Large-Scale Enterprise Revenue Forecasting in Action

Panpan Xu
xupanpan@amazon.com
Amazon Machine Learning Solutions Lab

Goktug T. Cinar
Goktug.Cinar@us.bosch.com
Bosch Center for Artificial Intelligence

Ryan Burt
Ryan.Burt@us.bosch.com
Bosch Center for Artificial Intelligence

Jasleen Grewal
gjaslee@amazon.com
Amazon Machine Learning Solutions Lab

Anton Iakovlev
anton.iakovlev@de.bosch.com
Bosch Center for Artificial Intelligence

Michael Binder
Michael.Binder3@de.bosch.com
Bosch

Selvan Senthivel
ssenthiv@amazon.com
Amazon Machine Learning Solutions Lab

Ruilin Zhang
ruilz@amazon.com
Amazon Machine Learning Solutions Lab

Adrian Horvath
AdrianLaszlo.Horvath@hu.bosch.com
Bosch Center for Artificial Intelligence

Miguel Romero Calvo
miguelrc@amazon.com
Amazon Machine Learning Solutions Lab

Lin Lee Cheong
lcheong@amazon.com
Amazon Machine Learning Solutions Lab

ABSTRACT

Revenue forecasting for large business organizations is a challenging but important problem. As a multinational business organization, Bosch has an estimated 2,000,000+ time series capturing monthly financial key figures at multiple organizational and product hierarchies, which are forecasted every month into the future 12 month horizon to inform financial and resource planning. To address this challenge, Bosch has developed an in-house forecasting solution serving 20+ forecasting models, while a meta-model is used for selecting a subset of these models best suitable for each individual time series. The framework is designed to be flexible and adaptive to support continuous introduction of new models, from both an algorithmic and architectural point of view.

We show how we can utilize the flexible development environment on AWS to explore a set of neural forecasting models and demonstrate a path for integration into the existing solution at Bosch via REST API. While doing so we place special attention on addressing the challenges brought forth by unexpected global events such as COVID-19. We investigate recent deep neural network (DNN)-based forecasters, which shows promising results for many forecasting problems. More specifically, we include the off-the-shelf Recurrent Neural Networks (RNN) and Convolutional Neural Networks (CNN) models (DeepAR+ and CNNQR) available from Amazon Forecast, as well as a custom-built Transformer model. The Transformer model incorporates a special module with attention adjustment to handle out-of-distribution COVID-19 period data. Backtest results from the Amazon Forecast models and the Transformer model are used to obtain ensemble forecasts, which lead to robust forecasts over time. The ensemble model serves the forecasting results to Bosch's in-house product from a forecasting pipeline implemented in the cloud in a modularized manner via REST API, which is deployed and currently in production.

ACM Reference Format:
Panpan Xu, Goktug T. Cinar, Ryan Burt, Jasleen Grewal, Anton Iakovlev, Michael Binder, Selvan Senthivel, Ruilin Zhang, Adrian Horvath, Miguel Romero Calvo, and Lin Lee Cheong. 2022. Large-Scale Enterprise Revenue Forecasting in Action. In *The 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '22), August 4–8, 2022, Washington, DC, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

KDD '22, August 14–18, 2022, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Revenue forecasting is a difficult but crucial task for strategic business decisions and fiscal planning in most business organizations [12]. Often, revenue forecasting is manually performed by financial analysts and is both time consuming and subjective. Such manual efforts are especially challenging for large-scale, multinational business organizations that require revenue forecasts across a wide range of product groups and geographical areas, as well as forecasts at multiple levels of granularity. This requires not only accuracy but also hierarchical coherence of the forecasts.

In this paper, first we give an overview a financial forecasting solution for large-scale hierarchical revenue data at Bosch, developed by Bosch Center for Artificial Intelligence (BCAI). We then introduce in detail how BCAI and Amazon Machine Learning Solutions Lab (MLSL) worked together to incorporate latest advances in DNN-based forecasting models. Bosch is a multinational corporation with more than 70B Euro in sales revenue in 2020[9] with

entities operating in multiple sectors including automotive, industrial solutions and consumer goods. Given the impact of accurate and coherent revenue forecasting on healthy business operations, Bosch Center for Artificial Intelligence has been heavily investing in the use of machine learning (ML) to improve the efficiency and accuracy of financial planning processes. The goal is to alleviate the manual processes by providing reasonable baseline revenue forecasts via ML, with only occasional adjustments needed by the financial analysts using their industry and domain knowledge. In an agile manner the product development started with models only leveraging historical information at a monthly granularity, with a forecasting horizon of 12-month into the future.[1]

To achieve this goal, Bosch has developed an internal forecasting framework capable of providing large-scale hierarchical forecasts via customized ensemble of a wide range of base models. A meta-learner selects the best performing models based on features extracted from each time series [7], following the practice described in [23]. The forecasts from the selected models are then averaged to obtain the aggregated forecast. The architectural design is modularized and extensible through the implementation of a REST-style interface protocol [22], which allows continuous performance improvement via the inclusion of additional models. The main goal of the collaboration between BCAI and Amazon ML Solutions Lab is to investigate a variety of neural forecasters and incorporate them as part of Bosch's model universe, which will be described in detail in the paper.

