Dirichlet Proportions Model for Hierarchically Coherent Probabilistic Forecasting

Abhimanyu Das[1] **Weihao Kong**[1] **Biswajit Paria**[2] **Rajat Sen**[1]

[1]Google Research, Mountain View
[2]Google, Mountain View

Abstract

Probabilistic, hierarchically coherent forecasting is a key problem in many practical forecasting applications – the goal is to obtain coherent probabilistic predictions for a large number of time series arranged in a pre-specified tree hierarchy. In this paper, we present an end-to-end deep probabilistic model for hierarchical forecasting that is motivated by a classical top-down strategy. It jointly learns the distribution of the root time series, and the (dirichlet) proportions according to which each parent time-series is split among its children at any point in time. The resulting forecasts are naturally coherent, and provide probabilistic predictions over all time series in the hierarchy. We experiment on several public datasets and demonstrate significant improvements of up to 26% on most datasets compared to state-of-the-art baselines. Finally, we also provide theoretical justification for the superiority of our top-down approach compared to the more traditional bottom-up modeling.

1 INTRODUCTION

A central problem in multivariate forecasting is the need to forecast a large group of time series arranged in a natural hierarchical structure, such that time series at higher levels of the hierarchy are aggregates of time series at lower levels. For example, hierarchical time series are common in retail forecasting applications [FMK19], where the time series may capture retail sales of a company at different granularities such as item-level sales, category-level sales, and department-level sales. In electricity demand forecasting [VEC15], the time series may correspond to electricity consumption at different granularities, starting with individual households, which could be progressively grouped into city-level, and then state-level consumption time-series. The hierarchical structure among the time series is usually represented as a tree, with leaf-level nodes corresponding to time series at the finest granularity, while higher-level nodes represent coarser-granularities and are obtained by aggregating the values from its children nodes.

Since businesses usually require forecasts at various different granularities, the goal is to obtain accurate forecasts for time series at every level of the hierarchy. Furthermore, to ensure decision-making at different hierarchical levels are aligned, it is essential to generate predictions that are *coherent* [HAAS11] with respect to the hierarchy, that is, the forecasts of a parent time-series should be equal to the sum of forecasts of its children time-series. For example, the sum of the sales predictions for each shoe style should equal the sales prediction for the shoe category [1]. Finally, to facilitate better decision making, there is an increasing shift towards probabilistic forecasting [BRGS10, GK14]; that is, the forecasting model should quantify the uncertainty in the output and produce a probability distribution, instead of a single point estimate, for predictions.

In this paper, we address the problem of obtaining coherent probabilistic forecasts for large-scale hierarchical time series. While there has been a plethora of work on multivariate forecasting, there is significantly limited research on multivariate forecasting for hierarchical time series that satisfy the requirements of both hierarchical coherence **and** probabilistic predictions.

There are numerous recent works on deep neural network-based multivariate forecasting [SFGJ20, OCCB19, RSG$^+$18, BRF$^+$20, SYD19, OCM$^+$22], including probabilistic multivariate forecasting [SBSC$^+$19, RSS$^+$21] and even graph neural network(GNN)-based models for forecasting on time series with graph-structure correlations [BYL$^+$20, CWD$^+$20, YYZ17, LYSL17]. However,

[1]Note that this is a non-trivial constraint. For example, generating independent predictions for each time series in the hierarchy using a standard multivariate forecasting model does not guarantee coherent predictions.

Accepted for the 39th Conference on Uncertainty in Artificial Intelligence (UAI 2023).

none of these works ensure coherent predictions for hierarchical time series.

On the other hand, several papers specifically address hierarchically-coherent forecasting [HLW16, TTH17, VEC15, HLW16, BTK19, WAH$^+$15, WTH20, MPS21, AHTB19], based on the idea of *reconciliation*. This involves a two-stage process where the first stage generates independent (possibly incoherent) univariate base forecasts, and is followed by a second *reconciliation* stage that adjusts these forecasts using the hierarchy structure, to finally obtain coherent predictions. These approaches are usually disadvantaged in terms of using the hierarchical constraints only as a post-processing step, and not during generation of the base forecasts. Furthermore, with the exception of [TTH17], which is a two-stage reconciliation-based model for coherent probabilistic hierarchical forecasting, most of these approaches cannot directly handle probabilistic forecasts.

More recently, there has been some work([RWB$^+$21, HDG21]) that propose single-stage, end-to-end deep neural architectures that directly produce hierarchically-coherent (or approximately coherent) probabilistic forecasts without a need for a post-processing step.

In this paper, we present an alternate approach to end-to-end deep probabilistic forecasting for hierarchical time series, motivated by a classical method that has not received much recent attention: top-down forecasting. The basic idea is to first model the top-level forecast in the hierarchy tree, along with the ratios or proportions according to how the top level forecasts should be distributed among the children time-series in the hierarchy. The resulting predictions are naturally coherent. Early top-down approaches were non-probabilistic, and were rather simplistic in terms of modeling the proportions; for example, by separately modeling the top-level forecast, and then deriving the proportions from historical averages [GS90], or from independently generated (incoherent) forecasts of each time-series from another model [AAH09]. In this paper, we showcase how modeling both the top-level forecast and the proportions jointly through a single-stage deep probabilistic model can obtain state-of-the-art results for probabilistic, hierarchically-coherent forecasts.

Crucially, our proposed model (and indeed all top-down approaches for forecasting) relies on the intuition that the top level time series in a hierarchy is usually much less noisy and less sparse, and hence much easier to predict. Furthermore, it might be easier to predict proportions (that are akin to scale-free normalized time-series) at the lower level nodes than the actual time series themselves.

Our approach involves learning a single end-to-end deep model to jointly model "families", consisting of a parent and its children timeseries, and predict both the parent timeseries and the proportions along which it is disaggregated among its children. We use a Dirichlet distribution [OR64] to model the distribution of proportions for each parent-children family in the hierarchy. The parameters of the Dirichlet distribution for each family is obtained from an MLP (Multi Layer Perceptron) based encoder-decoder model with multi-headed self-attention [VSP$^+$17], that is jointly learnt from the whole dataset.

We validate our model against state-of-the art probabilistic hierarchical forecasting baselines on six public datasets, and demonstrate significant gains using our approach, outperforming the baselines on most datasets with improvements of up to 26% in terms of CRPS scores.

Additionally, we theoretically analyze the advantage of the top-down approach (over a bottom-up approach) in a simplified regression setting for hierarchical prediction, and thereby provide theoretical justification for our top-down model. Specifically, we prove that for a 2-level hierarchy of d-dimensional linear regression with a single root node and K children nodes, the excess risk of the bottom-up approach is $\min(K, d)$ time bigger than the one of the top-down approach in the worst case. This validates our intuition that it is easier to predict proportions than the actual values.

2 BACKGROUND

Hierarchical forecasting is a multivariate forecasting problem, where we are given a set of N univariate time-series (each having T time points) that satisfy linear aggregation constraints specified by a predefined hierarchy. More specifically, the data can be represented by a matrix $\boldsymbol{Y} \in \mathbb{R}^{T \times N}$, where $\boldsymbol{y}^{(i)}$ denotes the T values of the i-th time series, $\boldsymbol{y}_t$ denotes the values of all N time series at time t, and $y_{t,i}$ the value of the i-th time series at time t. We will assume that $y_{i,t} \geq 0$, which is usually the case in all retail demand forecasting datasets (and is indeed the case in all public hierarchical forecasting benchmarks [WAH$^+$15]). We compactly denote the H-step history of $\boldsymbol{Y}$ by $\boldsymbol{Y}_{\mathcal{H}} = [\boldsymbol{y}_{t-H}, \cdots, \boldsymbol{y}_{t-1}]^\top \in \mathbb{R}^{H \times N}$ and the H-step history of $\boldsymbol{y}^{(i)}$ by $\boldsymbol{y}^{(i)}_{\mathcal{H}} = [\boldsymbol{y}^{(i)}_{t-H}, \cdots, \boldsymbol{y}^{(i)}_{t-1}] \in \mathbb{R}^{H}$. Similarly we can define the F-step future as $\boldsymbol{Y}_{\mathcal{F}} = [\boldsymbol{y}_t, \cdots, \boldsymbol{y}_{t+F-1}]^\top \in \mathbb{R}^{F \times N}$. We use the $\hat{\cdot}$ notation to denote predicted values, and the $\cdot^\top$ notation to denote the transpose. We denote the matrix of external covariates like holidays etc by $\boldsymbol{X} \in \mathbb{R}^{T \times D}$, where the t-th row denotes the D-dimensional feature vector at the t-th time step. For simplicity, we assume that the features are shared across all time series[2]. We define $\boldsymbol{X}_{\mathcal{H}}$ and $\boldsymbol{X}_{\mathcal{F}}$ similarly. We will also use numpy tensor notation i.e $\boldsymbol{X}[i : j, r : c]$ would denote the sub-matrix in rows $\{i, i+1, \cdots, j-1\}$ and columns $\{r, r+1, \cdots c-1\}$. Further, using : would denote selecting all rows or columns depending on the axis.

[2]Note that our modeling can handle both shared and time-series specific covariates in practice.

Hierarchy. The N time series are arranged in a tree hierarchy, with m leaf time-series, and $k = N - m$ non-leaf (or aggregated) time-series that can be expressed as the sum of its children time-series, or alternatively, the sum of the leaf time series in its sub-tree. Let $\boldsymbol{b}_t \in \mathbb{R}^m$ be the values of the m leaf time series at time t, and $\boldsymbol{r}_t \in \mathbb{R}^k$ be the values of the k aggregated time series at time t. The hierarchy is encoded as an aggregation matrix $\boldsymbol{R} \in \{0,1\}^{k \times m}$, where an entry R_{ij} is equal to 1 if the i-th aggregated time series is an ancestor of the j-th leaf time series in the hierarchy tree, and 0 otherwise. We therefore have the aggregation constraints $\boldsymbol{r}_t = \boldsymbol{R}\boldsymbol{b}_t$ or $\boldsymbol{y}_t = [\boldsymbol{r}_t^\top \ \ \boldsymbol{b}_t^\top]^\top = \boldsymbol{S}\boldsymbol{b}_t$ where $\boldsymbol{S}^T = [\boldsymbol{R}^\top | \boldsymbol{I}_m]$. Here, $\boldsymbol{I}_m$ is the $m \times m$ identity matrix. Such a hierarchical structure is ubiquitous in multivariate time series from many domains such as retail, traffic, etc, as discussed earlier. We provide an example tree with its $\boldsymbol{S}$ matrix in Figure 1. We can extend this equation to the matrix $\boldsymbol{Y} \in \mathbb{R}^{T \times N}$. Let $\boldsymbol{B} := [\boldsymbol{b}_1; \cdots; \boldsymbol{b}_m]^\top$ be the corresponding leaf time-series values arranged in a $T \times m$ matrix. Then coherence property of $\boldsymbol{Y}$ implies that $\boldsymbol{Y}^\top = \boldsymbol{S}\boldsymbol{B}^\top$. Note that we will use $\boldsymbol{B}_\mathcal{F}$ to denote the leaf-time series matrix corresponding to the future time-series in $\boldsymbol{Y}_\mathcal{F}$.

