/5	555	1998-2016	Monthly	12
`Traffic`	1	200	4	207	1/2008-3/2009	Daily	1

Table 2: Summary of publicly-available data used in our empirical evaluation.

3 Empirical Evaluation

In this section, we present our main empirical results. First, we describe the empirical set up. Second, we evaluate the proposed model, and compare with state-of-the-art hierarchical forecast models. Third, we present ablation study results that further analyze the source of improvements on variants of the `DeepCoFactor`.

3.1 Setting

Datasets. In our analysis, we consider three qualitatively different (public) datasets: `Favorita`, `Tourism-L`, and `Traffic`. They have different properties which are representative of more realistic non-public data, and forecasting all of them accurately requires substantial modeling flexibility. The `Favorita` dataset is a large retail dataset, and it contains both count data (whole items) and real-valued data (items sold by weight) for over 4000 items. The aggregation hierarchy is regional. We use it to test our method on a (relatively) large-scale problem. The `Tourism-L` dataset represents the number of visitors to different regions in Australia. The goal is to forecast thousands of visitors, i.e., rescaling count data by 1000. The aggregation is done according to a hierarchy over region and purpose of travel, allowing us to test a case where the aggregate levels have overlap. Finally, the `Traffic` dataset contains sum-aggregates of highway occupancy rates. The initial rates are hourly, but (following (Olivares et al., 2023; Rangapuram et al., 2021)) the dataset we consider is daily, i.e., it uses rates already aggregated to the daily level for each highway bend as bottom-level series. The hierarchy in this dataset was defined randomly over highway bends. We use the same hierarchy as previous work. This allows us to test whether our model requires aggregations to be in line with correlation structures to achieve high accuracy. For all three datasets, the forecasted quantities are non-negative. We describe dataset details in Appendix D.

Evaluation metrics. Our main evaluation metric is the mean scaled CRPS (Bolin & Wallin, 2019; Makridakis et al., 2022b) defined as the score described in Eqn. 16, divided by the sum of all target values. Let $\boldsymbol{l}^{(g)}$ be a vector of length $N_a + N_b$ consisting of binary indicators for a hierarchical level g, where for each $j \in [i]$, $l_j^{(g)} = 1$ if aggregated series j is included in hierarchical level g, and 0 otherwise. Then sCRPS for hierachical level g is defined as

$$\text{sCRPS}\left(\mathbf{y}_{[i][t+1:t+N_h]},\ \tilde{Y}_{[i][h],t} \mid \boldsymbol{l}^{(g)}\right) = \frac{\sum_{i=1}^{N_a+N_b}\left(\sum_{\eta=1}^{N_h} \text{CRPS}(y_{i,t+\eta}, \tilde{Y}_{i,\eta,t})\right) \cdot l_i^{(g)}}{\sum_{i=1}^{N_a+N_b} ||\mathbf{y}_{i,[t+1:t+N_h]}||_1 \cdot l_i^{(g)}}. \tag{16}$$

We also evaluate mean forecasts denoted by $\bar{\mathbf{y}}_{[i][h],t} := (\bar{\mathbf{y}}_{[i],1,t}, \cdots, \bar{\mathbf{y}}_{[i],N_h,t})$ through the *relative squared error* relSE (Hyndman & Koehler, 2006), that considers the ratio between squared error across forecasts in all levels over squared error of the `Naive` forecast (i.e., a point forecast using the last observation $\mathbf{y}_{[i],t}$) as described by

$$\text{relSE}\left(\mathbf{y}_{[i][h],t},\ \bar{\mathbf{y}}_{[i][t+1:t+N_h]} \mid \boldsymbol{l}^{(g)}\right) = \frac{\sum_{i=1}^{N_a+N_b} \|\mathbf{y}_{i,[t+1:t+N_h]} - \bar{\mathbf{y}}_{i,[h],t}\|_2^2 \cdot l_i^{(g)}}{\sum_{i=1}^{N_a+N_b} \|\mathbf{y}_{i,[t+1:t+N_h]} - y_{i,t} \cdot \mathbf{1}_{[h]}\|_2^2 \cdot l_i^{(g)}}. \tag{17}$$

Baseline Models. We compare our method with the following coherent probabilistic methods: (1) `DPMN-GroupBU` (Olivares et al., 2023), (2) `HierE2E` (Rangapuram et al., 2021), (3) `ARIMA-PERMBU-MinT` (Ben Taieb et al., 2017b), (4) `ARIMA-Bootstrap-BottomUp` (Panagiotelis et al., 2023) and (5) an `ARIMA`. In addition, we compare our method with the following coherent mean methods: (1) `DPMN-GroupBU`, (2) `ARIMA-ERM` (Ben

Table 3: Empirical evaluation of probabilistic coherent forecasts. Mean *scaled CRPS* (sCRPS) averaged over 5 runs, at each aggregation level, the best result is highlighted (lower values are preferred). Methods without standard deviation have deterministic solutions.

* The HierE2E results differ from Rangapuram et al. (2021), here the sCRPS quantile interval space has a finer granularity of 1 percent instead of 5 percent in Rangapuram et al. (2021).

** PERMBU-MinT on Tourism-L is unavailable because the original implementation cannot be applied to datasets with multiple hierarchies.