Recent advances in neural forecasting have demonstrated state-of-the-art performance for many practical forecasting tasks [6]. Compared to traditional forecasting models, many neural forecasters can incorporate additional covariates or meta-data about the time series. We evaluated CNNQR [8, 30] and DeepAR+ [27], two off-the-shelf Amazon Forecast models[2], as well as a custom Transformer model. The three models were chosen as they cover a representative set of the encoder backbones often used in neural forecasters: convolutional neural network (CNN), sequential recurrent neural network (RNN), and transformer-based encoders.

One of the key challenges faced by the BCAI-MLSL team was to provide robust and reasonable forecast under the impact of COVID-19, an unprecedented global event causing great volatility on financial figures for all corporations world-wide. As neural forecasters are trained on historical data, the forecasts generated based on out-of-distribution data from the more volatile periods could be significantly less accurate and reliable. We propose the addition of a masked attention mechanism in the transformer architecture to address this issue, as described in detail in Section. 3.2.

The neural forecasters we investigated can be bundled as a single ensemble model, or incorporated individually into Bosch's model universe, and accessed easily via REST API endpoints. We propose an approach to ensemble the neural forecasters through backtest results, which provides competitive and robust performance over time (described in Section. 3.1.3). Additionally, we investigated and evaluated a number of classical hierarchical reconciliation techniques to ensure that the forecasts aggregates coherently across product groups, geographies, and business organizations. There is no existing guidance on which techniques work better for neural forecasters, and the experimental results reported in Section. 3.3 show that either bottom-up or top-down approach with forecasting proportions provides the best performance with neural forecasters on this dataset based on the median-MAAPE (Mean Arctangent Absolute Percentage Error) and weighted-MAAPE metrics. All the experiments in this paper are reported on a synthetic dataset, described in detail in Section 4.1, provided by Bosch that statistically mimics the characteristics of the revenue figures from one business unit in the original dataset.

To summarize, the key contributions of this paper are:

- An adaptable system design with forecast model ensembles for large-scale automated financial forecasting that is highly parallelizable and reconfigurable.
- Evaluation of a representative set of neural forecasters for large-scale revenue forecasting.
- Evaluation of a set of hierarchical reconciliation techniques for neural forecasters.
- An ensemble strategy to combine off-the-shelf Amazon Forecast models with custom models to obtain better forecasting performance.
- Strategy to handle disruptive and non-recurring events such as COVID when using neural forecasters.

The rest of the paper is organized as follows: Section 2.1 gives an overview of the problem setting. Section. 2.2 gives an overview of Bosch's current forecasting framework. Section. 3 walks through each individual components in the neural forecasting pipelines for large-scale, hierarchical revenue data. Section. 4 describes the experiments covering the dataset, the evaluation metrics, the training setup and the model performance results. Section.5 introduce the architecture implemented as a cloud-based solution to enable highly parallelized and reconfigurable model training for large-scale ensemble model.

2 BACKGROUND

In this section we introduce the revenue forecasting problem setting at Bosch and give a brief overview of Bosch's current forecasting system architecture. In the next section we will cover the main body of the work.

2.1 Revenue Forecasting at Bosch

Financial analysts of every business entity are tasked with forecasting financial key figures, including revenue, operational costs, and R&D expenditure. These key figures provide insights at different levels of aggregation about the current business situation that enables data-driven decision making, and any automated forecasting solution needs to provide forecasts at any arbitrary level of the aggregation. At Bosch, the aggregations can be imagined as grouped time series as a more general form of hierarchical structure [14]. Figure. 1 shows a simplified example with a 2-level structure, which mimics the hierarchical revenue forecasting structure at Bosch. The total number of time series need to be forecasted at Bosch is at the scale of millions. Notice that the top-level time series can be split by either products or regions, creating multiple paths to the bottom level forecasts. The revenue needs to be forecasted at every node

[1] In future iterations the product incorporates external information such as customer orders, e.g. B2B(Business-to-Business) orders placed ahead of time. The analysis of the impact on accuracy due to use of external factors is not in the scope of the current paper.

in the hierarchy with a forecasting horizon of 12 months. Monthly historical data is available.

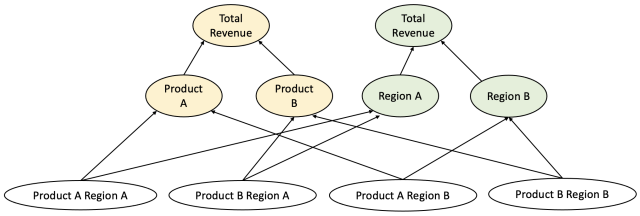

Figure 1: A simplified example of the hierarchical revenue forecasting structure used at Bosch, where the total revenue is split into multiple levels of aggregations based on *product* and *region*.