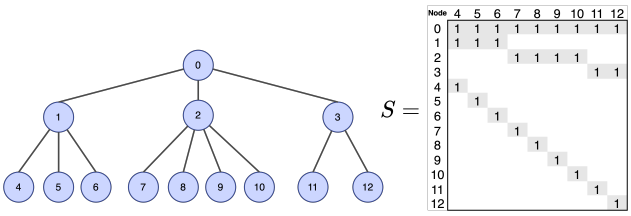

Figure 1: An example of a hierarchy and its corresponding $\boldsymbol{S}$ matrix. The rows and columns of the matrix are indexed by the corresponding nodes for easier interpretation. The empty cells of the matrix are zeros, and hence omitted from the figure.

Coherency. Clearly, an important property of hierarchical forecasting is that the forecasts also satisfy the deterministic hierarchical constraints $\hat{\boldsymbol{Y}}_\mathcal{F}^\top = \boldsymbol{S}\hat{\boldsymbol{B}}_\mathcal{F}^\top$. This is known as the *coherence* property which has been used in several prior works [HA18, TTH17]. Imposing the coherence property makes sense since the ground truth data $\boldsymbol{Y}$ is coherent by construction. Coherence is also critical for consistent decision making at different granularities of the hierarchy.

Our *objective* is to accurately predict the distribution of the future values $\hat{\boldsymbol{Y}}_\mathcal{F} \sim \hat{f}(\boldsymbol{Y}_\mathcal{F})$ conditioned on the history such that any relaization $\hat{\boldsymbol{Y}} \in \mathbb{R}^{F \times N}$ from the predicted distribution $\hat{f}(\boldsymbol{Y}_\mathcal{F})$ satisfies the deterministic coherence property above. In particular, $\hat{f}$ denotes the density function (multi-variate) of the future values $\boldsymbol{Y}_\mathcal{F}$ conditioned on the historical data $\boldsymbol{X}_\mathcal{H}, \boldsymbol{Y}_\mathcal{H}$ and the future features $\boldsymbol{X}_\mathcal{F}$. We omit the conditioning from the expression for readability.

The related work can be broadly classified into (i) Coherent point forecasting that includes but is not limited to approaches like [HAAS11, VEC15, WAH19]; (ii) Coherent probabilistic forecasting that includes among others [TTH17, RWB$^+$21, OMM$^+$21, PGAH23, ZAC22] and (iii) Approximately coherent methods like [MMV19, Gle20, HDG21, HHG21, PSAD21]. We include a detailed discussion of related literature in Appendix A.

3 THE MODEL

The basic input data-structure to our model is a *family* $(p, \mathcal{L}(p))$, where p is a parent node in the hierarchical tree (any non-leaf node) and $\mathcal{L}(p)$ are its children nodes. In Figure 1, the set of nodes $(2, [7, 8, 9, 10])$ denotes a family.

Our main contribution is a shared proportions model that takes in the past data points of a family and predicts (i) the future proportions of the children i.e the fractions by which the parent time-series diaggregates into the children time-series in the future (ii) the future values of the parent. The model is designed to capture dependencies among the children through appropriate applications of attention layers and also propagate information between the parent and the children. We train a single shared global proportions model for all the families in the tree.

Modeling Proportions. Before we describe the model for forecasting the proportions, we need to formally define the children proportions. For a family $(p, \mathcal{L}(p))$ let us define the proportions,

$$a_{s,c} = \frac{y_{s,c}}{\sum_{j \in \mathcal{L}(p)} y_{s,j}}, \quad \text{for all } s \in [T], c \in \mathcal{L}(p). \tag{1}$$

The matrix $\boldsymbol{A}(p) \in \mathbb{R}^{T \times C}$ denotes the proportions of the children over time, where $C := |\mathcal{L}(p)|$. We will drop the p in braces when it is clear from context that we are dealing with a particular family $(p, \mathcal{L}(p))$. As in Section 2, we use $\boldsymbol{A}_\mathcal{H}$ and $\boldsymbol{A}_\mathcal{F}$ to denote the historical and future proportions. Also, $\boldsymbol{a}_\mathcal{H}^{(i)}$ will denote the historical proportions for child $i \in \mathcal{L}(p)$ and a similar definition holds for $\boldsymbol{a}_\mathcal{F}^{(i)}$.

We are interested in predicting the distribution of $\boldsymbol{A}_\mathcal{F}$ given historical proportions $\boldsymbol{A}_\mathcal{H}$, the parent's history $\boldsymbol{y}_\mathcal{H}^{(p)}$ and covariates $\boldsymbol{X}$. Note that for any $s \in [T]$, the sth row of $\boldsymbol{A}(p)$, $\boldsymbol{a}_s \in \Delta^{C-1}$ (denotes the $(C-1)$-dimensional simplex). Therefore, our predicted distribution should also be a distribution over the simplex for each row. Hence, we use the Dirichlet [OR64] family to model the output distribution for each row of the predicted proportions, as detailed later.

We choose the simplest possible architectural building blocks for the task at hand: (i) we use a simple MLP (Multi-Layer Perceptron) encoder-decoder model for capturing the temporal dependencies in the proportions of a child (independently of other children) (ii) we use multi-headed attention [VSP$^+$17] to capture dependencies among the children of the family.

Encoder: We first form an encoding depending on the past for each child,

$$\boldsymbol{e}_i \leftarrow \texttt{enc}\left(\boldsymbol{a}_{\mathcal{H}}^{(i)}, \boldsymbol{y}_{\mathcal{H}}^{(p)}, \boldsymbol{X}\right)$$

where $\boldsymbol{e}_i \in \mathbb{R}^{d_E}$ and d_E is the encoding dimension. We represent all the children embeddings in the matrix $\boldsymbol{E} \in \mathbb{R}^{d_E \times C}$ such that the i-th column of $\boldsymbol{E}$ is the encoding $\boldsymbol{e}_i$ of the i-th child. The MLP encoder, $\texttt{enc}(\cdot)$ is applied independently for each child. Each child's embedding can also depend on the past of the parent $\boldsymbol{y}_{\mathcal{H}}^{(p)}$ and the covariates $\boldsymbol{X}$; thereby drawing information from higher level time-series.

Attention: Then we apply multi-headed attention to mix the encoded information across the children. The input to the attention is $\boldsymbol{E}' = [\boldsymbol{E}; \boldsymbol{1}] \in \mathbb{R}^{d_E \times (C+1)}$ i.e a dummy column added to the children embeddings (the value of that column will become clear later when we dicuss the parent prediction module). The attention layer is denoted by,

$$\boldsymbol{M}' \leftarrow \texttt{MultiHeadAtt}_{g,l}\left(\boldsymbol{E}'\right), \tag{2}$$

where g denotes the number of attention heads and l denotes the number of attention layers. Each attention layer is followed by a fully connected layer with ReLU activation and also equipped with a residual connection (similar to the original model in [VSP+17]). Note that the attention is only applied across the second dimension i.e across the children. The resulting $\boldsymbol{M}'$ is in $\mathbb{R}^{d_A \times (C+1)}$ where d_A is the value dimension in the attention layers. Let $\boldsymbol{M} = \boldsymbol{M}'[:, 1:C]$.

Decoder: Now that we have mixed the encoded information among the children, we are ready to decode to obtain the predicted distribution of future proportions of the children. The decoding follows the equations:

$$\begin{aligned}
\boldsymbol{D} &= \texttt{dec}(\boldsymbol{M}), \ \text{(Output shape: } d_D * F \times C), \\
\boldsymbol{D}_{\mathcal{F}} &= \text{(Reshape of } \boldsymbol{D} \text{ into } d_D \times F \times C), \\
\hat{a}_{s,c} &= \texttt{dec}_{\mathcal{F}}(\boldsymbol{D}_{\mathcal{F}}[:, s, c], \boldsymbol{X}_{\mathcal{F}}[s, :]), \ \text{(Output shape: 1)} \\
\hat{\boldsymbol{A}}_{\mathcal{F}} &\text{ is a matrix s.t } \hat{\boldsymbol{A}}_{\mathcal{F}}[s, c] = \exp(a_{s,c}).
\end{aligned}$$

In the first equation $\texttt{dec}(\cdot)$ is a MLP decoder with output dimensions $d_D * F$ that is applied on the first axis of $\boldsymbol{M}$. Then the output is reshaped into $\boldsymbol{D}_{\mathcal{F}}$ such that we have a decoded feature of length d_D for all F future time-points and all C children in the family. Then we apply another MLP layer with $\texttt{dec}_{\mathcal{F}}(\cdot)$ with output dimension 1. For each future time-point $s \in [F]$ and children $c \in C$, $\texttt{dec}_{\mathcal{F}}$ is applied independently on the concatenation of $\boldsymbol{D}_{\mathcal{F}}[:, s, c]$ and $\boldsymbol{X}_{\mathcal{F}}[s, :]$ to produce the output parameter $\hat{a}_{s,c}$. Intuitively, this final MLP combines the information in the d_D dimensional decoded feature for the children c at future time s along with the future covariates at that time-point to produce the final output. The output for all future time-points and all children are then collected in the matrix $\hat{\boldsymbol{A}}_{\mathcal{F}} \in \mathbb{R}^{F \times C}$ after passing through the exp. function to ensure positivity.

Loss Function: We would like to output the predicted distribution of the proportions of the children in the family for all future time-steps. Our loss function is designed such that the output from the preceding decoder step $\hat{\boldsymbol{A}}_{\mathcal{F}}$ can represent such a distribution. Recall that the predicted proportions distributions $\hat{f}(\boldsymbol{A}_{\mathcal{F}})$ have to be over the simplex Δ^{C-1} for each time-point. Therefore we model it by the Dirichlet distribution. In fact our final model output $\hat{\boldsymbol{A}}_{\mathcal{F}}$ represents the parameters of predictive Dirichlet distributions. Specifically, we minimize the loss

$$\ell_c(\boldsymbol{A}_{\mathcal{F}}, \hat{\boldsymbol{A}}_{\mathcal{F}}) = -\frac{1}{F}\sum_{s=1}^{F} \texttt{DirLL}\left(\boldsymbol{a}_s + \epsilon; \hat{\mathbf{a}}_s\right), \tag{3}$$

where $\boldsymbol{a}_s$ is the proportion of the children nodes in the family at time s as defined in Equation (1). $\texttt{DirLL}(\boldsymbol{a}; \vec{\beta})$ denotes the log-likelihood of Dirichlet distribution for target $\boldsymbol{a}$ and parameters $\vec{\beta}$.

$$\texttt{DirLL}(\boldsymbol{a}; \vec{\beta}) := \sum_i (\beta_i - 1)\log(a_i) - \log B(\vec{\beta}), \tag{4}$$

where $B(\vec{\beta})$ is the normalization constant. In Eq. (3), we add a small ϵ to avoid undefined values when the target proportion for some children are zero. Here, $B(\vec{\beta}) = \prod_{i=1}^{C}\Gamma(\beta_i)/\Gamma(\sum_i \beta_i)$ where $\Gamma(.)$ is the well-known Gamma function that is differentiable. In practice, we use Tensorflow Probability [DLT+17] to optimize the above loss function.