Data	Level	DeepCoFactor (coherent)	DPMN-GroupBU (coherent)	HierE2E * (coherent)	PERMBU-MinT ** (coherent)	Bootstrap-BottomUp (coherent)	ARIMA (not coherent)
Favorita	Overall	**0.2908 ± 0.0025**	0.4020 ± 0.0182	0.5298 ± 0.0091	0.4670 ± 0.0096	0.4110 ± 0.0085	0.4373
	1 (geo.)	**0.1841 ± 0.0033**	0.2760 ± 0.0149	0.4714 ± 0.0103	0.2692 ± 0.0076	0.2900 ± 0.0067	0.3112
	2 (geo.)	**0.2754 ± 0.0026**	0.3865 ± 0.0207	0.5182 ± 0.0107	0.3824 ± 0.0092	0.3877 ± 0.0082	0.4183
	3 (geo.)	**0.2945 ± 0.0025**	0.4068 ± 0.0206	0.5291 ± 0.0129	0.6838 ± 0.0108	0.4490 ± 0.0098	0.4446
	4 (geo.)	**0.4092 ± 0.0022**	0.5387 ± 0.0253	0.6012 ± 0.0131	0.5532 ± 0.0116	0.5749 ± 0.0003	0.5749
Tourism-L	Overall	**0.1197 ± 0.0037**	0.1249 ± 0.0020	0.1472 ± 0.0029	-	0.1375 ± 0.0013	0.1416
	1 (geo.)	0.0292 ± 0.0042	0.0431 ± 0.0042	0.0842 ± 0.0051	-	0.0622 ± 0.0026	**0.0263**
	2 (geo.)	**0.0593 ± 0.0049**	0.0637 ± 0.0032	0.1012 ± 0.0029	-	0.0820 ± 0.0019	0.0904
	3 (geo.)	**0.1044 ± 0.0030**	0.1084 ± 0.0033	0.1317 ± 0.0022	-	0.1207 ± 0.0010	0.1389
	4 (geo.)	**0.1540 ± 0.0046**	0.1554 ± 0.0025	0.1705 ± 0.0023	-	0.1646 ± 0.0007	0.1878
	5 (prp.)	**0.0594 ± 0.0076**	0.0700 ± 0.0038	0.0995 ± 0.0061	-	0.0788 ± 0.0018	0.0770
	6 (prp.)	0.1100 ± 0.0049	**0.1070 ± 0.0023**	0.1336 ± 0.0042	-	0.1268 ± 0.0017	0.1270
	7 (prp.)	**0.1824 ± 0.0024**	0.1887 ± 0.0032	0.1955 ± 0.0025	-	0.1949 ± 0.0010	0.2022
	8 (prp.)	**0.2591 ± 0.0050**	0.2629 ± 0.0034	0.2615 ± 0.0016	-	0.2698 ± 0.0004	0.2834
Traffic	Overall	**0.0171 ± 0.0036**	0.0907 ± 0.0024	0.0375 ± 0.0058	0.0677 ± 0.0061	0.0736 ± 0.0024	0.0751
	1 (geo.)	**0.0026 ± 0.0012**	0.0397 ± 0.0044	0.0183 ± 0.0091	0.0331 ± 0.0085	0.0468 ± 0.0031	0.0376
	2 (geo.)	**0.0029 ± 0.0014**	0.0537 ± 0.0024	0.0183 ± 0.0081	0.0341 ± 0.0081	0.0483 ± 0.0030	0.0412
	3 (geo.)	**0.0044 ± 0.0022**	0.0538 ± 0.0022	0.0209 ± 0.0071	0.0417 ± 0.0061	0.0530 ± 0.0025	0.0549
	4 (geo.)	**0.0587 ± 0.0106**	0.2155 ± 0.0022	0.0974 ± 0.0021	0.1621 ± 0.0027	0.1463 ± 0.0017	0.1665

Table 4: Empirical evaluation of mean hierarchical forecasts. *Relative squared error* (relSE) averaged over 5 runs, at each aggregation level, the best result is highlighted (lower values are preferred). Methods without standard deviation have deterministic solutions.

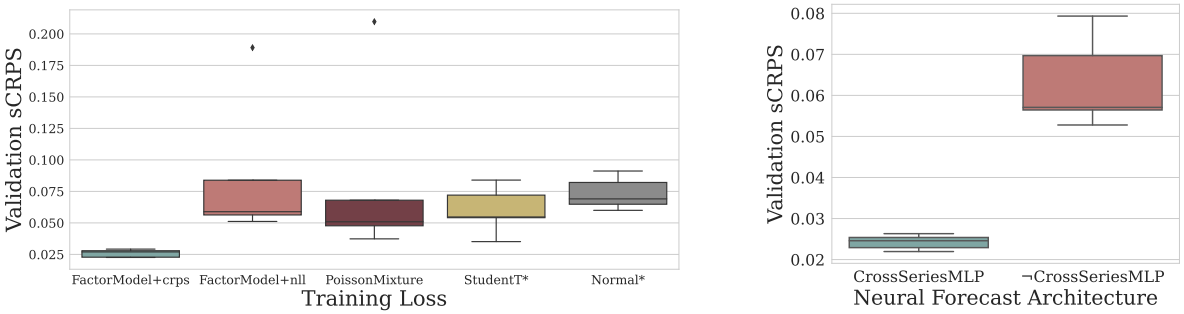
* The ARIMA-ERM results for Tourism-L differ from Rangapuram et al. (2021), as we improved the numerical stability of their implementation.

Data	Level	DeepCoFactor (hier.)	DPMN-GroupBU (hier.)	ARIMA-ERM * (hier.)	ARIMA-MinT-ols (hier.)	ARIMA-BottomUp (hier.)	ARIMA (not hier.)	SNaive (not hier.)
Favorita	Overall	**0.5885 ± 0.0291**	0.7563 ± 0.0713	0.8163	0.9465	0.8276	0.9665	1.1420
	1 (geo.)	**0.6109 ± 0.0400**	0.7944 ± 0.0568	0.8362	0.8999	0.8415	0.9217	1.1269
	2 (geo.)	**0.5618 ± 0.0265**	0.7355 ± 0.1057	0.7830	1.0057	0.8050	1.0451	1.1078
	3 (geo.)	**0.5619 ± 0.0256**	0.7303 ± 0.1035	0.7986	1.0418	0.8192	1.0881	1.1315
	4 (geo.)	**0.5854 ± 0.0130**	0.6770 ± 0.0351	0.8199	0.8808	0.8228	0.8228	1.2815
Tourism-L	Overall	**0.0951 ± 0.0145**	0.1113 ± 0.0158	0.1178	0.1251	0.2979	0.1414	0.1306
	1 (geo.)	0.0447 ± 0.0171	0.0597 ± 0.0212	0.0596	0.0472	0.4002	**0.0343**	0.0582
	2 (geo.)	**0.1014 ± 0.0180**	0.1121 ± 0.0152	0.1293	0.1476	0.3340	0.2530	0.1628
	3 (geo.)	0.2309 ± 0.0124	**0.2250 ± 0.0196**	0.2529	0.3556	0.4238	0.4429	0.3695
	4 (geo.)	0.3075 ± 0.0134	**0.2980 ± 0.0197**	0.3236	0.4288	0.4012	0.4835	0.4766
	5 (prp.)	**0.0596 ± 0.0195**	0.0798 ± 0.0195	0.0895	0.0856	0.1703	0.0973	0.0615
	6 (prp.)	**0.1199 ± 0.0115**	0.1403 ± 0.0150	0.1466	0.1537	0.1986	0.1663	0.1577
	7 (prp.)	**0.2484 ± 0.0119**	0.2654 ± 0.0212	0.2705	0.3017	0.3151	0.2914	0.3699
	8 (prp.)	0.3432 ± 0.0157	**0.3302 ± 0.0235**	0.3543	0.3970	0.3769	0.3769	0.4969
Traffic	Overall	**0.0008 ± 0.0004**	0.1750 ± 0.0099	0.0199	0.0425	0.0217	0.0433	0.0709
	1 (geo.)	**0.0001 ± 0.0002**	0.1619 ± 0.0099	0.0133	0.0344	0.0168	0.0302	0.0547
	2 (geo.)	**0.0001 ± 0.0003**	0.1835 ± 0.0101	0.0135	0.0380	0.0180	0.0392	0.0676
	3 (geo.)	**0.0005 ± 0.0007**	0.1819 ± 0.0100	0.0373	0.0647	0.0295	0.0850	0.0989
	4 (geo.)	**0.1354 ± 0.0325**	0.9964 ± 0.043	0.6355	0.5876	0.5669	0.5669	1.3118