According to [14], the hierarchical structure in Figure. 1 can be represented using the following form using the notation of a summing matrix S.

$$Y = Sb \tag{1}$$

where:

$$Y = \begin{bmatrix} y_{Total}(t) \\ y_{ProdA}(t) \\ y_{ProdB}(t) \\ y_{RegionA}(t) \\ y_{RegionB}(t) \\ y_{ProdA,RegionA}(t) \\ y_{ProdB,RegionA}(t) \\ y_{ProdA,RegionB}(t) \\ y_{ProdB,RegionA}(t) \end{bmatrix}, S = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, b = \begin{bmatrix} y_{ProdA,RegionA}(t) \\ y_{ProdB,RegionA}(t) \\ y_{ProdA,RegionB}(t) \\ y_{ProdB,RegionB}(t) \end{bmatrix} \tag{2}$$

b represents the bottom-level time series at time t.

2.2 Bosch's In-House Forecasting Solution

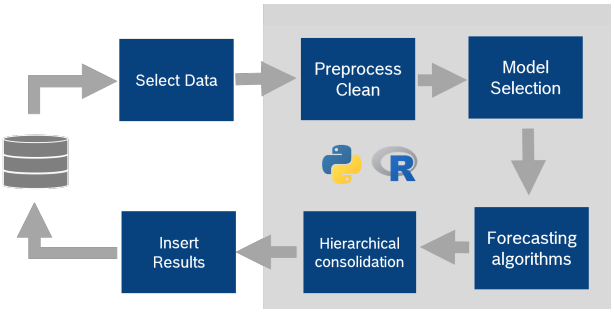

Figure 2: System architecture of Bosch's forecasting solution.

Figure. 2 illustrates the current architecture of the large scale automated forecasting solution implemented at Bosch. The system architecture developed by BCAI is deliberately designed to be modular and easily scalable. It involves a few critical building blocks as described below.

2.2.1 Model Universe. Every model is a simplification of reality, with simplifications achieved by making certain assumptions regarding the underlying behavior of the data. When these assumptions are accurate, the models generalize well and will provide high-quality forecasts. For an automated large-scale forecasting solution that serves many different financial key figures and business entities, it is not possible to assume a consistent statistical behavior among all time series (i.e. there are no universally true assumptions) and a model ensembling approach is needed - hence the term model universe. The model incorporates time-series models such as ARIMA, ETS [13, 16] that take in historical monthly revenue data for forecasting. The motivation is to use a large set of models (i.e. a large variety of assumptions) to create a robust forecasting solution.

2.2.2 Meta-learner. The most intuitive approach to combine multiple base models is the mean/median approach, where all models in the model universe are applied to a given time series. In [23], the authors describe Feature-based Forecasting Model Averaging (FFORMA) where it is possible to utilize a classification model to select the most suitable models for a given time series, which performs better than simple averaging of all models. This paradigm is called a meta-learner as it involves a process of learning to learn. The meta-learner takes in a variety of time series features such as length of timeseries and strength of peak for model selection. Bosch's current architecture utilize FFORMA to incorporate different forecasters in the model universe. FFORMA is a flexible framework where the model universe can be continuously expanded to improve the performance. The main goal of the BCAI-MLSL collaboration is to evaluate comprehensively a set of neural forecasters that can be taken in as a part of the model universe.

In Section. 3.1.3 we further introduce an alternative end-to-end model ensemble approach that does not explicitly depend on time series feature extraction and meta-learner for ensembling a set of (neural) forecasters. The model ensembling results can be fed into the model universe, or used as a standalone solution with the potential of incorporating more models in the future.

2.2.3 Hierarchical Reconciliation. For hierarchical time series, typically the base forecasts for each individual time-series are first generated and post-processing steps are used to consolidate the base forecasts and achieve hierarchical coherence [15]. Bottom-up (BU) and top-down (TD) approaches are two typical post-processing techniques. Recently, global methods such as Ordinary Least Squares (OLS) and Weighted Least Squares (WLS) [14] also gained popularity. Bosch's current framework uses WLS. Since no comprehensive evaluation of hierarchical reconciliation technique for neural forecasters exist, we also evaluated and compared the techniques mentioned above in this work, for the specific revenue forecasting scenario.

2.2.4 System Architecture and Adaptiveness of the Model Universe. Currently at Bosch, the forecasting solution need to be able to process >2 million time series within 3 hours. Therefore, distributed programming [10] is utilized as all nodes in the hierarchy can be forecasted independently prior to hierarchical reconciliation at individual business unit level.

There is additional built-in motivation to continuously add new models to the model universe to gain further robustness and coverage, with meta-learner eventually optimizing forecasting performance. Thus, Bosch strives to add performant models that might perform significantly better than existing models on a subset of data. These models can be called directly as software packages or via REST APIs. A predefined json format is used to query and retrieve forecasts from different models. In the next section we will be introducing how we jointly introduced new models to the Bosch model universe using Amazon Forecast and custom-built models using SageMaker.