Modeling the Parent. The remaining task is to predict the future values of the parent node in a family. Recall that the output of the attention layer $\boldsymbol{M}'$ in Eq. (2) has an extra dimension in the second axis. We will use the output in that dimension as an input to the decoder that predicts the future of the parent. This allows us to distill historical information from the children that might also be useful for predicting the future values of the parent. The decoding for the parent prediction comprises of the following equations:

$$\begin{aligned}
\boldsymbol{P} &= \texttt{dec}^{(p)}(\boldsymbol{M}[:, -1], \boldsymbol{y}_{\mathcal{H}}^{(p)}) \ \text{(Output shape: } d_D * F), \\
\boldsymbol{P}_{\mathcal{F}} &= \text{(Reshape of } \boldsymbol{P} \text{ into } F \times d_D), \\
\hat{\boldsymbol{p}}_s &= \texttt{dec}_{\mathcal{F}}^{(p)}(\boldsymbol{P}_{\mathcal{F}}[s, :], \boldsymbol{X}_{\mathcal{F}}[s, :]), \ \text{(Output shape: 2)} \\
\hat{\boldsymbol{P}}_{\mathcal{F}} &\text{ is a matrix s.t } \hat{\boldsymbol{p}}[s, :] = \hat{\boldsymbol{p}}_s.
\end{aligned}$$

In the first equation we use the MLP decoder to map the last dimension of the attention encoding from the children along with the past of the parent, to the future decoding of shape $d_D * F$. The decoding is then reshaped to have shape $F \times d_D$. The final parent decoding layer is an MLP $\texttt{dec}_{\mathcal{F}}^{(p)}$ with output dimension 2 that maps each future time-step's decoded features $\boldsymbol{P}_{\mathcal{F}}[s, :]$ along with the covariates at that time-step to the final predicted parameters for that time-step.

Loss Function. Since we are interested in probabilistic forecasting, we would like to predict the distribution of the

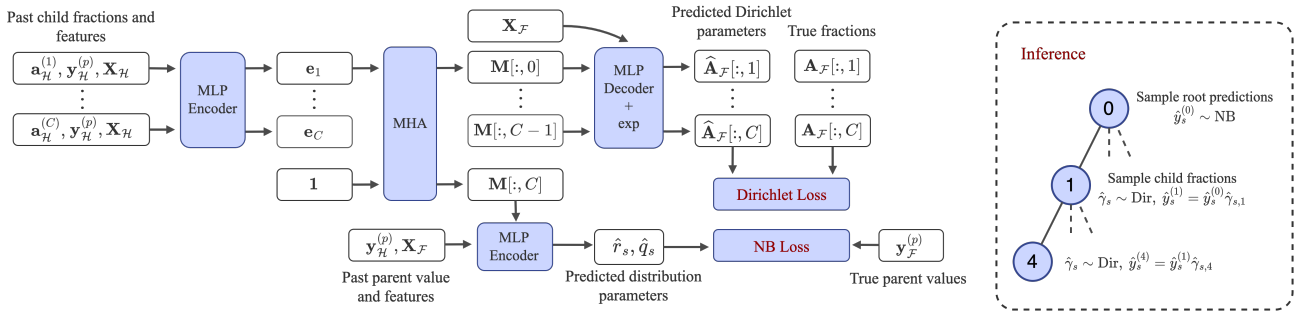

Figure 2: We provide a complete description of the training architecture. MHA denotes multi-head self attention layers. The indices $\{1, \cdots, C\}$ are used to denote the indices of the children of the parent node p. During inference, the top level predictions are first sampled from the negative binomial distribution with the predicted parameters $\hat{q}_s, \hat{r}_s$. The children prediction are then successively sampled by first sampling the fractions $\hat{\gamma}_s$ from the Dirichlet distribution and then multiplying with the parent sample.

parent's future values. Therefore we will map the output parameters in each future time step ($\hat{\boldsymbol{p}}_s$ for time-step $s \in [F]$) to the parameters of a negative-binomial distribution. The p.m.f of negative binomial distribution with total count $r > 0$ and success probability $q \in [0, 1]$ is given by,

$$\texttt{NBL}(k; r, q) = \frac{\Gamma(k+r)}{\Gamma(k+1)\Gamma(r)}(1-q)^k q^r \text{ for } k = 1, 2, \cdots \tag{5}$$

We first map the predicted parameters from the parent decoder $\hat{\boldsymbol{p}}_s$ to the distribution parameters $\hat{r}_s, \hat{q}_s$ using link functions (designed to keep (r, q) in valid ranges):

$$\hat{r}_s = \sigma(\hat{\boldsymbol{p}}_s[0])$$
$$\hat{q}_s = \frac{1}{1 + \sigma(\hat{\boldsymbol{p}}_s[1])},$$

where $\sigma(x) := (2 + x - |x|)/(2 - x + |x|)$. Thus σ maps $\mathbb{R} \to \mathbb{R}^+$. Finally we minimize the negative log likelihood of observing the actual future values of the parents under the negative binomial distribution formed by the predicted parameters,

$$\ell_p(\boldsymbol{y}_{\mathcal{F}}^{(p)}, \hat{\boldsymbol{P}}_{\mathcal{F}}) = -\frac{1}{F}\sum_{s=1}^{F} \log\left(\texttt{NBL}(y_{p,s}; \hat{r}_s, \hat{q}_s)\right). \tag{6}$$

We use the negative binomial distribution since a most hierarchical forecasting datasets contain count data that is positive. In such cases, the negative binomial loss has been used with great success in the context of time-series forecasting [ADSS21, SFGJ20].

The final loss is the summation of the children loss in Eq. (3) and the parent loss in Eq. (6). We provide a full illustration of our model in Figure 2. We refer to our model as DirProp.

Training. We train our model with mini-batch gradient descent where each batch corresponds to different history and future time-intervals of the *same family*. For example, if the time batch-size is b and we are given a family $(p, \mathcal{L}(p))$ the input proportions that are fed into the model are of shape $b \times H \times C$ and the output distribution parameters for the Dirchlet loss are of shape $b \times F \times C$, where $C = |\mathcal{L}(p)|$. Note that we only need to load all the time-series of a given family into a batch.

Note that the number of parameters of the model as well as the size of the mini-batch does not need to scale with the number of time-series in the tree, unlike prior works like [RWB+21].

Inference. At inference we have to output a representation of the predicted cumulative distribution $\hat{F}(\boldsymbol{Y}_{\mathcal{F}})$ such that the samples are reconciled as in Section 2. For ease of illustration, we will demonstrate the procedure for one time point $s \in \{1, \cdots, F\}$.

We first sample $\hat{y}_s^{(p)}$ from the predictive distribution of the parent for the family containing the root node. For instance, in the tree of Figure 1, this family would be $(0, [1, 2, 3])$. Then for each family $(p, \mathcal{L}(p))$ in the tree we generate a sample from the Dirchlet distribution with parameters $\hat{\boldsymbol{A}}_{\mathcal{F}}[s, :]$ that represents a sample of the predicted children proportions for that family. The proportion samples and the root sample can be combined to form a reconciled forecast sample $\hat{\boldsymbol{y}}_s$. We can generate many such samples and then take empirical statistics to form the predictive distribution $\hat{f}(\boldsymbol{Y}_s)$, which is by definition reconciled.

4 THEORETICAL JUSTIFICATION FOR THE TOP-DOWN APPROACH

In this section, we theoretically analyze the advantage of the top-down approach over the bottom-up approach for hierarchical prediction in a simplified setting. Again, the intuition is that the root level time series is much less noisy and hence much easier to predict, and it is easier to predict proportions at the children nodes than the actual values themselves. As a result, combining the root level prediction with the propor-

tions prediction actually yields a much better prediction for the children nodes. Consider a 2-level hierarchy of linear regression problem consisting of a single root node (indexed by 0) with K children. For each time step $t \in [n]$, a global covariate $\mathbf{x}_t \in \mathbb{R}^d$ is independently drawn from a Gaussian distribution $\mathbf{x}_t \sim \mathcal{N}(0, \Sigma)$, and the value for each node is defined as follows:

- The value of the root node at time t is $y_{t,0} = \theta_0^\top \mathbf{x}_t + \eta_t$, where $\eta_t \in \mathbb{R}$ is independent of $\mathbf{x}_t$, and satisfies $\mathbb{E}[\eta_t] = 0, \text{Var}[\eta_t] = \sigma^2$.
- A random K-dimensional vector $\mathbf{a}_t \in \mathbb{R}^K$ is independently drawn from distribution P such that $\mathbb{E}[a_{t,i}] = p_i$ and $\text{Var}[a_{t,i}] = s_i$, where $a_{t,i}$ is the i-th coordinate of $\mathbf{a}_t$. For the i-th child node, the value of the node is defined as $a_{t,i} \cdot y_{t,0}$.

Notice that for the i-th child node, $\mathbb{E}[y_{t,i}|\mathbf{x}_t] = p_i \theta_0^\top \mathbf{x}_t$, and therefore the i-th child node follows from a linear model with coefficients $\theta_i := p_i \theta_0$.

Now we describe the bottom-up approach and top-down approach and analyze the expected excess risk of them respectively. In the bottom-up approach, we learn a separate linear predictor for each child node seprately. For the i-th child node, the ordinary least square (OLS) estimator is

$$\hat{\theta}_i^{\mathbf{b}} = \left(\sum_{t=1}^{n} \mathbf{x}_t \mathbf{x}_t^\top \right)^{-1} \sum_{t=1}^{n} \mathbf{x}_t y_{t,i},$$

and the prediction of the root node is simply the summation of all the children nodes.

In the top-down approach, a single OLS linear predictor is first learnt for the root node:

$$\hat{\theta}_0^{\mathbf{t}} = \left(\sum_{t=1}^{n} \mathbf{x}_t \mathbf{x}_t^\top \right)^{-1} \sum_{t=1}^{n} \mathbf{x}_t y_{t,0}$$

Then the proportion coefficient $\hat{p}_i, i \in [K]$ is learnt for each node separately as $\hat{p}_i = \frac{1}{n} \sum_{t=1}^{n} y_{t,i}/y_{t,0}$ and the final linear predictor for the ith child is $\hat{\theta}_i^{\mathbf{t}} = \hat{p}_i \hat{\theta}_0^{\mathbf{t}}$. Let us define the excess risk of an estimator $\hat{\theta}_i$ as $r(\hat{\theta}_i) = (\hat{\theta}_i - \theta_i)^\top \Sigma (\hat{\theta}_i - \theta_i)$. The expected excess risk of both approaches are summarized in the following theorem, proved in Appendix B.1.