Taieb & Koo, 2019), (3) ARIMA-MinT (Wickramasuriya et al., 2019), (4) ARIMA-BottomUp, (5) an ARIMA and (6) Seasonal Naive. We use the implementation of statistical methods available in StatsForecast and HierarchicalForecast libraries (Olivares et al., 2022b; Garza et al., 2022).

3.2 Forecasting Results

As mentioned earlier, we compare the proposed model to the DPMN (Olivares et al., 2023), the HierE2E (Rangapuram et al., 2021), and two ARIMA-based reconciliation methods (Wickramasuriya et al., 2019; Panagiotelis et al., 2023). Following previous work, we report the sCRPS at all levels of the defined hierarchies; see



(a) Learning Objectives

(b) Cross Series MLP

Figure 3: a) The CRPS learning objective shows clear advantages over classic negative log likelihood estimation, while the Factor Model approach shows to be a more flexible approach compared to alternative distributions. b) In the presence of strong correlations within the hierarchy the series cross learning strategy shows significant accuracy improvements.*The Normal and StudentT are non coherent forecast distributions, in contrast to the Gaussian Factor Model and the Poisson Mixture.

Table 3. The `ARIMA` reconciliation results are generated using Olivares et al. (2022b), with confidence interval computed based on 10 independent runs. Results for HierE2E are generated based on three independent runs using hyperparameters tuned by Olivares et al. (2022b). All metrics for `DPMN` are quoted from Olivares et al. (2023) with identical experimental setting on all datasets.

Our model achieves better overall CRPS for the test sets of all three datasets, improving on previous methods by 27.67%, 4.16% and 54.40% on `Favorita`, `Tourism-L` and `Traffic`, respectively, as seen on the *Overall* rows of Table 3. For `Favorita` and `Tourism-Large`, our model achieves better accuracy at almost every single level of the defined hierarchy. On `Traffic`, our model achieves remarkably better results than `DPMN` and `HierE2E`. The accuracy improvements due to its ability to model VAR relationships accurately. It is important to consider that aggregate levels are much smaller in sample size for which we prefer the bottom-level measurements as an indicator of the methods' accuracy. The accuracy gains on sCRPS are mirrored by the accuracy gains on relSE; see Table 4. Finally, we qualitatively show `DeepCoFactor` forecast distributions for a hierarchical structure in the `Favorita` dataset in Appendix E.

3.3 Ablation Studies

To analyze the source of improvements in the `DeepCoFactor`, we performed ablation studies on variants of the `MQCNN` (Wen et al., 2017; Olivares et al., 2022b). We investigate the effects of the network's learning objective, and the effects of leveraging a VAR-like cross-series MLP. For the ablation studies we use a simplified experimental setup over the `Traffic` dataset, where we consider the same forecasting task as the main experiment but we evaluate the sCRPS in the validation set, for 5 randomly initialized neural networks. Details available in Appendix F.

In our ablation study of the learning objective effects, we compared the CRPS-based optimization, as described in Eqn. 14, with the classic negative log-likelihood estimation for the Gaussian factor model introduced in Section 2.1. Additionally, to demonstrate the viability of the factor model, we also compared it with other likelihood-estimated distributions, including Poisson Mixture, Student-T, and univariate Gaussian.

We observed that the CRPS-optimized factor model, improves forecasting accuracy by nearly 60% when compared to the log likelihood optimized factor model.

In our ablation study of the effects of a cross-series MLP that mimics the vector autoregressive model, we compare the `DeepCoFactor` architecture with and without the cross series multilayer perceptron (CrossSeriesMLP) introduced in Eqn. 11. Such a module enables the network to share information of the series in the hierarchy with minimal modifications in the architecture. We observed that the CrossSeriesMLP improved `Traffic`

forecasting accuracy by 66% when compared to variants without it. We attribute the effectiveness of the VAR approach to the presence of Granger-causal relationships in the traffic intersections. The cross series learning approach allowed us to breach the `Traffic` performance gap between `DPMN` and `HierE2E`.

4 Conclusion

This study pioneers the use of factor models to capture correlations among hierarchical series structures, while maintaining forecast coherence. While we focus on parametrizing the multivariate predictive distribution as a Gaussian multivariate factor model, our framework is versatile and can accommodate other distributions that support sample differentiability. This is of special interest for outlier quantiles that cannot be well approximated by Gaussian variables. Exciting future research directions include extending the reparameterization trick to handle discrete distributions, which could further enhance the accuracy of forecast distributions built on this framework. Finally, we have only scratched the surface in exploring different learning objectives. Extensions could involve exploring the energy score (Gneiting & Raftery, 2007), which naturally extends the univariate CRPS objective to a multivariate context. Alternatively, investigating quadratic objectives or other robust learning objective functions could also prove interesting.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL `https://www.tensorflow.org/`. Software available from tensorflow.org.

Marc G Bellemare, Ivo Danihelka, Will Dabney, Shakir Mohamed, Balaji Lakshminarayanan, Stephan Hoyer, and Rémi Munos. The cramer distance as a solution to biased wasserstein gradients. *arXiv preprint arXiv:1705.10743*, 2017.

Souhaib Ben Taieb and Bonsoo Koo. Regularized regression for hierarchical forecasting without unbiasedness conditions. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, pp. 1337–1347, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450362016. doi: 10.1145/3292500.3330976. URL `https://doi.org/10.1145/3292500.3330976`.

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

Souhaib Ben Taieb, James W. Taylor, and Rob J. Hyndman. Coherent probabilistic forecasts for hierarchical time series. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 3348–3357. PMLR, 06–11 Aug 2017b. URL `http://proceedings.mlr.press/v70/taieb17a.html`.

Souhaib Ben Taieb, James W. Taylor, and Rob J Hyndman. Hierarchical probabilistic forecasting of electricity demand with smart meter data. *Journal of the American Statistical Association*, 116:27 – 43, 2020. URL `https://api.semanticscholar.org/CorpusID:43214772`.