3 METHODOLOGY

This section details the main body of the work for this paper. We introduce the neural forecasters and an end-to-end forecasting solution collaboratively investigated by BCAI and Amazon MLSL. We first give a brief introduction to the two off-the-shelf neural forecasters, CNNQR and DeepAR+, from Amazon Forecast, as well as a custom Transformer model and an end-to-end model ensembling strategy to combine the neural forecasters. Then we give an introduction to the COVID disruption problem and describe the method for handling it. In the end we give an overview of the post-processing steps and the hierarchical reconciliation strategy. Section. 5 will give a detailed description about how the forecasting workflow is implemented as an automatic, highly parallelizable and extensible architecture deployed on the cloud.

3.1 Model Architecture

3.1.1 Amazon Forecast models. Amazon Forecast is a fully-managed AI/ML service from Amazon Web Services (AWS) that provides preconfigured, state-of-the-art time series forecasting models [2]. It combines these offerings with its internal capabilities for automated hyper-parameter optimization, model ensembling (for the models provided by Amazon Forecast), and probabilistic forecast generation. This allows users to easily ingest custom datasets, preprocess data, train forecasting models, and generate robust forecasts. The service's modular design further enables us to easily query and absorb predictions from additional custom models developed in parallel.

We incorporate two neural forecasters from Amazon Forecast in this investigation - CNN-QR [30], DeepAR+ [27]. Both are supervised deep learning methods that train a global model for the entire set of time series. Both CNNQR and DeepAR+ models can take in static meta data information about each time series, which are the corresponding product, region and business organization in our case. Both CNNQR and DeepAR+ automatically add temporal features such as month of the year as part of the input to the model.

3.1.2 Transformer Architecture. The Transformer architecture, originally designed for natural language processing (NLP), recently emerged as a popular architectural choice for time series forecasting and has achieved state-of-the-art performance on benchmark datasets [19, 20, 33]. Here, we used the transformer architecture described in [33] without probabilistic log sparse attention [2]. The model uses a typical architecture design by combining an encoder and a decoder, as illustrated in Figure. 4. For revenue forecasting, we configure the decoder to directly output the forecast in the 12-month horizon instead of generating the forecast month-by-month in an autoregressive manner [21]. Based on the frequency of the time series, additional time related features such as month of the year is added as the input variable. Additional categorical variables describing the meta information (e.g. product, region, business organization) are fed into the network via a trainable embedding layer. Further modifications to specifically address the volatile COVID period are described in Section 3.2, which can be extended to address exception situations beyond COVID [3].

3.1.3 Model Ensemble. Model ensemble often outperforms single models for forecasting - it improves model generalizability and is better at handling time series data with varying characteristics in periodicity and intermittency. We incorporate a series of model ensembling strategies to improve model performance and robustness of forecasts. One common form of deep learning model ensemble is to aggregate results from model runs with different random weight initializations, or from different training epochs [11]. We utilize this strategy to obtain forecasts for the Transformer model. To further build an ensemble on top of different model architectures including Transformer, CNNQR, DeepAR+ and etc., we use a pan-model ensembling strategy that selects the top-k best performing models for each time series based on the backtest results and obtain their averages [4]. As backtest results can be exported directly from trained Amazon Forecast [2] models, this strategy enables us to easily combine the benefits of turn-key services like Amazon Forecast with improvements gained from custom models such as Transformer. Such an end-to-end model ensembling approach does not require training a meta-learner or calculating time series features for model selection.

3.2 COVID Anomaly Handling with Masked Transformer

The COVID-19 pandemic brought significant challenges for forecasting due to its disruptive and unprecedented effects on almost all aspects of work and social life. Besides short-term forecasting problems (e.g. demand forecasting) being greatly affected [24], we observed that in our case of long-term revenue forecasting, the COVID disruption also brought unexpected downstream impact. To illustrate this problem, Figure 3 shows a sample time series where the product revenue experienced a significant drop at the start of the pandemic and gradually recovered afterwards. A typical neural forecasting model will take revenue data including the out-of-distribution (OOD) COVID period as the historical context input, as well as the ground truth for model training, and as a result the forecasts are no longer reliable. To address this problem, we have considered alternative strategies such as utilizing auxiliary data (e.g. Google keyword search trends, stock market data and micro-economics data) to capture COVID impact on the revenue [24]. However, since the revenue forecasts need to be generated 12 months in advance, such data are insufficient in serving as leading

[2]We used a context window with <36 month data as more historical revenue data are not available. The computational cost is sufficiently low to allow for the use of full-attention.

[3]For example, recently chip shortage has greatly impacted several industries and there is no historical data reflecting this situation.

[4]The hyper-parameter k can be selected for each dataset using backtest.