Theorem 4.1 (Expected excess risk comparison between top-down and bottom-up approaches)**.** *The total expected excess risk of the bottom-up approach for all the children nodes satisfies*

$$\sum_{i=1}^{K} \mathbb{E}[r(\hat{\theta}_i^{\mathbf{b}})] \geq \sum_{i=1}^{K} (s_i + p_i^2) \frac{d}{n-d-1} \sigma^2,$$

and the total expected excess risk of the top-down approach satisfies

$$\sum_{i=1}^{K} \mathbb{E}[r(\hat{\theta}_i^{\mathbf{t}})] = \frac{\sum_{i=1}^{K} s_i}{n} \theta_0^\top \Sigma \theta_0 + \left(\frac{\sum_{i=1}^{K} s_i}{n} + \sum_{i=1}^{K} p_i^2 \right) \frac{d}{n-d-1} \sigma^2,$$

Applying the theorem to the case where the proportion distribution $\mathbf{a}_t$ is drawn from a uniform Dirichlet distribution, we show the excess risk of the traditional bottom-up approach is $\min(K, d)$ times bigger than our proposed top-down approach in the following corollary. A proof of the corollary can be found in Appendix B.2

Corollary 4.2. *Assuming that for each time-step $t \in [n]$, the proportion coefficient $\mathbf{a}_t$ is drawn from a K-dimensional Dirichlet distribution Dir(α) with $\alpha_i = \frac{1}{K}$ for all $i \in [K]$ and $\theta_0^\top \Sigma \theta_0 = \sigma^2$, then*

$$\frac{\mathbb{E}[\sum_{i=1}^{K} r(\hat{\theta}_i^{\mathbf{b}})]}{\mathbb{E}[\sum_{i=1}^{K} r(\hat{\theta}_i^{\mathbf{t}})]} = \Omega(\min(K, d)).$$

In Section 5, we show that even the basic topdown approach analyzed in this section outperforms several state of the art methods, thus conforming to our theoretical justification. Our learnt top-down model is a further improvement over the historical fractions.

5 EXPERIMENTS

We implement our model in Tensorflow [ABC+16][3] and compare our approach with state of the art models for coherent probabilistic forecasting on 6 hierarchical forecasting datasets. We now describe the datasets along with the corresponding forecasting setups.

Datasets. We experiment with two retail forecasting datasets, M5 [M520] and Favorita [Fav17]. These are our largest datasets with 3060 and 4471 total time-series in their respective hierarchices. For both these datasets, we use the product hierarchy i.e each leaf time-series corresponds to the sales of an item aggregated across the stores. The other datasets include: Tourism-L [Tou19, WAH19] which is a dataset consisting of tourist count data; Labour [oS20], consisting of monthly employment data; Traffic [Cut11], consisting of daily occupancy rates of cars on freeways; and Wiki2 [Wik17], consisting of daily views on Wikipedia articles. For Tourism-L we benchmark on both the (Geo)graphic and (Trav)el history based hierarchy. More details about the dataset and the features used for each dataset can be found in Appendix C and Table 6.

[3] Our implementation can be found here.

Table 1: We present the normalized metrics WAPE / NRMSE across all levels. We report normalized metrics so that they can be averaged to produce the mean column. All the numbers are averaged over 5 runs. The numbers in bold represent the statistically significant best performances in each column, while the italized ones represent the second best.

Dataset	Method	L0	L1	L2	L3	Mean
M5	DirProp	**0.0404 / 0.0487**	**0.0518 / 0.0773**	**0.0662 / 0.1061**	**0.3238 / 0.6818**	**0.1205 / 0.2285**
	Fedformer-Base (incoherent)	*0.0585 / 0.0715*	*0.0659 / 0.1036*	*0.0718 / 0.1219*	0.3453 / 0.7065	*0.1354 / 0.2509*
	Fedformer-BU	0.0641 / 0.0781	0.0739 / 0.1167	0.0795 / 0.1167	0.3453 / 0.7065	0.1407 / 0.2603
	Fedformer-TD	*0.0585 / 0.0715*	0.0680 / 0.1067	0.0738 / 0.1258	*0.3443 / 0.699*	*0.1361 / 0.2507*
	Fedformer-ERM	0.0979 / 0.1145	0.102 / 0.1462	0.1101 / 0.1755	0.3914 / 0.8100	0.1753 / 0.3110
	Fedformer-MinT	0.0619 / 0.0749	0.0714 / 0.1131	0.0772 / 0.136	0.3462 / 0.7058	0.1392 / 0.2575
Favorita	DirProp	**0.0485 / 0.0614**	**0.0948 / 0.2336**	*0.1513 / 0.4504*	**0.3039 / 1.0925**	**0.1496 / 0.4595**
	Fedformer-Base (incoherent)	*0.0667 / 0.0869*	*0.1004 / 0.2562*	0.1904 / 0.4447	0.3875 / 1.1264	0.1863 / 0.4786
	Fedformer-BU	0.0887 / 0.101	0.1114 / 0.2848	0.1605 / 0.4551	0.3875 / 1.1264	0.187 / 0.4918
	Fedformer-TD	*0.0667 / 0.0869*	*0.098 / 0.2492*	0.2482 / 2.4286	0.5343 / 7.3585	0.2368 / 2.5308
	Fedformer-ERM	0.0746 / 0.0967	*0.0994 / 0.2721*	**0.1385 / 0.4397**	*0.3116 / 1.1243*	*0.156 / 0.4832*
	Fedformer-MinT	*0.0667 / 0.0869*	0.1035 / 0.2541	0.1777 / 0.4403	0.4089 / 1.1309	0.1892 / 0.478

Note that for the sake of reproducibility, except for the additional M5 and Favorita datasets, the datasets and experimental setup are largely identical to that in [RWB$^+$21] with an increased horizon for traffic and wiki datasets. In [RWB$^+$21], the prediction window for the latter two datasets were chosen to be only 1 time-step which is extremely small; moreover on traffic the prediction window only includes the day Dec 31st which is atypical especially because the dataset includes only an year of daily data. Therefore we decided to increase the validation and test size to 7. In all datasets the last F time-steps form the test window while the preceding F time-steps form the validation window – the same convention was followed in [RWB$^+$21].

Benefits of end to end hierarchical modeling. Before comparing our method with state-of-the-art probabilistic coherent forecasting baselines on the benchmark datasets, we would like to showcase the benefit of end-to-end hierarchical modeling as a whole. To that end, we first choose a simpler task of accurate point forecasting on our two largest datasets, Favorita and M5.

In our baselines, as a base forecaster, we choose a recently published strong multivariate point forecasting method, FEDformer [ZMW$^+$22], that uses a frequency-enhanced transformer (along with other techniques like separate modeling of seasonality and trend) to achieve state-of-the-art results in several long-horizon forecasting tasks. Then we reconcile these base forecasts using popular reconciliation techniques like Bottom-Up [HAAS11], Top-Down [HAAS11], ERM [BTK19], MinT [WTH20] to yield the corresponding baselines FEDformer-BU, FEDformer-TD, FEDformer-ERM and FEDformer-MinT. Lastly we also report the metrics for the incoherent base forecasts dubbed FEDformer-Base. We use the open source package Nixtla [OGL$^+$22] for the reconciliation and the FEDformer repository for the base forecasts.

We present the WAPE / NRMSE metrics (defined in Appendix C) for the baseline forecasts and our p50 forecasts in Table 1. Our end to end coherent method achieves better performance across all levels compared to the base forecasts FEDformer-Base, even though we use a relatively simple architecture. Post-hoc reconciliation seems to help in some cases. For example, FEDformer-ERM has better performance on the Favorita dataset than the base forecasts, but even that falls short of our model (except in L2 of Favorita). This suggests that using coherence as an inductive bias during training might be important in propagating higher level signals to leaf levels. We provide details about all our hyperparameters in Appendix E.

Probabilistic Forecasting. Now that we have seen the benefits of end-to-end hierarchical modeling, we are ready to present our main empirical results for probabilistic hierarchical forecasting. We compare our models to state-of-the-art strictly coherent probabilistic forecasting baselines. The first two baselines capture dependencies using the tree structure during generating the initial probabilistic forecasts even before reconciliation: (i) Hier-E2E [RWB$^+$21] is an end-to-end deep-learning approach for coherent probabilistic forecasts. This method by design produces coherent probabilistic forecasts. (ii) PERMBU [TTH17] is a copula based approach for producing probabilistic hierarchical forecasts. The copula is used to capture dependencies among each family and the the samples are reconciled using well-known reconciliation methods like MinT and BottomUP (BU). We report the best numbers between PERMBU-MinT

Table 2: Normalized CRPS scores for M5 and Favorita. We average the deep learning based methods over 5 independent runs. The rest of the methods had very little variance. We report the corresponding standard error and only bold numbers that are the statistically significantly better than the rest. The second best numbers in each column are italicized.

M5	L0	L1	L2	L3	Mean
DirProp	**0.0379** $\pm$ 0.0014	**0.0422** $\pm$ 0.0004	**0.0536** $\pm$ 0.0023	**0.2543** $\pm$ 0.0067	**0.0970** $\pm$ 0.0013
Hier-E2E	0.1129 $\pm$ 0.0008	0.1106 $\pm$ 0.0008	0.1167 $\pm$ 0.0010	0.2940 $\pm$ 0.0012	0.1586 $\pm$ 0.0005
PERMBU	0.0639	0.0673	0.0737	0.2978	0.1257
Best of Nixtla (AutoARIMA-TD)	*0.0599*	*0.0643*	*0.0713*	*0.2808*	*0.1191*
Favorita	**L0**	**L1**	**L2**	**L3**	**Mean**
DirProp	**0.0430** $\pm$ 0.0024	**0.0709** $\pm$ 0.0016	**0.1132** $\pm$ 0.0017	**0.2446** $\pm$ 0.0023	**0.1179** $\pm$ 0.0018
Hier-E2E	0.0955 $\pm$ 0.0009	0.1211 $\pm$ 0.0018	0.1648 $\pm$ 0.0039	0.3305 $\pm$ 0.0060	0.1780 $\pm$ 0.0028
PERMBU	*0.0561*	0.8279	0.6142	*0.3184*	0.4541
Best of Nixtla (AutoARIMA-BU)	*0.0563*	**0.0697**	**0.1119**	*0.3190*	*0.1392*

Table 3: Normalized CRPS scores averaged over all levels for all remaining datasets introduced in Sec 5. The full set of level-wise metrics can be found in Appendix D. We report the corresponding standard error and only bold numbers that are the statistically significantly better than the rest. The second best numbers in each column are italicized.

Mean metrics	Labour	Traffic	Wiki2	Tourism-L
DirProp	**0.0250** $\pm$ 0.0015	**0.0526** $\pm$ 0.0028	**0.2706** $\pm$ 0.0048	**0.1407**
Hier-E2E	*0.0340* $\pm$ 0.0088	**0.0506** $\pm$ 0.0011	*0.2769* $\pm$ 0.004	0.1520
PERMBU	0.0393	0.1019	0.5033	0.2518
Best of Nixtla	*0.0346* (ERM)	*0.0757* (TD)	0.3631 (TD)	*0.1474* (MinT)

and PERMBU-BU. (iii) For the sake of completeness we also include post-hoc reconciliation baselines. We use the Nixtla package [OGL+22] that produces base probabilistic forecasts from AutoARIMA and then uses MinT, Bottom-Up (BU), TopDown (TD), and ERM reconciliation using a reverse engineered empirical covariance matrix to provide probabilistic forecasts. (Note that we cannot use Fedformer for base forecasts here, since it does not generate probabilistic forecasts.) In the interest of space, we only provide the numbers for the best performing Nixtla method in Table 2, while providing the detailed numbers in the Appendix.