David Bolin and Jonas Wallin. Local scale invariance and robustness of proper scoring rules, 2019. URL `https://arxiv.org/abs/1912.05642`.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL `http://github.com/google/jax`.

Byron J. Dangerfield and John S. Morris. Top-down or bottom-up: Aggregate versus disaggregate extrapolations. *International Journal of Forecasting*, 8(2):233–241, 1992. URL `https://www.sciencedirect.com/science/article/pii/016920709290121O`.

A. Das, W. Kong, B. Paria, and R. Sen. Dirichlet proportions model for hierarchically coherent probabilistic forecasting. In Robin J. Evans and Ilya Shpitser (eds.), *Proceedings of the Thirty-Ninth Conference on Uncertainty in Artificial Intelligence*, volume 216 of *Proceedings of Machine Learning Research*, pp. 518–528. PMLR, 31 Jul–04 Aug 2023. URL `https://proceedings.mlr.press/v216/das23b.html`.

Corporación Favorita, inversion, Julia Elliott, and Mark McDonald. Corporación favorita grocery sales forecasting, 2017. URL `https://kaggle.com/competitions/favorita-grocery-sales-forecasting`.

Michael Figurnov, Shakir Mohamed, and Andriy Mnih. Implicit reparameterization gradients. In *Neural Information Processing Systems*, 2018.

Valentin Flunkert, David Salinas, and Jan Gasthaus. Deepar: Probabilistic forecasting with autoregressive recurrent networks. *ArXiv*, abs/1704.04110, 2017.

Federico Garza, Max Mergenthaler Canseco, Cristian Challú, and Kin G. Olivares. StatsForecast: Lightning fast forecasting with statistical and econometric models. PyCon Salt Lake City, Utah, US 2022, 2022. URL `https://github.com/Nixtla/statsforecast`.

Tilmann Gneiting and Matthias Katzfuss. Probabilistic forecasting. *Annual Review of Statistics and Its Application*, 1(1):125–151, 2014. URL https://doi.org/10.1146/annurev-statistics-062713-085831.

Tilmann Gneiting and Adrian E Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American statistical Association*, 102(477):359–378, 2007.

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

Tao Hong, Pierre Pinson, and Shu Fan. Global energy forecasting competition 2012, 2014.

Rob J Hyndman and George Athanasopoulos. *Forecasting: Principles and Practice*. OTexts, Melbourne, Australia, 2018. available at https://otexts.com/fpp2/.

Rob J. Hyndman and Anne B. Koehler. Another look at measures of forecast accuracy. *International Journal of Forecasting*, 22(4):679 – 688, 2006. ISSN 0169-2070. doi: https://doi.org/10.1016/j.ijforecast.2006.03.001. URL http://www.sciencedirect.com/science/article/pii/S0169207006000239.

Rob J. Hyndman, Roman A. Ahmed, George Athanasopoulos, and Han Lin Shang. Optimal combination forecasts for hierarchical time series. *Computational Statistics & Data Analysis*, 55(9):2579–2589, 2011. URL https://www.sciencedirect.com/science/article/pii/S0167947311000971.

Rob J. Hyndman, Alan J. Lee, and Earo Wang. Fast computation of reconciled forecasts for hierarchical and grouped time series. *Comput. Stat. Data Anal.*, 97(C):16–32, may 2016. URL https://doi.org/10.1016/j.csda.2015.11.007.

Martin Jankowiak and Fritz Obermeyer. Pathwise derivatives beyond the reparameterization trick. *ArXiv*, abs/1806.01851, 2018.

Jooyoung Jeon, Anastasios Panagiotelis, and Fotios Petropoulos. Probabilistic forecast reconciliation with applications to wind power and electric load. *European Journal of Operational Research*, 279(2):364–379, 2019.

Harshavardhan Kamarthi, Lingkai Kong, Alexander Rodríguez, Chao Zhang, and B Aditya Prakash. Profhit: Probabilistic robust forecasting for hierarchical time-series. *arXiv preprint arXiv:2206.07940*, 2022.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. *CoRR*, abs/1312.6114, 2013.

Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. M5 accuracy competition: Results, findings, and conclusions. *International Journal of Forecasting*, 38(4):1346–1364, 2022a. URL https://www.sciencedirect.com/science/article/pii/S0169207021001874. Special Issue: M5 competition.

Spyros Makridakis, Evangelos Spiliotis, Vassilios Assimakopoulos, Zhi Chen, Anil Gaba, Ilia Tsetlin, and Robert L. Winkler. The m5 uncertainty competition: Results, findings and conclusions. *International Journal of Forecasting*, 38(4):1365–1385, 2022b. ISSN 0169-2070. doi: https://doi.org/10.1016/j.ijforecast.2021.10.009. URL https://www.sciencedirect.com/science/article/pii/S0169207021001722. Special Issue: M5 competition.

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

Konstantin Mishchenko, Mallory Montgomery, and Federico Vaggi. A self-supervised approach to hierarchical forecasting with applications to groupwise synthetic controls. *ArXiv*, abs/1906.10586, 2019.

Kin G. Olivares, Cristian Challú, Federico Garza, Max Mergenthaler Canseco, and Artur Dubrawski. NeuralForecast: User friendly state-of-the-art neural forecasting models. PyCon Salt Lake City, Utah, US 2022, 2022a. URL https://github.com/Nixtla/neuralforecast.

Kin G. Olivares, Federico Garza, David Luo, Cristian Challú, Max Mergenthaler, Souhaib Ben Taieb, Shanika L. Wickramasuriya, and Artur Dubrawski. HierarchicalForecast: A reference framework for hierarchical forecasting in python. *Work in progress paper, submitted to Journal of Machine Learning Research.*, abs/2207.03517, 2022b. URL https://arxiv.org/abs/2207.03517.

Kin G. Olivares, O. Nganba Meetei, Ruijun Ma, Rohan Reddy, Mengfei Cao, and Lee Dicker. Probabilistic hierarchical forecasting with deep poisson mixtures. *International Journal of Forecasting*, 2023. URL https://www.sciencedirect.com/science/article/pii/S0169207023000432.

Anastasios Panagiotelis, Puwasala Gamakumara, George Athanasopoulos, and Rob J Hyndman. Probabilistic forecast reconciliation: Properties, evaluation and score optimisation. *European Journal of Operational Research*, 306(2):693–706, 2023.