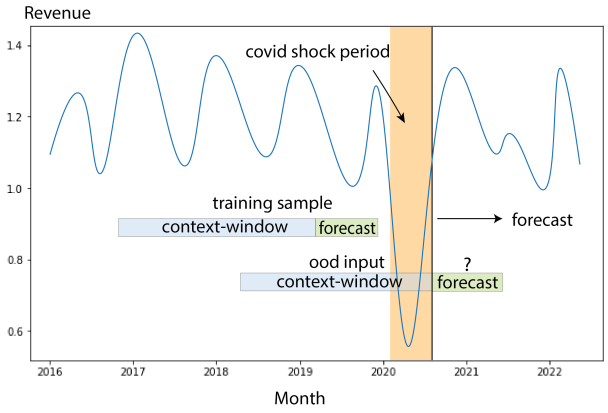

Figure 3: Illustrative sketch of a time series containing COVID shock period, mimicking the trend we observed in real dataset. At the start of COVID, significant and one-off disruptions are recorded and are not representative of longer trends. Training or back-testing neural models directly with time series containing such out-of-distribution (OOD) data results in degradation of performance and unreliable forecasts.

signals. We propose an alternative strategy to mitigate the impact of OOD context windows, by using Transformer with attention masks as illustrated in Figure 4. The model is trained to apply very little attention (set as zero in the experiments) on the COVID period that contains outliers via masking, and perform forecast with masked information. The attention mask is applied throughout every layer of the decoder and encoder architecture. The masked window can be either specified manually or through an outlier detection algorithm (e.g. seasonality decomposition based method such as [1]). Additionally, when using time window containing outliers as the training labels, the loss are not back-propagated. In Section 4.2, we compare the performance of transformer model before and after attention masking to demonstrate the effectiveness of the proposed approach. We anticipate that such attention masking based method can be applied to treat disruptions and OOD cases brought by other rare events as well and improve the robustness of the forecasts.

3.3 Post-Processing

3.3.1 Hierarchical reconciliation. While there are plenty of hierarchical reconciliation methods proposed in the literature [15, 32], there is a lack of guidance on what methods are more suitable for neural forecasting models. We investigated a wide range of techniques including bottom-up (BU), top-down (TD), ordinary least square (OLS) and weighted least square (WLS). We find that bottom-up (BU) or top-down reconciliation with forecasting proportions (TDFP) [5, 15] returned the best results in most cases, for the specific dataset and evaluation metrics under investigation. In this paper, all the experimental results are reported using top-down reconciliation with forecasting proportions. For experimental results comparing different hierarchical reconciliation techniques, please refer to Section 4.

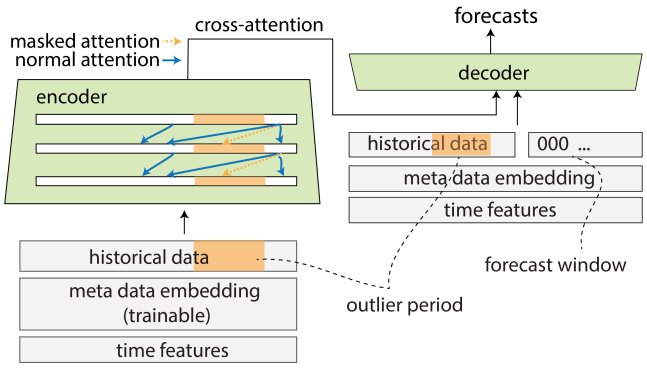

Figure 4: Illustrative sketch of the Transformer architecture and the attention masking mechanism. Attention masking is applied throughout all the encoder and decoder layers, as highlighted in orange in the figure, to prevent OOD data from affecting the forecasts.

4 EXPERIMENTS

4.1 Setup

Dataset. In this paper we focus on a revenue forecasting use case. To protect financial privacy of Bosch while using a realistic dataset, we used a synthetic dataset that has similar statistical characteristics to a real world revenue dataset from one business unit at Bosch . The dataset contains 1216 time series in total with revenue recorded in a monthly frequency, covering the time range from 2016-01 to 2022-04. The dataset is delivered with 877 time series at the most granular level (a.k.a bottom time series), with a corresponding grouped time series structure represented as a summing matrix S. The summing matrix size is 1216×877 (as described in Section. 2.1). Each time series is associated with three static categorical attributes, which corresponds to product category, region and organizational unit in the real dataset [5].

Backtest windows. We use rolling 12 months backtest windows to compare model performance. Figure. 5 illustrates the backtest windows used in the experiments and highlights the corresponding data used for training and hyperparameter optimization (HPO). The rolling backtest windows start from July 2019, when the 12 months forward looking forecasting accuracy is already greatly impacted by COVID – it is not quite possible to foresee COVID happening. For backtest windows after COVID starts the result will be affected by OOD inputs from April to May 2020, based on what we observed from the revenue time series. After May the financial analysts can already see revenue recovering, however, the neural forecasters take in the two months data for forecasting which can lead to unreliable results.

Model setup and training. For Transformer training we used *Quantile loss* and scaled each time series using its historical mean value before feeding it into Transformer and computing the training loss. The final forecasts are rescaled back to calculate the accuracy metrics, using the *MeanScaler* implemented in GluonTS[3]. We use a context window with monthly revenue data from the past 18

[5] Anonymized in the synthetic data.