Evaluation. We evaluate forecasting accuracy using the continuous ranked probability score (CRPS). The CRPS is minimized when the predicted quantiles match the true data distribution [GR07]. This is the standard metric used to benchmark probabilistic forecasting in numerous papers [RSG+18, RWB+21, TTH17]. Similar to [RWB+21], we also normalize the CRPS scores at each level, by the absolute sum of the true values of all the nodes of that level. We mathematically define the CRPS score in Appendix D.

We present level-wise performance of all methods on M5 and Favorita, as well as mean performance on the other datasets (the full level-wise metrics on all datasets can be seen in Appendix D). In these tables, we highlight in bold numbers that are statistically significantly better than the rest. The second best numbers in each column are italicized. The deep learning based methods are averaged over 5 runs while other methods had very little variance.

M5: We see that overall in all columns, DirProp performs much better than all the baselines (around 18% better than the best baseline (AutoARIMA-TD) on the mean). Interestingly, even a simple top-down baseline like AutoARIMA-TD outperforms recent state-of-the-art models like Hier-E2E and PERMBU, attesting to the power of top-down approaches. We hypothesize that Hier-E2E does not work well on these larger datasets because the DeepVAR model needs to be applied to thousands of time-series, which leads to a prohibitive size of the fully connected input layer and a hard joint optimization problem.

Favorita: As in the previous results, even in Favorita, DirProp outperforms the other models by a large margin, resulting in a 15% better mean performance than the best baseline. Interestingly again, a simple AutoARIMA-BU model outperforms both Hier-E2E and PERMBU.

Other Datasets: Table 3 presents mean CRPS scores on Labor, Traffic, Tourism and Wiki2 datasets. In Traffic, DirProp is within statistical error of the best baseline, but in the other datasets, DirProp comfortably outperforms all baselines. In three of the four smaller datasets, Hier-E2E performs better than the other baselines, which suggest that Hier-E2E works reasonably well on smaller datasets compared to reconciliation-based methods (though DirProp is still significantly better on Labor, Wiki2, and Tourism by 26%, 2% and 4.5% respectively. We provide more detailed results for all these datasets in Appendix D.

Table 4: We provide an ablation study in the Favorita dataset. We only provide the mean forecast for the sake of brevity but our original model outperforms the ablated baselines on all levels.

Favorita Ablation	Mean
DirProp	**0.1179** $\pm$ 0.0018
DirProp - No Attention	0.1340 $\pm$ 0.0028
DirProp Root + historical fractions	0.1436 $\pm$ 0.0010

Ablation. In Table 4, we study the role of various components in our model on the Favorita dataset. We first remove the attention layers after the encoder and see a 13% drop in mean metric. Thus mixing the information among the children in important. We also use historical static fractions in combination to the root (L0) predictions from our model. It can be seen that our learnt proportions model outperforms historical proportions.

6 CONCLUSION

In this paper, we proposed a probabilistic top-down based hierarchical forecasting approach, that obtains coherent, probabilistic forecasts without the need for a separate reconciliation stage. Our approach is built around a novel deep-learning model for learning the distribution of proportions according to which a parent time series is disaggregated into its children time series. We show in empirical evaluation on several public datasets, that our model obtains state-of-the-art results compared to previous methods.

For future work, we plan to explore extending our approach to handle more complex hierarchical structural constraints, beyond trees. We would also like to note that currently our theoretical justification only applies to learning historical proportions; it would be interesting to extend it to predicted future proportions.

References

[AAH09] George Athanasopoulos, Roman A Ahmed, and Rob J Hyndman. Hierarchical forecasts for australian domestic tourism. *Int. J. Forecasting*, 25(1):146–166, 2009.

[ABBS+20] Alexander Alexandrov, Konstantinos Benidis, Michael Bohlke-Schneider, Valentin Flunkert, Jan Gasthaus, Tim Januschowski, Danielle C. Maddix, Syama Rangapuram, David Salinas, Jasper Schulz, Lorenzo Stella, Ali Caner Türkmen, and Yuyang Wang. Gluonts: Probabilistic and neural time series modeling in python. *J. Mach. Learn. Res.*, 2020.

[ABC+16] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th USENIX symposium on operating systems design and implementation (OSDI 16)*, pages 265–283, 2016.

[ADSS21] Pranjal Awasthi, Abhimanyu Das, Rajat Sen, and Ananda Theertha Suresh. On the benefits of maximum likelihood estimation for regression and forecasting. In *International Conference on Learning Representations*, 2021.

[AGP+20] George Athanasopoulos, Puwasala Gamakumara, Anastasios Panagiotelis, Rob J Hyndman, and Mohamed Affan. Hierarchical forecasting. In *Macroeconomic forecasting in the era of big data*, pages 689–719. Springer, 2020.

[AHTB19] Mahdi Abolghasemi, Rob J Hyndman, Garth Tarr, and Christoph Bergmeir. Machine learning applications in time series hierarchical forecasting. *arXiv preprint arXiv:1912.00370*, 2019.

[BRF+20] Konstantinos Benidis, Syama Sundar Rangapuram, Valentin Flunkert, Bernie Wang, Danielle Maddix, Caner Turkmen, Jan Gasthaus, Michael Bohlke-Schneider, David Salinas, Lorenzo Stella, et al. Neural forecasting: Introduction and literature overview. *arXiv preprint arXiv:2004.10240*, 2020.

[BRGS10] Veronica J Berrocal, Adrian E Raftery, Tilmann Gneiting, and Richard C Steed. Probabilistic weather forecasting for winter road maintenance. *J. Amer. Statist. Assoc.*, 105(490):522–537, 2010.

[BTK19] 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 and Data Mining*, pages 1337–1347, 2019.

[BYL+20] Lei Bai, Lina Yao, Can Li, Xianzhi Wang, and Can Wang. Adaptive graph convolutional recurrent network for traffic forecasting. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 17804–17815. Curran Associates, Inc., 2020.

[Cut11] M. Cuturi. Fast global alignment kernels. In *ICML*, 2011.

[CWD+20] Defu Cao, Yujing Wang, Juanyong Duan, Ce Zhang, Xia Zhu, Congrui Huang, Yunhai Tong, Bixiong Xu, Jing Bai, Jie Tong, and Qi Zhang. Spectral temporal graph neural network for multivariate time-series forecasting. In *Advances in Neural Information Processing Systems*, 2020.

[DLT+17] Joshua V Dillon, Ian Langmore, Dustin Tran, Eugene Brevdo, Srinivas Vasudevan, Dave Moore, Brian Patton, Alex Alemi, Matt Hoffman, and Rif A Saurous. Tensorflow distributions. *arXiv preprint arXiv:1711.10604*, 2017.

[Fav17] Favorita. Favorita forecasting dataset. <https://www.kaggle.com/c/favorita-grocery-sales-forecast>, 2017.

[FMK19] Robert Fildes, Shaohui Ma, and Stephan Kolassa. Retail forecasting: Research and practice. *Int. J. Forecasting*, 2019.

[GK14] Tilmann Gneiting and Matthias Katzfuss. Probabilistic forecasting. *Annual Review of Statistics and Its Application*, 1:125–151, 2014.

[Gle20] Jeffrey L Gleason. Forecasting hierarchical time series with a regularized embedding space. *San Diego*, 7, 2020.

[GR07] Tilmann Gneiting and Adrian E Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American statistical Association*, 102(477):359–378, 2007.

[GS90] Charles W Gross and Jeffrey E Sohl. Disaggregation methods to expedite product line forecasting. *J Forecast*, 9(3):233–254, 1990.

[HA18] Rob J Hyndman and George Athanasopoulos. *Forecasting: principles and practice*. OTexts, 2018.

[HAAS11] Rob J Hyndman, Roman A Ahmed, George Athanasopoulos, and Han Lin Shang. Optimal combination forecasts for hierarchical time series. *Comput Stat Data Anal*, 55(9):2579–2589, 2011.

[HDG21] Xing Han, Sambarta Dasgupta, and Joydeep Ghosh. Simultaneously reconciled quantile forecasting of hierarchically related time series. In *International Conference on Artificial Intelligence and Statistics*, pages 190–198. PMLR, 2021.

[HHG21] Xing Han, Jing Hu, and Joydeep Ghosh. Mecats: Mixture-of-experts for quantile forecasts of aggregated time series. *arXiv preprint arXiv:2112.11669*, 2021.

[HLW16] Rob J Hyndman, Alan J Lee, and Earo Wang. Fast computation of reconciled forecasts for hierarchical and grouped time series. *Comput Stat Data Anal*, 97:16–32, 2016.

[KKR+22] 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.

[LYSL17] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. *arXiv preprint arXiv:1707.01926*, 2017.

[M520] M5. M5 forecasting dataset. <https://www.kaggle.com/c/m5-forecasting-accuracy/>, 2020.

[MMV19] Konstantin Mishchenko, Mallory Montgomery, and Federico Vaggi. A self-supervised approach to hierarchical forecasting with applications to groupwise synthetic controls. *arXiv preprint arXiv:1906.10586*, 2019.

[MPS21] Paolo Mancuso, Veronica Piccialli, and Antonio M Sudoso. A machine learning approach for forecasting hierarchical time series. *Expert Syst. Appl.*, 182:115102, 2021.

[OCCB19] Boris N Oreshkin, Dmitri Carpov, Nicolas Chapados, and Yoshua Bengio. N-beats: Neural basis expansion analysis for interpretable time series forecasting. *arXiv preprint arXiv:1905.10437*, 2019.

[OCM+22] Kin G. Olivares, Cristian Challu, Grzegorz Marcjasz, Rafał Weron, and Artur Dubrawski. Neural basis expansion analysis with exogenous variables: Forecasting electricity prices with nbeatsx. *Int. J. Forecasting*, 2022.

[OGL+22] 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. *arxiv*, 2022.

[OMM+21] Kin G Olivares, O Nganba Meetei, Ruijun Ma, Rohan Reddy, Mengfei Cao, and Lee Dicker. Probabilistic hierarchical forecasting

with deep poisson mixtures. *arXiv preprint arXiv:2110.13179*, 2021.

[OR64] Ingram Olkin and Herman Rubin. Multivariate beta distributions and independence properties of the wishart distribution. *The Annals of Mathematical Statistics*, pages 261–269, 1964.

[oS20] Australian Bureau of Statistics. Labour force, australia, dec 2020. <https://www.abs.gov.au/statistics/labour/employment-and-unemployment/labour-force-australia/latest-release.>, 2020.

[PGA+20] Anastasios Panagiotelis, Puwasala Gamakumara, George Athanasopoulos, Rob J Hyndman, et al. Probabilistic forecast reconciliation: Properties, evaluation and score optimisation. *Monash econometrics and business statistics working paper series*, 26:20, 2020.

[PGAH23] 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.

[PSAD21] Biswajit Paria, Rajat Sen, Amr Ahmed, and Abhimanyu Das. Hierarchically regularized deep forecasting. *arXiv preprint arXiv:2106.07630*, 2021.