Biswajit Paria, Rajat Sen, Amr Ahmed, and Abhimanyu Das. Hierarchically Regularized Deep Forecasting. In *Submitted to Proceedings of the 39th International Conference on Machine Learning*. PMLR. Working Paper version available at arXiv:2106.07630, 2021.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pp. 8024–8035. Curran Associates, Inc., 2019. URL http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

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

Francisco J. R. Ruiz, Michalis K. Titsias, and David M. Blei. The generalized reparameterization gradient. In *NIPS*, 2016.

Steven R. Vitullo. Disaggregating time series data for energy consumption by aggregate and individual customer. *Department of Electrical and Computer Engineering, Ph. D. Dissertation.*, 2011.

Yuyang Wang, Alex Smola, Danielle Maddix, Jan Gasthaus, Dean Foster, and Tim Januschowski. Deep factors for forecasting. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 6607–6617. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/wang19k.html.

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

Shanika L. Wickramasuriya. Probabilistic forecast reconciliation under the Gaussian framework. *Accepted at Journal of Business and Economic Statistics*, 2023.

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

Yuan Yao, Lorenzo Rosasco, and Caponneto Andrea. On early stopping in gradient descent learning. *Constructive Approximation*, 26(2):289–315, 2007. URL https://doi.org/10.1007/s00365-006-0663-2.

A Multivariate Factor Model Coherence and Covariance

A.1 Coherent Aggregation Properties

The coherence of `DeepCoFactor` is a special case of the bootstrap sample reconciliation technique (Panagiotelis et al., 2023), as explored by Olivares et al. (2023).

Lemma A.1. *Let $(\Omega_{[b]}, \mathcal{F}_{[b]}, \mathbb{P}_{[b]})$ be a probabilistic forecast space, with $\mathcal{F}_{[b]}$ a σ-algebra on $\Omega_{[b]}$. If a forecast distribution $\mathbb{P}_{[i]}$ assigns a zero probability to sets that don't contain coherent forecasts, it defines a coherent probabilistic forecast space $(\Omega_{[i]}, \mathcal{F}_{[i]}, \mathbb{P}_{[i]})$ with $\Omega_{[i]} = \mathbf{S}_{[i][b]}(\Omega_{[b]})$.*

$$\mathbb{P}_{[a]}\left(\mathbf{y}_{[a]} \notin \mathbf{A}_{[a][b]}(\mathcal{B}) \mid \mathcal{B}\right) = 0 \implies \mathbb{P}_{[i]}\left(\mathbf{S}_{[i][b]}(\mathcal{B})\right) = \mathbb{P}_{[b]}\left(\mathcal{B}\right) \quad \forall \mathcal{B} \in \mathcal{F}_{[b]}. \tag{18}$$

Proof. We note the following:

$$\begin{aligned}\mathbb{P}_{[i]}\left(\mathbf{S}_{[i][b]}(\mathcal{B})\right) &= \mathbb{P}_{[i]}\left(\begin{bmatrix}\mathbf{A}_{[a][b]}\\ \mathbf{I}_{[b][b]}\end{bmatrix}(\mathcal{B})\right) = \mathbb{P}_{[i]}\left(\{\begin{bmatrix}\mathbf{A}_{[a][b]}(\mathcal{B})\\ \mathbb{R}^{N_b}\end{bmatrix}\} \cap \{\begin{bmatrix}\mathbb{R}^{N_a}\\ \mathcal{B}\end{bmatrix}\}\right)\\ &= \mathbb{P}_{[a]}\left(\mathbf{A}_{[a][b]}(\mathcal{B}) \mid \mathcal{B}\right)\mathbb{P}_{[b]}\left(\mathcal{B}\right) = \left(1 - \mathbb{P}_{[a]}\left(\mathbf{y}_{[a]} \notin \mathbf{A}_{[a][b]}(\mathcal{B}) \mid \mathcal{B}\right)\right) \times \mathbb{P}_{[b]}\left(\mathcal{B}\right) = \mathbb{P}_{[b]}\left(\mathcal{B}\right).\end{aligned}$$

The first equality is the image of a set $\mathcal{B} \in \Omega_{[b]}$ corresponding to the constraints matrix transformation, the second equality defines the spanned space as a subspace intersection of the aggregate series and the bottom series, the third equality uses the conditional probability multiplication rule, the final equality uses the zero probability assumption.

By construction of the samples of our model $\tilde{\mathbf{y}}_{[i]} = \mathbf{S}_{[i][b]}\left(\hat{\mathbf{y}}_{[b]}\right)_{+}$ and $\tilde{\mathbf{y}}_{[a]} = \mathbf{A}_{[a][b]}\left(\hat{\mathbf{y}}_{[b]}\right)_{+}$, satisfying the assumptions of the lemma and proving the coherence of our approach.

□

A.2 Covariance Structure

Here we prove the covariance structure of our factor model introduced in Section 2.1.

Lemma A.2. *Let our factor model be defined by*

$$\hat{\mathbf{y}}_{[b],\eta,t} = \hat{\boldsymbol{\mu}}_{[b],\eta,t} + \mathrm{Diag}(\hat{\boldsymbol{\sigma}}_{[b],\eta,t})\mathbf{z}_{[b],\eta,t} + \hat{\mathbf{F}}_{[b][k],\eta,t}\boldsymbol{\epsilon}_{[k],\eta,t}, \qquad \eta = 1, \cdots, N_h, \tag{19}$$

with independent factors $\mathbf{z}_{[b],\eta} \sim \mathcal{N}(\mathbf{0}_{[b]}, \mathbf{I}_{[b][b]})$, and $\boldsymbol{\epsilon}_{[k],\eta} \sim \mathcal{N}(\mathbf{0}_{[k]}, \mathbf{I}_{[k][k]})$, its covariance satisfies

$$\mathrm{Cov}\left(\hat{\mathbf{y}}_{[b],\eta,t}\right) = \mathrm{Diag}(\hat{\boldsymbol{\sigma}}^{2}_{[b],\eta,t}) + \hat{\mathbf{F}}_{[b][k],\eta,t}\hat{\mathbf{F}}^{\top}_{[b][k],\eta,t}. \tag{20}$$