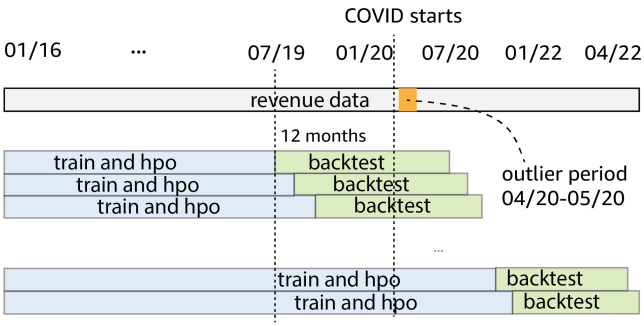

Figure 5: Illustrative sketch of the rolling backtest windows used in experimental result reporting. We use a rolling 12-month forecasting window to report model performance, starting from July 2019. Most of the forecasting windows are affected by COVID.

months, selected via hyperparameter optimization in the backtest window from July 2018 to June 2019. The model is trained with batch size 32 for 10 epochs with a learning rate of 1.0e-4, using Adam optimizer. Each batch consists of a set of uniformly randomly sampled time series segments from the training data. For each epoch we use 2048 random samples. The Transformer architecture used in the experiment contains 3 encoder layers and 2 decoder layers, 8 attention heads and the dimension of the internal representation is 512. Additional metadata about each time series in the form of static categorical variables are fed into the model, via an embedding layer before feeding it to the transformer layers. The dimension of the embedding is computed automatically based on the cardinality of the categorical variable, same as what is implemented in GluonTS[3], and linearly projected to the model dimension (512). We train the Transformer with five different random weight initialization and average the forecast results from the last three epochs for each run, in total averaging 15 models. The hyperparameters mentioned above are kept the same throughout the experiments described in the paper to obtain comparable results. For masked Transformer, we indicate the months from April to May 2020 as outliers.

For all Amazon Forecast model training, we enabled automatic HPO which can select the model and training parameters based on a user-specified back-test period, which is set to the last 12 months in the data window used for training and HPO.

Evaluation metrics. We use *median*-Mean Arctangent Absolute Percentage Error(median-MAAPE) and *weighted*-MAAPE to evaluate the model performance and perform comparative analysis, which are the standard metrics used at Bosch. MAAPE was introduced in [18] to address the shortcomings of Mean Absolute Percentage Error(MAPE) metric commonly used in business context. *median*-MAAPE gives an overview of the model performance by computing the median of the MAAPEs calculated individually on each time series. *weighted*-MAAPE reports a weighted combination of the individual MAAPEs. The weights are the proportion of the revenue for each time series compared to the aggregated revenue of the entire dataset, *weighted*-MAAPE better reflects downstream business impacts of the forecasting accuracy. Both metrics are reported on the entire dataset of 1216 time series.

Table 1: Comparison of different hierarchical reconciliation techniques, using the *median*-MAAPE metric. The detailed description of each method can be found in [15].

Backtest period	Algo.	BU	TDFP	OLS	WLS
07/19-06/20	CNNQR	0.**4466**	0.4482	0.4670	0.4581
	DeepAR+	0.4515	**0.4488**	0.4785	0.4613
	Trans.	**0.4966**	0.4987	0.5236	0.5141
11/19-10/20	CNNQR	0.4969	**0.4935**	0.5129	0.5024
	DeepAR+	**0.4756**	0.4760	0.5075	0.4901
	Trans.	**0.5378**	0.5509	0.5749	0.5561
03/20-02/21	CNNQR	0.4624	**0.4584**	0.4796	0.4606
	DeepAR+	**0.4776**	0.4805	0.5196	0.4905
	Trans.	**0.5217**	0.5236	0.5333	0.5236
07/20-06/21	CNNQR	**0.3810**	0.3823	0.3922	0.3875
	DeepAR+	0.4581	**0.4163**	0.4391	0.4309
	Trans.	0.3618	0.3614	0.3796	**0.3601**
11/20-10/21	CNNQR	0.3033	**0.3019**	0.3260	0.3134
	DeepAR+	0.3927	0.3855	**0.3837**	0.3881
	Trans.	**0.3289**	0.3441	0.3579	0.3447
03/21-02/22	CNNQR	**0.3272**	0.3277	0.3562	0.3359
	DeepAR+	**0.3169**	0.3179	0.3585	0.3295
	Trans.	**0.3487**	0.3604	0.3931	0.3768
	Average	0.4214	**0.4208**	0.4435	0.4291

4.2 Results and Discussion

4.2.1 Comparison of Hierarchical Reconciliation Techniques. We first compare several hierarchical reconciliation techniques to select an approach for the subsequent experiments. The methods we included in the comparison are bottom-up (BU), top-down with forecasting proportions (TDFP), ordinary least squares (OLS) and weighted least squares (WLS). For WLS we evaluated structural scaling based method. The detailed description of all the methods can be found in [15]. We compared the hierarchical reconciliation techniques across the three different neural forecasters and over a set of rolling backtest windows. The results are displayed in Table. 1. From the results it can be observed that for the revenue data using *median*-MAAPE as the evaluation metric, in most cases BU and TDFP has the best performance among the four methods. We use top-down with forecasting proportions for all the following experiments.