[RSG+18] Syama Sundar Rangapuram, Matthias W Seeger, Jan Gasthaus, Lorenzo Stella, Yuyang Wang, and Tim Januschowski. Deep state space models for time series forecasting. *Adv Neural Inf Process Syst*, 31:7785–7794, 2018.

[RSS+21] Kashif Rasul, Abdul-Saboor Sheikh, Ingmar Schuster, Urs M Bergmann, and Roland Vollgraf. Multivariate probabilistic time series forecasting via conditioned normalizing flows. In *ICLR*, 2021.

[RWB+21] 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 *ICML*, pages 8832–8843. PMLR, 2021.

[SBSC+19] David Salinas, Michael Bohlke-Schneider, Laurent Callot, Roberto Medico, and Jan Gasthaus. High-dimensional multivariate forecasting with low-rank gaussian copula processes. *arXiv preprint arXiv:1910.03002*, 2019.

[SFGJ20] David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. Deepar: Probabilistic forecasting with autoregressive recurrent networks. *Int. J. Forecasting*, 36(3):1181–1191, 2020.

[SYD19] Rajat Sen, Hsiang-Fu Yu, and Inderjit S Dhillon. Think globally, act locally: A deep neural network approach to high-dimensional time series forecasting. *NeurIPS*, 32, 2019.

[Tou19] Tourism. Tourism forecasting dataset. <https://robjhyndman.com/publications/mint/>, 2019.

[TTH17] Souhaib Ben Taieb, James W Taylor, and Rob J Hyndman. Coherent probabilistic forecasts for hierarchical time series. In *ICML*, pages 3348–3357. PMLR, 2017.

[VEC15] Tim Van Erven and Jairo Cugliari. Game-theoretically optimal reconciliation of contemporaneous hierarchical time series forecasts. In *Modeling and stochastic learning for forecasting in high dimensions*, pages 297–317. Springer, 2015.

[VSP+17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, 2017.

[WAH+15] Shanika L Wickramasuriya, George Athanasopoulos, Rob J Hyndman, et al. Forecasting hierarchical and grouped time series through trace minimization. *Department of Econometrics and Business Statistics, Monash University*, 105, 2015.

[WAH19] Shanika L Wickramasuriya, George Athanasopoulos, and Rob J Hyndman. Optimal forecast reconciliation for hierarchical and grouped time series through trace minimization. *J. Amer. Statist. Assoc.*, 114(526):804–819, 2019.

[Wik17] Wiki. Web traffic time series forecasting dataset. <https://www.kaggle.com/c/web-traffic-time-series-forecasting/data>, 2017.

[WTH20] Shanika L Wickramasuriya, Berwin A Turlach, and Rob J Hyndman. Optimal non-negative forecast reconciliation. *Stat. Comput.*, 30(5):1167–1182, 2020.

[YYZ17] Bing Yu, Haoteng Yin, and Zhanxing Zhu. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. *arXiv preprint arXiv:1709.04875*, 2017.

[ZAC22] Lorenzo Zambon, Dario Azzimonti, and Giorgio Corani. Efficient probabilistic reconciliation of forecasts for real-valued and count time series. *stat*, 1050:8, 2022.

[ZMW+22] Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *ICML*, pages 27268–27286. PMLR, 2022.

A RELATED WORK ON HIERARCHICAL FORECASTING

Coherent Point Forecasting: As mentioned earlier, many existing coherent hierarchical forecasting methods rely on a two-stage reconciliation approach. More specifically, given non-coherent base forecasts $\hat{\boldsymbol{y}}_t \in \mathbb{R}^N$, reconciliation approaches aim to design a projection matrix $\boldsymbol{P} \in \mathbb{R}^{m \times N}$ that can project the base forecasts linearly into new leaf forecasts, which are then aggregated using $\boldsymbol{S}$ to obtain (coherent) revised forecasts $\tilde{\boldsymbol{y}}_t = \boldsymbol{S}\boldsymbol{P}\hat{\boldsymbol{y}}_t \in \mathbb{R}^N$. The post-processing is call reconciliation or on other words base forecasts are reconciled. Different hierarchical methods specify different ways to optimize for the $\boldsymbol{P}$ matrix. The naive Bottom-Up approach [HA18] simply aggregates up from the base leaf predictions to obtain revised coherent forecasts. The MinT method [WAH19] computes $\boldsymbol{P}$ that obtains the minimum variance unbiased revised forecasts, assuming unbiased base forecasts. The ERM method from [BTK19] optimizes $\boldsymbol{P}$ by directly performing empirical risk minimization over the mean squared forecasting errors. Several other criteria [HAAS11, VEC15, PGA+20] for optimizing for $\boldsymbol{P}$ have also been proposed. Note that some of these reconciliation approaches like MinT can be used to generate confidence intervals by estimating the empirical covariance matrix of the base forecasts [HA18].

Coherent Probabilistic Forecasting: The PERMBU method in [TTH17] is a reconciliation based hierarchical approach for probabilistic forecasts. It starts with independent marginal probabilistic forecasts for all nodes, then uses samples from marginals at the leaf nodes, applies an empirical copula, and performs a mean reconciliation step to obtain revised (coherent) samples for the higher level nodes. [AGP+20] also discuss two approaches for coherent probabilistic forecasting: (i) using the empirical covariance matrix under the Gaussian assumption and (ii) using a non-parametric bootstrap method. The recent work of [RWB+21] is a single-stage end-to-end method that uses deep neural networks to obtain coherent probabilistic hierarchical forecasts. Their approach is to use a neural-network based multivariate probabilistic forecasting model to jointly model all the time series and explicitly incorporate a differentiable reconciliation step as part of model training, by using sampling and projection operations. A recent approach of [OMM+21] uses a Deep Poisson Mixture Network to models the joint probability of the leaf time series as a finite mixture of Poisson distributions.

Approximately Coherent Methods: Several approximately-coherent hierarchical models have also been recently proposed, that mainly use the hierarchy information for improving prediction quality, but do not guarantee strict coherence, and often do not generate probabilistic predictions. Many of them [MMV19, Gle20, HDG21, HHG21, PSAD21] use regularization-based approaches to incorporate the hierarchy tree into the model via ℓ_2 regularization. [KKR+22] imposes approximate coherence on probabilistic forecasts via regularization of the output distribution.

B PROOFS

B.1 PROOF OF THEOREM 4.1

We prove the claims about the excess risk of top-down and bottom-up approaches in the following two sections. Recall that ordinary least square (OLS) estimator $\hat{\theta} = \left(\sum_{i=1}^n \mathbf{x}_i \mathbf{x}_i^\top\right)^{-1} \sum_{i=1}^n \mathbf{x}_i y_i$. The population squared error of a linear predictor is defined as $(\hat{\theta} - \theta)^\top \Sigma (\hat{\theta} - \theta)$, which is also known as excess risk.

B.1.1 Excess risk of the top down approach

For the root node, the OLS predictor is written as

$$\hat{\theta}_0 = \left(\sum_{t=1}^n \mathbf{x}_t \mathbf{x}_t^\top\right)^{-1} \sum_{t=1}^n \mathbf{x}_t y_{t,0},$$

and the expected excess risk is

$$\begin{aligned}
&\mathbb{E}[(\hat{\theta}_0 - \theta_0)^\top \Sigma (\hat{\theta}_0 - \theta_0)] \\
&\overset{\text{(a)}}{=} \mathbb{E}\left[\sigma^2 \sum_{t=1}^{n} \mathbf{x}_t^\top \left(\sum_{r=1}^{n} \mathbf{x}_t \mathbf{x}_t^\top\right)^{-1} \Sigma \left(\sum_{r=1}^{n} \mathbf{x}_t \mathbf{x}_t^\top\right)^{-1} \mathbf{x}_t\right] \\
&\overset{\text{(b)}}{=} \mathrm{Tr}\left[\mathbb{E}\left[\sigma^2 \left(\sum_{t=1}^{n} \mathbf{x}_t \mathbf{x}_t^\top\right)^{-1} \Sigma\right]\right] \\
&\overset{\text{(c)}}{=} \sigma^2 d/(n-d-1)
\end{aligned} \tag{7}$$

where equation (a) holds by expanding $y_{i,0} = \mathbf{x}_i^\top \theta_0 + \eta_i$ and the fact that η_i is independent of $\mathbf{x}_i$, equation (b) holds by the property of trace, and equation (c) follows from the mean of the inverse-Wishart distribution.

For each children node, we learn the proportion coefficient with

$$\hat{p}_i = \frac{1}{n} \sum_{t=1}^{n} \frac{y_{t,i}}{y_{t,0}}.$$

Notice that

$$\begin{aligned}
\mathrm{Var}[\hat{p}_i] &= \frac{1}{n} \mathrm{Var}\left[\frac{y_{1,i}}{y_{1,0}}\right] \\
&= \frac{1}{n} \mathrm{Var}[a_i] \\
&= s_i/n.
\end{aligned} \tag{8}$$

Recall that the optimal linear predictor of the i-th child node is $p_i \theta_0$. Therefore, the expected excess risk of the top down predictor is

$$\begin{aligned}
&\mathbb{E}\left[(\hat{p}_i \hat{\theta}_0 - p_i \theta_0)^\top \Sigma (\hat{p}_i \hat{\theta}_0 - p_i \theta_0)\right] \\
&= \mathbb{E}\left[(\hat{p}_i \hat{\theta}_0 - p_i \hat{\theta}_0 + p_i \hat{\theta}_0 - p_i \theta_0)^\top \Sigma (\hat{p}_i \hat{\theta}_0 - p_i \hat{\theta}_0 + p_i \hat{\theta}_0 - p_i \theta_0)\right] \\
&= \mathbb{E}\left[(\hat{p}_i - p_i)^2 \hat{\theta}_0^\top \Sigma \hat{\theta}_0 + p_i^2 (\hat{\theta}_0 - \theta_0)^\top \Sigma (\hat{\theta}_0 - \theta_0)\right] \\
&\overset{\text{(a)}}{=} \frac{1}{n} s_i \theta_0^\top \Sigma \theta_0 + \left(\frac{1}{n} s_i + p_i^2\right) \frac{d}{n-d-1} \sigma^2,
\end{aligned}$$

where we have applied Equation 8 and Equation 7 in equality (a). Taking summation over all the children, we get the total excess risk equals

$$\frac{\sum_{i=1}^{K} s_i}{n} \theta_0^\top \Sigma \theta_0 + \left(\frac{\sum_{i=1}^{K} s_i}{n} + \sum_{i=1}^{K} p_i^2\right) \frac{d}{n-d-1} \sigma^2$$

B.1.2 Excess risk of the bottom up approach

For the i-th child node, the OLS estimator is

$$\hat{\theta}_i = \left(\sum_{t=1}^{n} \mathbf{x}_t \mathbf{x}_t^\top\right)^{-1} \sum_{t=1}^{n} \mathbf{x}_t y_{t,i}.$$