Proof. First, we observe that

$$\begin{aligned}\mathrm{Cov}\left(\hat{\mathbf{y}}_{[b],\eta,t}, \hat{\mathbf{y}}_{[b],\eta,t}\right) &= \mathrm{Cov}\left(\mathrm{Diag}(\hat{\boldsymbol{\sigma}}_{[b],\eta,t})\mathbf{z}_{[b],\eta,t}, \mathrm{Diag}(\hat{\boldsymbol{\sigma}}_{[b],\eta,t})\mathbf{z}_{[b],\eta,t}\right)\\ &+ \quad 2\mathrm{Cov}\left(\mathrm{Diag}(\hat{\boldsymbol{\sigma}}_{[b],\eta,t})\mathbf{z}_{[b],\eta,t}, \hat{\mathbf{F}}_{[b][k],\eta,t}\boldsymbol{\epsilon}_{[k],\eta,t}\right)\\ &+ \quad \mathrm{Cov}\left(\hat{\mathbf{F}}_{[b][k],\eta,t}\boldsymbol{\epsilon}_{[k],\eta,t}, \hat{\mathbf{F}}_{[b][k],\eta,t}\boldsymbol{\epsilon}_{[k],\eta,t}\right).\end{aligned} \tag{21}$$

By bilinearity of covariance and independence of the sampled factors, it follows that

$$\mathrm{Cov}\left(\hat{\mathbf{y}}_{[b],\eta,t}, \hat{\mathbf{y}}_{[b],\eta,t}\right) = \mathrm{Diag}(\hat{\boldsymbol{\sigma}}_{[b],\eta,t})\mathrm{Cov}\left(\mathbf{z}_{[b],\eta,t}, \mathbf{z}_{[b],\eta,t}\right)\mathrm{Diag}(\hat{\boldsymbol{\sigma}}_{[b],\eta,t})^{\intercal} + \hat{\mathbf{F}}_{[b][k],\eta,t}\mathrm{Cov}\left(\boldsymbol{\epsilon}_{[k],\eta,t}, \boldsymbol{\epsilon}_{[k],\eta,t}\right)\hat{\mathbf{F}}^{\intercal}_{[b][k],\eta,t}.$$

We conclude that

$$\mathrm{Cov}\left(\hat{\mathbf{y}}_{[b],\eta,t}, \hat{\mathbf{y}}_{[b],\eta,t}\right) = \mathrm{Diag}(\hat{\boldsymbol{\sigma}}^{2}_{[b],\eta,t}) + \hat{\mathbf{F}}_{[b][k],\eta,t}\hat{\mathbf{F}}^{\intercal}_{[b][k],\eta,t}. \tag{22}$$

□

B Code Script for Sampling

```
    def sample(self, distr_args, window_size, num_samples=None):
        """
        **Parameters**
        `distr_args`: Forecast Distribution arguments.
        `window_size`: int=1, for reconciliation reshapes in sample method.
        `num_samples`: int=500, number of samples for the empirical quantiles.

        **Returns**
        `samples`: tensor, shape [B,H,`num_samples`].
        `quantiles`: tensor, empirical quantiles defined by `levels`.
        """
        means, factor_loading, stds = distr_args
        collapsed_batch, H, _ = means.size()

        # [collapsed_batch,H]:=[B*N*Ws,H,F] -> [B,N,Ws,H,F]
        factor_loading = factor_loading.reshape(
            (-1, self.n_series, window_size, H, self.n_factors)
        ).contiguous()
        factor_loading = torch.einsum(
            "iv,bvwhf->biwhf", self.SP, factor_loading
        )  # v = i but i represents reconciled forecasts and v base forecast

        means = means.reshape(-1, self.n_series, window_size, H, 1).contiguous()
        stds = stds.reshape(-1, self.n_series, window_size, H, 1).contiguous()

        # Factor model loads factor for covariance Diag(stds) + F F^t -> (SPF)(SPF^t)
        hidden_factor = Normal(
            loc=torch.zeros(
                (factor_loading.shape[0], window_size, H, self.n_factors)
            ).to(means.device),
            scale=1.0)
        sample_factors = hidden_factor.rsample(sample_shape=(self.num_samples,))
        sample_factors = sample_factors.permute(
            (1, 2, 3, 4, 0)
        ).contiguous()  # [n_items, window_size, H, F,num_samples]

        sample_loaded_factors = torch.einsum(
            "bvwhf,bwhfn->bvwhn", factor_loading, sample_factors)

        # [n_items, n_base, window_size, H, num_samples]
        sample_loaded_means = means + sample_loaded_factors

        # Sample Normal
        normal = Normal(loc=torch.zeros_like(sample_loaded_means), scale=1.0)
        samples = normal.rsample()
        samples = F.relu(sample_loaded_means + stds * samples)

        samples = torch.einsum("iv,bvwhn->biwhn", self.SP, samples)
        samples = samples.reshape(collapsed_batch, H, self.num_samples).contiguous()

        # Compute quantiles and mean
        quantiles_device = self.quantiles.to(means.device)
        quants = torch.quantile(input=samples, q=quantiles_device, dim=-1)
        quants = quants.permute((1, 2, 0))  # [Q,B,H] -> [B,H,Q]
        sample_mean = torch.mean(samples, dim=-1, keepdim=True)
        return samples, sample_mean, quants
```

Figure 4: PyTorch function for sampling from our Gaussian Factor model. Note that the factor samples are shared across all bottom-level distributions. The samples are differentiable with regard to the function inputs. We can easily adapt this function to sample from other distributions.

Table 5: *Deep Coherent Factor Model Neural Network* (`DeepCoFactor`) architecture hyperparameters.
* SGD batch selection as well as model dimensions follow mostly GPU memory constraints.

Parameter	Notation	Considered Values		
		Favorita	Tourism-L	Traffic
Activation Function.	-	ReLU	ReLU	ReLU
Temporal Convolution Dilations.	N_{ck}	[1,2,4,8,16,32]	[1,2,3,6,12]	[1,7,14,28]
Temporal Convolution Channel Size.	N_p	30	30	10
Future Encoder Dimension.	N_f	50	50	20
Static Encoder Dimension.	N_s	50	20	5
Horizon Agnostic Decoder Dimensions.	N_{ag}	20	20	20
Horizon Specific Decoder Dimensions.	N_{sp}	5	5	5
Factor Model Components.	N_k	5	10	10
Cross Series MLP Hidden Size.	N_k	5	50	200
SGD Batch Size.	-	4	1	1
SGD Effective Batch Size.	-	744	555	207
SGD Max steps.	-	80e3	2e3	2e3
Learning Rate.	-	5e-4	5e-4	5e-4
Random Seeds.	-	$\{1, 2, 3, 4, 42\}$	$\{1, 2, 3, 4, 42\}$	$\{1, 2, 3, 4, 42\}$
GPU Training Configuration.	-	1 x NVIDIA V100	1 x NVIDIA V100	1 x NVIDIA V100

C Training Methodology and Hyperparameters

Here we complement and extend the description of our method in Section 2.