4.2.2 Comparison of Masked/Unmasked Transformer. We train masked (Trans.+M) and unmasked (Trans.) transformer, using the same set of hyperparameters described in Section. 4.1 and compared their performance for backtest windows immediately after COVID shock. In masked Transformer, the masked two months are April and May, 2020. Table.2 shows the results from a series of backtest periods with 12 months forecasting window starting from June 2020. It can be observed that masked Transformer consistently outperforms unmasked version.

4.2.3 Performance Evaluation of Back-test Based Ensemble Strategy. We further performed evaluation on the model ensemble strategy

Table 2: Compare before and after adding attention masks in Transformer to handle COVID shock. For both *median*-MAAPE and *weighted*-MAAPE the performance is better for masked Transformer.

	median-MAAPE		*weighted*-MAAPE	
Backtest period	Trans	Trans+M	Trans	Trans+M
06/20-05/21	**0.4015**	0.4045	0.2393	**0.2106**
07/20-06/21	0.3928	**0.3614**	0.2088	**0.1756**
08/20-07/21	0.3781	**0.3549**	0.1734	**0.1708**
09/20-08/21	0.3489	**0.3435**	**0.1611**	0.1761
10/20-09/21	0.3520	**0.3472**	0.1805	**0.1801**
11/20-10/21	**0.3377**	0.3441	0.1988	**0.1954**
Average	0.3738	**0.3696**	0.1978	**0.1922**

based on backtest results. In particular, we compare the two cases when only the top performing model is selected vs. when top two performing model are selected and model averaging is performed by computing the mean value of the forecasts. We compare the performance of the base models and the ensemble models in Figure. 6, notice that no neural forecasters consistently out-perform others for the rolling backtest windows. On average, Table. 4 shows that ensemble the top two models gives the best performance. CNNQR on the other hand provides the second best results. Table. 3 compare the top two ensemble and top one ensemble in more detail.

Table 3: Model Ensemble. We compared two strategies to perform heterogeneous model ensemble based on backtest results. One selects forecast from the top performing model, another one selects the top-2 model and average the forecasts.

	median-MAAPE		*weighted*-MAAPE	
Backtest period	Top-2	Top-1	Top-2	Top-1
07/19 - 06/20	**0.4683**	0.4937	**0.2963**	0.3008
09/19 - 08/20	**0.4750**	0.5236	**0.2757**	0.3118
11/19 - 10/20	**0.5102**	0.5315	**0.3096**	0.3274
01/20 - 12/20	**0.4725**	0.5251	**0.2892**	0.3148
03/20 - 02/21	**0.4785**	0.5193	**0.2951**	0.3176
05/20 - 04/21	**0.3927**	0.4307	0.2405	**0.2281**
07/20 - 06/21	**0.3615**	0.3986	**0.2346**	0.3136
09/20 - 08/21	**0.3497**	0.3927	**0.2140**	0.3021
11/20 - 10/21	**0.3215**	0.3273	**0.1844**	0.1903
01/21 - 12/21	**0.3158**	0.3628	**0.1771**	0.2266
03/21 - 02/22	**0.3018**	0.3371	**0.1954**	0.2483
average	**0.4043**	0.4402	**0.2465**	0.2801

5 CLOUD-BASED ARCHITECTURE DESIGN

We developed an automated end-to-end workflow on AWS to generate revenue forecasts utilizing services including Amazon Forecast, Amazon SageMaker Training, Amazon S3 Cloud Storage, AWS Lambda (an event-driven compute service), AWS Step Functions (distributed compute), and AWS Cloud Development Kit (AWS

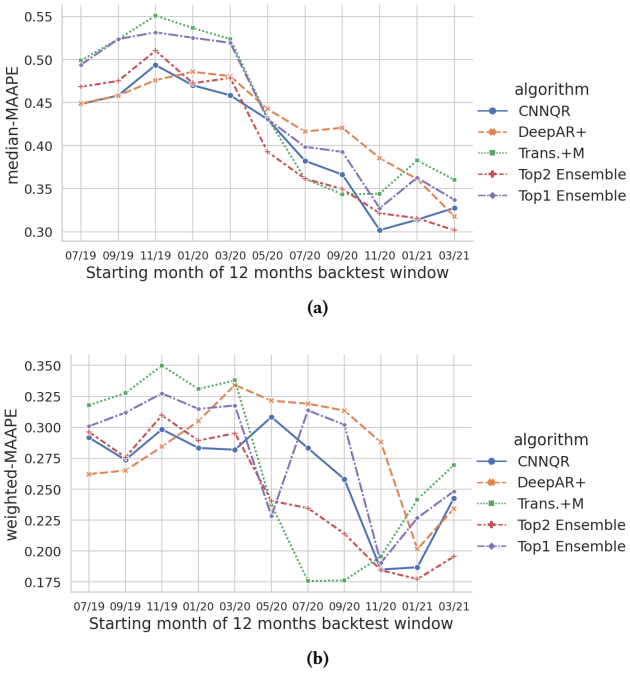

Figure 6: Plotting performance of ensemble and individual models over the rolling backtest windows in Table. 3. Overall top-2 ensemble provides more stable forecasting accuracy over time.