Recall that the best linear predictor of the i-th child node is $p_i \theta_0$ The excess risk is

$$\begin{aligned}
&\mathbb{E}(\hat{\theta}_i - p_i \theta_0)^\top \Sigma (\hat{\theta}_i - p_i \theta_0) \\
&= \mathbb{E}\left[((\hat{\theta}_i - p_i \hat{\theta}_0) + (p_i \hat{\theta}_0 - p_i \theta_0))^\top \Sigma ((\hat{\theta}_i - p_i \hat{\theta}_0) + (p_i \hat{\theta}_0 - p_i \theta_0))\right]
\end{aligned}$$

Notice that the cross term has 0 expectation as

$$\begin{aligned}
&\mathbb{E}[(\hat{\theta}_i - p_i\hat{\theta}_0)^\top \Sigma(p_i\hat{\theta}_0 - p_i\theta_0)] \\
&\overset{(a)}{=} \mathbb{E}\left[\mathbb{E}_{\mathbf{a}}\left[\sum_{t=1}^{n}(a_{t,i}y_{t,0} - p_i y_{t,0})\mathbf{x}_t^\top \left(\sum_{t=1}^{n}\mathbf{x}_t\mathbf{x}_t^\top\right)^{-1}\Sigma(p_i\hat{\theta}_0 - p_i\theta_0)\right]\right] \\
&\overset{(b)}{=} 0,
\end{aligned}$$

where the first equality holds by the definition of node i-th value $y_{t,i}$. Therefore, it holds that

$$\begin{aligned}
&\mathbb{E}\left[((\hat{\theta}_i - p_i\hat{\theta}_0) + (p_i\hat{\theta}_0 - p_i\theta_0))^\top \Sigma((\hat{\theta}_i - p_i\hat{\theta}_0) + (p_i\hat{\theta}_0 - p_i\theta_0))\right] \\
&= \mathbb{E}\left[(\hat{\theta}_i - p_i\hat{\theta}_0)^\top \Sigma(\hat{\theta}_i - p_i\hat{\theta}_0)\right] + p_i^2\frac{d}{n-d-1}\sigma^2 \\
&= \mathbb{E}\,\mathrm{Tr}\left[\sum_{j=1}^{n}(a_{j,i} - p_i)^2 y_{t,0}^2 \mathbf{x}_j\mathbf{x}_j^\top \left(\sum_{t=1}^{n}\mathbf{x}_t\mathbf{x}_t^\top\right)^{-1}\Sigma\left(\sum_{t=1}^{n}\mathbf{x}_t\mathbf{x}_t^\top\right)^{-1}\right] + p_i^2\frac{d}{n-d-1}\sigma^2 \\
&= s_i\mathbb{E}\,\mathrm{Tr}\left[\sum_{j=1}^{n}\left((\theta_0^\top\mathbf{x}_j)^2 + \eta_j^2\right)\mathbf{x}_j\mathbf{x}_j^\top \left(\sum_{t=1}^{n}\mathbf{x}_t\mathbf{x}_t^\top\right)^{-1}\Sigma\left(\sum_{t=1}^{n}\mathbf{x}_t\mathbf{x}_t^\top\right)^{-1}\right] + p_i^2\frac{d}{n-d-1}\sigma^2 \\
&\overset{(a)}{\geq} s_i\sigma^2\mathbb{E}\,\mathrm{Tr}\left[\left(\sum_{j=1}^{n}\mathbf{x}_j\mathbf{x}_j^\top\right)\left(\sum\mathbf{x}_i\mathbf{x}_i^\top\right)^{-1}\Sigma\left(\sum\mathbf{x}_i\mathbf{x}_i^\top\right)^{-1}\right] + p_i^2\frac{d}{n-d-1}\sigma^2 \\
&\overset{(b)}{=} (s_i + p_i^2)\sigma^2\frac{d}{n-d-1},
\end{aligned}$$

where inequality (a) holds since $(\theta_0^\top\mathbf{x}_j)^2$ term is non-negative, equality (b) holds by the property of inverse-Wishart distribution. Taking summation over all the children, we get the total excess risk is lower bounded by

$$\sum_{i=1}^{K}(s_i + p_i^2)\frac{d}{n-d-1}\sigma^2$$

This concludes the proof.

B.2 PROOF OF COROLLARY 4.2

In this section, we apply Theorem 4.1 to Dirichlet distribution to show that the excess risk of bottom-up approach is $\min(d, K)$ times higher than top-down approach for a natural setting.

Recall that a random vector $\mathbf{a}$ drawn from a K-dimensional Dirichlet distribution $\mathrm{Dir}(\alpha)$ with parameters α has mean $\mathbb{E}[\mathbf{a}] = \frac{1}{\sum_{i=1}^{K}\alpha_i}\alpha$, and the variance $\mathrm{Var}[a_i] = \frac{\alpha_i(1-\alpha_i)}{\sum_{i=1}^{K}\alpha_i+1}$. Let $\alpha_i = \frac{1}{K}$ for all $i \in [K]$, $\theta_0^\top\Sigma\theta_0 = \sigma^2$. The total excess risk of the top-down approach is

$$\begin{aligned}
\mathbb{E}\left[\sum_{i=1}^{K} r(\hat{\theta}_i^{\mathbf{t}})\right] &= \frac{\sum_{i=1}^{K} s_i}{n}\theta_0^\top\Sigma\theta_0 + \left(\frac{\sum_{i=1}^{K} s_i}{n} + \sum_{i=1}^{K} p_i^2\right)\frac{d}{n-d-1}\sigma^2 \\
&= \left(\frac{1-1/K}{2n} + \left(\frac{1-1/K}{2n} + \frac{1}{K}\right)\frac{d}{n-d-1}\right)\sigma^2.
\end{aligned}$$

The total excess risk of the bottom-up approach is lower bounded by

$$\begin{aligned}
\mathbb{E}\left[\sum_{i=1}^{K} r(\hat{\theta}_i^{\mathbf{b}})\right] &= (s_i + p_i^2)\frac{d}{n-d-1}\sigma^2 \\
&= \left(\frac{1-1/K}{2} + \frac{1}{K}\right)\frac{d}{n-d-1}\sigma^2
\end{aligned}$$

Now assuming that $n \geq 2d$, the top-down approach has expected risk $\mathbb{E}[\sum_{i=1}^{K} r(\hat{\theta}_i^{\mathbf{t}})] = O(\frac{1}{n} + \frac{d}{nK})$, and the bottom-up approach has expected risk $\mathbb{E}[\sum_{i=1}^{K} r(\hat{\theta}_i^{\mathbf{t}})] = \Omega(\frac{d}{n})$. Therefore, it holds that

$$\frac{\mathbb{E}[\sum_{i=1}^{K} r(\hat{\theta}_i^{\mathbf{b}})]}{\mathbb{E}[\sum_{i=1}^{K} r(\hat{\theta}_i^{\mathbf{t}})]} = \Omega(\min(d, K))$$

C DATASETS

We use publicly available benchmark datasets for our experiments.

1. M5 [4]: It consists of time series data of product sales from 10 Walmart stores in three US states. The data consists of two different hierarchies: the product hierarchy and store location hierarchy. For simplicity, in our experiments we use only the product hierarchy consisting of 3k nodes and 1.8k time steps. Time steps 1907 to 1913 constitute a test window of length 7. Time steps 1 to 1906 are used for training and validation.
2. Favorita [5]: It is a similar dataset, consisting of time series data from Corporación Favorita, a South-American grocery store chain. As above, we use the product hierarchy, consisting of 4.5k nodes and 1.7k time steps. Time steps 1681 to 1687 constitute a test window of length 7. Time steps 1 to 1686 are used for training and validation.
3. Tourism-L[6]: consists of monthly domestic tourist count data in Australia across 7 states which are sub-divided into regions, sub-regions, and visit-type. The data consists of around 500 nodes and 228 time steps. This dataset consists of two hierarchies (Geo and Trav) as also followed in [RWB$^+$21]. Time steps 1 to 221 are used for training and validation. The test metrics are computed on steps 222 to 228.
4. Traffic [Cut11]: Consists of car occupancy data from freeways in the Bay Area, California, USA. The data is aggregated in the same way as [BTK19], to create a hierarchy consisting of 207 nodes spanning 366 days. Time steps 1 to 359 are used for training and validation. The remaining 7 time steps are used for testing.
5. Labour: Australian employement data consisting of 514 time steps sampled monthly, and 57 node hierarchy.
6. Wiki2: This dataset is derived from a larger dataset consisting of daily views of 145k Wikipedia articles. We use a smaller version of the dataset introduced by [BTK19] which consists of a subset of 150 bottom level time series, and 199 total time series.

For both M5 and Favorita we used time features corresponding to each day including day of the week and month of the year. We also used holiday features, in particular the distance to holidays passed through a squared exponential kernel. In addition, for M5 we used features related to SNAP discounts, and features related to oil prices for Favorita. For Tourism, Traffic, Labour, and Wiki2 we only used date features such as day of the week, month of the year, and holiday features from the GluonTS package [ABBS$^+$20]. All the input features were normalized to -0.5 to 0.5.

Metrics: For point forecasting we use the metrics WAPE (Weighted Average Percentage Error) and NRMSE (Normalized Root Mean Squared Error). The definitions are as follows:

$$\text{wape}\left(\boldsymbol{Y}, \hat{\boldsymbol{Y}}\right) = \frac{\sum_{t,i} |Y_{t,i} - \hat{Y}_{t,i}|}{\sum_{t,i} |Y_{t,i}|}$$

$$\text{nrmse}\left(\boldsymbol{Y}, \hat{\boldsymbol{Y}}\right) = \frac{\sqrt{1/(T * N) \sum_{t,i} (Y_{t,i} - \hat{Y}_{t,i})^2}}{1/(T * N) \sum_{t,i} |Y_{t,i}|}.$$

For probabilistic forecasting we use the CRPS metric. Denote the F step q-quantile prediction for time series i by $\hat{\boldsymbol{Q}}_{\mathcal{F}}^{(i)}(q) \in \mathbb{R}^F$. $\hat{\boldsymbol{Q}}_s^{(i)}(q)$ denotes the q-th quantile prediction for the s-th future time-step for time-series i, where $s \in [F]$. Then the CRPS loss is:

[4] https://www.kaggle.com/c/m5-forecasting-accuracy/

[5] https://www.kaggle.com/c/favorita-grocery-sales-forecasting/

[6] https://robjhyndman.com/publications/mint/

Table 5: Normalized CRPS scores on all datasets. We average the deep learning based methods over 5 independent runs. The rest of the methods did not show any variance. We report the corresponding standard error and only bold the numbers that are significantly better than the rest. We also report the mean performance across all levels in the corresponding column. For ease of comparison, we restate some of the PERMBU numbers by [RSG$^+$18].