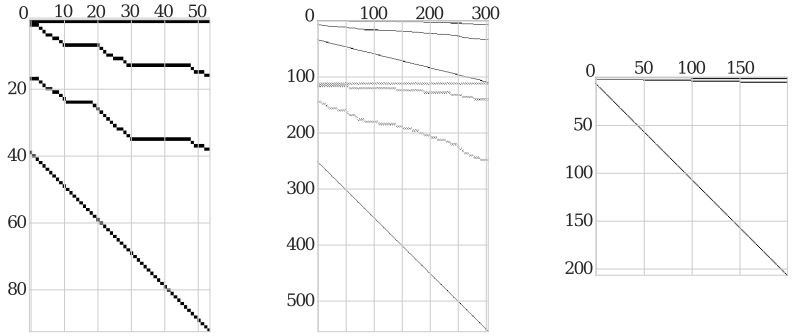
To avoid information leakage we perform ablation studies in the validation set preceding the test set, where we explored variants of the probabilistic method, as well as its optimization. We report these ablation studies in Appendix F. For each dataset, given the prediction horizon h, the test set is composed of the last h time-steps. The validation set is composed of the h time-steps preceding the test set time range. The training set is composed of all dates previous to the validation time-range. When reporting final accuracy results of our model on test set, we used the settings that perform the best in validation set.

We tune minimally the architecture and its parameters varying only its size and the convolution kernel filters to match the seasonalities present in each dataset. For the `Favorita` dataset we use dilations of $[1, 2, 4, 8, 16, 32]$ to match weekly and monthly seasonalities, for the `Tourism-L` dataset we use dilations of $[1, 2, 3, 6, 12]$ to match the monthly and yearly seasonalities, for the `Traffic` dataset we use dilations of $[1, 7, 14, 28]$ as multiples of 7 to match the weekly seasonalities.

The selection of the number of factors follows mostly the memory constraints of the GPU, as the effective batch size implied by our probabilistic model grows rapidly as a function of the multivariate series. In the `Favorita` dataset more factors are likely to continue to improve accuracy but with the tradeoff of the computational speed. Similarly the Cross series MLP hidden size is selected following the GPU memory constraints.

We share a learning rate of 5e-4 constant across the three datasets, which shows that the method is reasonably robust across different forecasting tasks. During the optimization of the networks we use adaptive moments stochastic gradient descent (Kingma & Ba, 2014) with early stopping (Yao et al., 2007) guided by the sCRPS signal measured in the validation set. We use a learning rate scheduler that decimates the learning rate four times during the optimization (SGD Maxteps/4), to ensure the convergence of the optimization.

The `DeepCoFactor` model is implemented using Pytorch (Paszke et al., 2019), with the NeuralForecast library framework (Olivares et al., 2022a).



(a) Favorita (b) Tourism-L (c) Traffic

Figure 5: Visualization of the hierarchical constraints of the empirical evaluation datasets. (a) `Favorita` classifies its grocery sales by store, city, state, and country levels. (b) `Tourism-L` categorizes its 555 regional visit series based on travel purpose, zones, states, and country-level geographical aggregations. (c) `Traffic` organizes the occupancy series of 200 highways into quarters, halves, and totals.

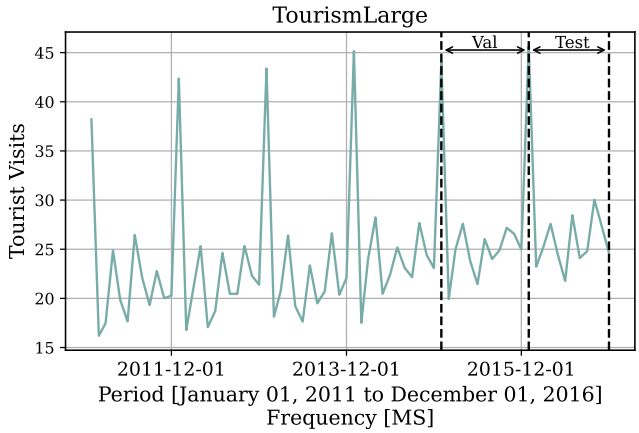


Figure 6: `Tourism-L` dataset partition into train, validation, and test sets used in our experiments. All datasets use the last horizon window as defined in Table 2 (marked by the second dotted line), and the previous window preceding the test set as validation (between the first and second dotted lines). Validation provides the signal for hyperparameter selection and the ablation studies.

D Dataset Details

`Favorita` The Favorita dataset (Favorita et al., 2017) contains grocery sales of the Ecuatorian Corporación Favorita in $N = 54$ stores. We perform geographical aggregation of the sales at the store, city, state and national levels, following (Olivares et al., 2023). This yields a total of $M = 94$ aggregates. Concerning features, we use past unit sales and number of transactions as historical data. In the `Favorita` dataset we include item perishability static information, geographic state dummy variables, and for the historic exogenous features and future exogenous features we use promotions and day of the week.

`Tourism-L` The `Tourism-L` dataset (Wickramasuriya et al., 2019) represents visits to Australia, at a monthly frequency, between January 1998 and December 2016. We use 2015 for validation, and 2016 for testing, and all previous years for training. The dataset contains 228 monthly observations. For each month, we have the

number of visits to each of Australia's 78 regions, which are aggregated to the zone, state and national level, and for each of four purposes of travel. These two dimensions of aggregation total $N = 304$ leaf entities (a region-purpose pair), with a total of $M = 555$ series in the hierarchy. We pre-process the data to include static features we use purpose of travel as well as state dummies, for the historical information we use month dummies, and for the future exogenous we use month and a seasonal naive anchor forecast that helps greatly to account for the series seasonality.

`Traffic` The Traffic dataset (Ben Taieb & Koo, 2019) contains aggregates of daily freeway occupancy rates for 200 sampled (out of 963) car lanes in the San Francisco Bay Area between January 2008 to March 2009. We follow the aggregation defined in Ben Taieb & Koo (2019). We note that this scheme aggregates occupancy rates by adding them up. There are three aggregated levels: four groups of 50 car lanes, two groups of 100 car lanes, and an overall group of 200 lanes. Each group was chosen randomly in Ben Taieb & Koo (2019); we keep the same grouping. We follow previous experiments in the literature (Ben Taieb & Koo, 2019; Rangapuram et al., 2021; Olivares et al., 2023), and split the dataset into training, validation, and test dataset of size 120, 120 and 126. In Table 3, we report accuracy numbers for the last date of 126 dates only, following the experimentation setting in (Rangapuram et al., 2021; Olivares et al., 2023). In the `Traffic` dataset we use geographic node dummies, that identify the quarter and halves belonging, for the historic and future exogenous we use Saturday and Sunday dummies as well as the distance to the next Saturday.

E Forecast Distributions Visualization

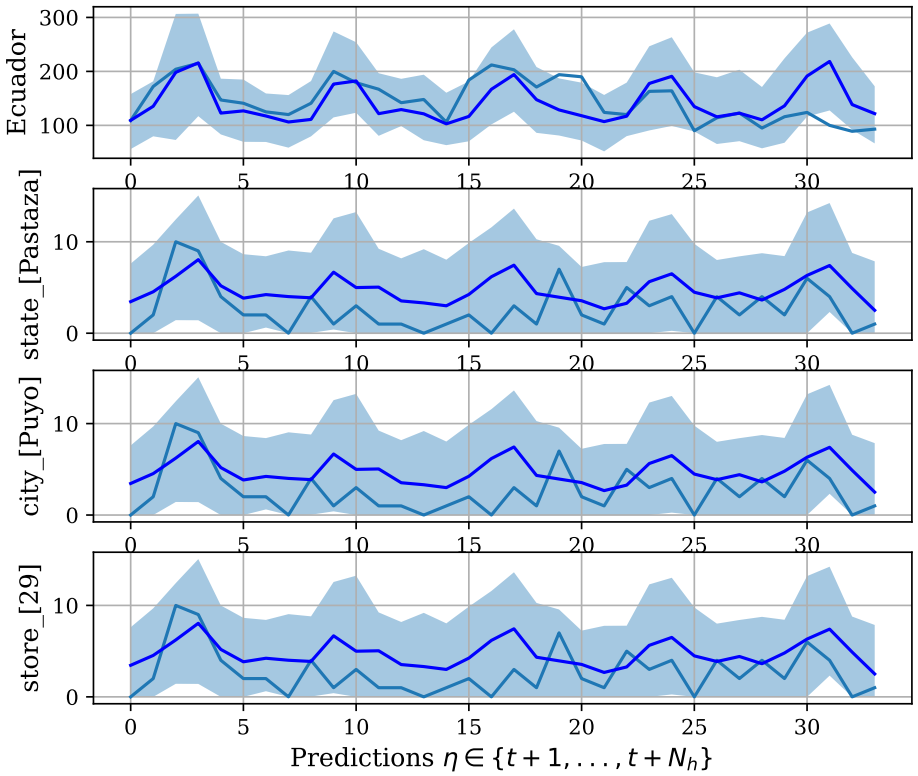


Figure 7: `DeepCoFactor` forecast distributions on a hierarchically linked time series of the `Favorita` dataset. We show the forecasted demand for a grocery item on a store of the Puyo City, in the State of Pastaza and the whole country demand in the top row. Forecast distributions show the 90% forecast intervals in light blue, and the forecasted median in dark blue. The clipped Normal distribution achieves non-negative predictions and a point mass at zero.

Table 6: Ablation study on the `Traffic` dataset. empirical evaluation of probabilistic coherent forecasts. Mean *scaled continuous ranked probability score* (sCRPS) averaged over 5 runs, at each aggregation level, the best result is highlighted (lower measurements are preferred).
* The Normal and StudentT are non coherent forecast distributions, in contrast to the Factor Model and the Poisson Mixture.

Level	FactorModel+crps	FactorModel+nll	PoissonMixture	StudentT*	Normal*
Overall	0.0259±0.0060	0.0879±0.1136	0.0827±0.1408	0.0600±0.0367	0.0734±0.0253
Total	0.0023±0.0022	0.0623±0.1335	0.0667±0.1817	0.0280±0.0283	0.0556±0.0327
Halves	0.0028±0.0018	0.0640±0.1318	0.0765±0.2030	0.0296±0.0300	0.0510±0.0316
Quarters	0.0043±0.0015	0.0644±0.1313	0.0742±0.1782	0.0301±0.0294	0.0450±0.0334
Lanes	0.0942±0.0195	0.1608±0.0615	0.1136±0.0056	0.1523±0.0752	0.1422±0.0554

Table 7: Ablation study on the `Traffic` dataset, empirical evaluation of probabilistic coherent forecasts. Mean *scaled continuous ranked probability score* (sCRPS) averaged over 5 runs, at each aggregation level, the best result is highlighted (lower measurements are preferred).

Level	CrossSeriesMLP	¬CrossSeriesMLP
Overall	0.0242±0.0035	0.0613±0.0257
Total	0.0035±0.0015	0.0432±0.0296
Halves	0.0048±0.0030	0.0437±0.0289
Quarters	0.0041±0.0022	0.0440±0.0281
Lanes	0.0905±0.0084	0.1145±0.0174

F Ablation Studies Details

To analyze the sources of improvements in our model, we conducted ablation studies on variants of the `DeepCoFactor/MQCNN/DPMN` (Wen et al., 2017; Olivares et al., 2022b). We utilized a simplified setup on the `Traffic` dataset, focusing on the same forecasting task as the main experiment. We evaluated the sCRPS from Eqn. 16 on the validation set across 5 randomly initialized neural networks. The experiments use the same hyperparameters as reported in Table 5, and vary a single characteristic of interest of the network and measuring its effects on the validation dataset.

In our first ablation study we explore the effects of the learning objective alternatives to Eqn. 14, for this purpose we augment the `DeepCoFactor` architecture with different distribution outputs including Normal, Student-T and Poisson Mixture distributions (Olivares et al., 2023). In addition we also compare with our own Factor model approach, as we can see in Table 6 and Figure 3, the CRPS optimization of the Factor Model improves upon the negative log likelihood by 60 percent the mean sCRPS in the validation set. The difference is highly driven by outlier runs, but it is expected as the CRPS objective has much convenient numerical properties, starting by its bounded gradients. Another important note is that in the literature factor model estimation is usually done using evidence lower bound optimization, as the latent factors can quickly land in subpar local minima. In this ablation study we show the CRPS offers a reliable alternative to both.

In our second ablation study we explore the effects of the impact of including vector autorregresive relationships of the hierarchy through the CrossSeriesMLP module described in Eqn. 11. In the experiment we train a `DeepCoFactor` with and without the module on the `Traffic` dataset. As we can see in Table 6 and Figure 3 using the CrossSeriesMLP improves sCRPS upon the alternative (without) by 66 percent. The technique breaches the gap to the `HierE2E` (Rangapuram et al., 2021), that previously outperformed all alternative methods by over 50 percent. It is important to note that `HierE2E` is also a VAR approach. We attribute the improvements to the heavy presence of Granger causal relationships between the traffic lanes, as they carry lag historical information that influence each other.