M5	L0	L1	L2	L3	Mean
DirProp	**0.0379** $\pm$ 0.0014	**0.0422** $\pm$ 0.0004	**0.0536** $\pm$ 0.0023	**0.2543** $\pm$ 0.0067	**0.0970** $\pm$ 0.0013
Hier-E2E	0.1129 $\pm$ 0.0008	0.1106 $\pm$ 0.0008	0.1167 $\pm$ 0.0010	0.2940 $\pm$ 0.0012	0.1586 $\pm$ 0.0005
PERMBU	0.0639	0.0673	0.0737	0.2978	0.1257
AutoARIMA-BU	0.1188	0.1173	0.1202	0.2945	0.1627
AutoARIMA-ERM	2.9453	3.0110	3.0552	14.613	5.9062
AutoARIMA-TD	0.0599	0.0643	0.0713	0.2808	0.1191
AutoARIMA-MinT	0.0566	0.0725	0.0880	0.3074	0.1311

Favorita	L0	L1	L2	L3	Mean
DirProp	**0.0430** $\pm$ 0.0024	**0.0709** $\pm$ 0.0016	**0.1132** $\pm$ 0.0017	**0.2446** $\pm$ 0.0023	**0.1179** $\pm$ 0.0018
Hier-E2E	0.0955 $\pm$ 0.0009	0.1211 $\pm$ 0.0018	0.1648 $\pm$ 0.0039	0.3305 $\pm$ 0.0060	0.1780 $\pm$ 0.0028
PERMBU	0.0561	0.8279	0.6142	0.3184	0.4541
AutoARIMA-BU	0.0563	**0.0697**	**0.1119**	0.3190	0.1392
AutoARIMA-ERM	1.4857	1.7470	2.4220	4.8256	2.6201
AutoARIMA-TD	0.0802	0.2606	0.5253	1.1120	0.4945
AutoARIMA-MinT	0.0781	0.1539	0.2456	0.4448	0.2306

Tourism	L0	L1 (Geo)	L2 (Geo)	L3 (Geo)	L1 (Trav)	L2 (Trav)	L3 (Trav)	L4 (Trav)	Mean
DirProp	0.0457 $\pm$ 0.0033	0.0823 $\pm$ 0.0027	0.1341 $\pm$ 0.0028	**0.1769** $\pm$ 0.0027	0.0812 $\pm$ 0.0018	0.1358 $\pm$ 0.0008	**0.2015** $\pm$ 0.0009	**0.2684** $\pm$ 0.0008	**0.1407** $\pm$ 0.0022
Hier-E2E	0.0810 $\pm$ 0.0053	0.1030 $\pm$ 0.0030	0.1361 $\pm$ 0.0024	**0.1752** $\pm$ 0.0026	0.1027 $\pm$ 0.0062	0.1403 $\pm$ 0.0047	0.2050 $\pm$ 0.0028	0.2727 $\pm$ 0.0017	0.1520 $\pm$ 0.0033
PERMBU	0.131	0.129	0.1723	0.2189	0.1698	0.3063	0.5461	0.3415	0.2518
AutoARIMA-BU	0.1240	0.1166	0.1612	0.2134	0.1249	0.1554	0.2401	0.3428	0.1848
AutoARIMA-ERM	0.0465	0.1181	0.1970	0.2781	**0.0678**	0.1571	0.2952	0.4304	0.1987
AutoARIMA-TD	0.0332	0.0823	0.1547	0.2137	0.4237	0.7107	1.0285	1.2487	0.4869
AutoARIMA-MinT	**0.0322**	**0.0673**	**0.1270**	0.1999	0.0733	**0.1298**	0.2149	0.3352	0.1474

Labour	L0	L1	L2	L3	Mean
DirProp	**0.0172** $\pm$ 0.0019	**0.0242** $\pm$ 0.0018	**0.0243** $\pm$ 0.0016	**0.0345** $\pm$ 0.0007	**0.0250** $\pm$ 0.0015
Hier-E2E	0.0311 $\pm$ 0.0120	0.0336 $\pm$ 0.0089	0.0336 $\pm$ 0.0082	0.0378 $\pm$ 0.0060	0.0340 $\pm$ 0.0088
PERMBU	0.0406	0.0389	0.0382	0.0397	0.0393
AutoARIMA-BU	0.0314	0.0402	0.0393	0.0361	0.0368
AutoARIMA-ERM	0.0246	0.0306	0.0335	0.0495	0.0346
AutoARIMA-TD	0.0343	0.0458	0.0462	0.0462	0.0431
AutoARIMA-MinT	0.0396	0.0392	0.0410	0.0436	0.0409

Traffic ($F = 7$)	L0	L1	L2	L3	Mean
DirProp	**0.0213** $\pm$ 0.0041	**0.0247** $\pm$ 0.0039	**0.0296** $\pm$ 0.0032	0.1350 $\pm$ 0.0001	0.0527 $\pm$ 0.0028
Hier-E2E	0.0245 $\pm$ 0.0011	**0.0268** $\pm$ 0.001	**0.0307** $\pm$ 0.0011	**0.1206** $\pm$ 0.0019	**0.0506** $\pm$ 0.0011
PERMBU	0.0780	0.0744	0.0708	0.1844	0.1019
AutoARIMA-BU	0.0682	0.0648	0.0621	0.1832	0.0946
AutoARIMA-ERM	0.0997	0.1086	0.1117	0.3364	0.1641
AutoARIMA-TD	0.0486	0.0507	0.0549	0.1485	0.0757
AutoARIMA-MinT	0.0340	0.0429	0.0570	0.1859	0.0800

Wiki2 ($F = 7$)	L0	L1	L2	L3	L4	Mean
DirProp	**0.1483** $\pm$ 0.0116	**0.2096** $\pm$ 0.0056	**0.2817** $\pm$ 0.0042	**0.29** $\pm$ 0.0036	**0.4233** $\pm$ 0.005	**0.2706** $\pm$ 0.0048
Hier-E2E	**0.133** $\pm$ 0.0102	**0.2094** $\pm$ 0.0057	0.2942 $\pm$ 0.0032	0.3057 $\pm$ 0.0031	0.4421 $\pm$ 0.0016	0.2769 $\pm$ 0.004
PERMBU	0.1859	0.3437	0.5551	0.5635	0.8685	0.5033
AutoARIMA-BU	0.1954	0.3853	0.6083	0.6155	0.9732	0.5555
AutoARIMA-ERM	0.3238	0.4981	0.6285	0.6458	0.9778	0.6148
AutoARIMA-TD	0.2449	0.3398	0.3841	0.389	0.4577	0.3631
AutoARIMA-MinT	0.2171	0.3651	0.5525	0.6542	1.2531	0.6084

Table 6: Dataset characteristics. F denotes the horizon.

Dataset	Total time series	Leaf time series	Levels	Observations	F
M5	3060	3049	4	1913	35 days
Favorita	4471	4100	4	1687	35 days
Tourism-L (Geo)	111	76	4	228	12 months
Tourism-L (Trav)	445	304	5	228	12 months
Traffic	207	200	4	366	7 days
Labour	57	32	4	514	8 months
Wiki2	199	150	5	366	7 days

$$\mathrm{CRPS}(\hat{\boldsymbol{Q}}_{\mathcal{F}}^{(i)}(q), \boldsymbol{Y}_{\mathcal{F}}^{(i)}) = \frac{1}{F} \sum_{s \in [F]} \int_0^1 2(\mathbb{I}[\boldsymbol{Y}_s^{(i)} \leq \hat{\boldsymbol{Q}}_s^{(i)}(q)] - q)(\hat{\boldsymbol{Q}}_s^{(i)}(q) - \hat{\boldsymbol{Y}}_s^{(i)}) dq.$$

We normalize the score by the absoluted true values.

D FULL RESULTS ON ALL DATASETS

Tables 5 show the full set of results for all remaining datasets for all probabilistic baselines.

E ADDITIONAL EXPERIMENTAL DETAILS

Hyper-parameters and validation. As mentioned before, we use the last F time-points as the test set and the F time-points before the test window as the validation set. All hyper-parameters are tuned using the validation set. Then a model with the best hyperparameter (hparams) is trained on training + validation set. We report the metrics obtained by this model on the test set.

In order to reduce the total number of hparams we use a single hparam `hiddenSize` for all hidden state dimension parameters. This controls the size of hidden state in `enc`, `dec`,$\texttt{dec}_{\mathcal{F}}$, $\texttt{dec}^{(p)}$, $\texttt{dec}_{\mathcal{F}}^{(p)}$ and the fully connected layer after each attention layer in `MultiHeadAtt`. We tuned this between [128, 256, 512]. The number of hidden layers in `enc` is dubbed `numEncoderLayers` and the number of hidden layers in `dec` and $\texttt{dec}^{(p)}$ is controlled by `numDecoderLayers`. Both of them were tuned within [2, 3]. The other decoders have only one hidden layer. Learning rate is controlled by `learningRate`, which was tuned in log-scale from 1e-5 to 1e-2. The attention layer has parameters `numAttHeads` (tuned in [8, 16, 32]) and `numAttLayers` (tuned in [2, 3, 5]). We also tune the `batchSize` within [8, 16, 32].

Now we will specify the chosen hparams for all datasets. Note that we also tune the context length among a few values for each dataset similar to what was done in [RWB$^+$21].

Favorita. `learningRate`: 0.00085, `hiddenSize`: 128, `numAttLayers`: 3, `numAttHeads`: 8, `batchSize`: 32, `numEncoderLayers`: 3, `numDecoderLayers`: 3. The context length is tuned between [140, 70, 35, 28] and 70 was chosen.

M5. `learningRate`: 0.00034, `hiddenSize`: 128, `numAttLayers`: 3, `numAttHeads`: 16, `batchSize`: 32, `numEncoderLayers`: 2, `numDecoderLayers`: 2. The context length is tuned between [140, 70, 35, 28] and 28 was chosen.

Toursim-L. `learningRate`: 0.00007, `hiddenSize`: 512, `numAttLayers`: 5, `numAttHeads`: 16, `batchSize`: 16, `numEncoderLayers`: 3, `numDecoderLayers`: 2. The context length was fixed to 36.

Traffic. `learningRate`: 0.0001, `hiddenSize`: 512, `numAttLayers`: 2, `numAttHeads`: 8, `batchSize`: 16, `numEncoderLayers`: 3, `numDecoderLayers`: 3. The context length was fixed to 300.

Labour. `learningRate`: 0.00006, `hiddenSize`: 256, `numAttLayers`: 3, `numAttHeads`: 8, `batchSize`: 16, `numEncoderLayers`: 3, `numDecoderLayers`: 2. The context length was tuned in [8, 16, 32 64] and 32 was chosen.

Wiki2. `learningRate`: 0.00006, `hiddenSize`: 512, `numAttLayers`: 2, `numAttHeads`: 16, `batchSize`: 8, `numEncoderLayers`: 3, `numDecoderLayers`: 3. The context length was tuned in [140, 70, 35, 28] and 28 was chosen.

Training details. Our model is implemented in Tensorflow [ABC+16] and trained using the Adam optimizer with default parameters. We set a step-wise learning rate schedule that decays by a factor of 0.5 a total of 8 times over the schedule. The max. training epoch is set to be 50 while we early stop with a patience of 10. All our experiments were performed on a single server with a 32 core Intel Xeon CPU and an Tesla V100 GPU.

Baselines. We used the experimental framework released by [RWB+21] for running the baselines PERMBU, Hier-E2E. On the Favoroita dataset, the original R code of PERMBU does not work because of non positive definite covariance matrix. Therefore we use the implementation in [OGL+22] as that code is more modular and easy to debug.