Table 4: Compare performance of ensemble and individual models over the rolling backtest windows in Table. 3. Result shows that top-2 ensemble provides the best performance overall.

metric	Top-2	Top-1	Trans.+M	CNNQR	DeepAR+
mMAAPE	**0.4043**	0.4402	0.4416	0.4046	0.4266
wMAAPE	**0.2465**	0.2801	0.2691	0.2629	0.2844

CDK). The solution architecture is illustrated in Figure. 7. The deployed solution provides individual time series forecasts through REST API by returning it in predefined json format.

Key design considerations for the architecture are *versatility (extensibility)*, *performance* and *user-friendliness*. The system should be sufficiently versatile to support incorporating a diverse set of algorithms during development and deployment, with minimal required changes, and can be extended to easily incorporate new algorithms in the future. The system should also add minimum overhead and support parallelized training for both Amazon Forecast and custom models to obtain latest forecasts in a short period of time. Finally, the system should be simple to use for experimentation purposes.

The end-to-end workflow in Figure 7 shows that the system sequentially runs through following modules: 1) a pre-processing module for data reformatting and transformation; 2) a model training module incorporating both Amazon Forecast and custom model

training, running both in parallel, and 3) a post-processing module supporting model ensembling, hierarchical reconciliation, metrics and report generation. Amazon Step Function is used to organize and orchestrate the workflow from end-to-end as a state machine. The state machine execution is configured with a json file containing all the necessary information including location of the historical revenue .csv files in S3 storage, forecast start time, and model hyperparameter settings to run the forecast workflow end-to-end. Asynchronous calls to Amazon Forecast and Amazon Sagemaker Training is enabled for parallelized model training in the state machine, using Lambda functions. All the historical data, config files, forecast results, as well as intermediate results such as backtesting results are stored in S3. The REST API is built on top of the S3 storage to provide queryable interface for forecasting results. The system can be extended to incorporate new forecast models and supporting functions such as generate visualization reports of the forecasts.

6 RELATED WORK

6.1 Revenue Forecasting

Revenue forecasting is a key component of business administration [17] and government planning [26]. In recent years the application of ML to this use case has been widely studied in a variety of scenarios. [31] describes the development of a revenue forecasting tool to extend the forecasting horizon from one to three months. [4] explores a more granular scenario of hotel demand forecasting and compares traditional time series and machine learning approaches.

6.2 Hierarchical Neural Forecasting Methods

Hierarchical Neural Forecasting Methods fall under two categories: first, point forecast approaches output a single point-estimate for each time series in the hierarchy. [26] introduced a multivariate two-step approach by differentiating between bottom level time series and aggregated time series, their implementation is based on a Deep Long Short-Term Memory Auto-Encoder (DLSTM-AE) and leveraged transfer learning to produce coherent aggregations (with small error). Similarly, [29] restricted the learning of the forecasting map to bottom time series while introducing a regularization parameter penalizing incoherent aggregates and leveraging a bottom-up reconciliation approach to compute the forecast of the aggregated time series. Second, probabilistic forecasting approaches output a probability distribution for each time series in the hierarchy. [28] models the conditional distributions for each time series in the hierarchy and computes their individual CDF for drawing samples from the bottom time series. Those samples are then re-ordered to match the error copula of the aggregated time series and are then used to compute its probabilistic forecast. [25] proposed an end- to-end multivariate probabilistic approach to forecast coherent hierarchical time series using DeepVAR that leverages the reparametrization trick and a projection computed to find a solution to the optimization problem inferred from the hierarchical matrix.

However, large corporations often have several hundreds or hundreds of thousands of time series grouped by product lines, geographies, organizational structure, etc. This paper builds upon previous revenue forecasting by providing a method to efficiently forecast large scale time series data in a hierarchically coherent manner. To the best of our knowledge, prior point and probabilistic forecast approaches applied to revenue forecasting restricts the learned function to be a single global model for all time series in the hierarchy while exploiting their hierarchical structure to output coherent results. However, as noted in [25], in real-life base time series are often sparse and their behaviour might be better modeled using different algorithms. In this paper we build upon existing work by creating a reliable ensemble mechanism that leverages a set of K models and selectively fuses information for each time series in the hierarchy followed by a post-hoc reconciliation step.

7 CONCLUSION AND FUTURE WORK

In this paper, we present a system for large scale, hierarchically coherent revenue forecasting that is highly parallelizable and reconfigurable. We also propose a new method applying attention masks on Transformer model to handle unexpected events such as COVID for forecasting and proposed a simple model ensembling approach based on backtest results. Experimental result shows that it provides robust and stable forecast accuracy over time compared to individual models. We expect that the architecture described in the paper can be used for not only revenue forecast, but also other forecasting applications as well.

Future work includes 1) investigate incorporating additional forecasting models in the current framework besides the current three models 2) perform comparative analysis of masked Transformer with other outlier handling techniques such as outlier detection and data imputation technique 3) evaluate the models on a broader set of metrics such as Mean Squared Error (MSE) or Mean Absolute Percentage Error (MAPE). 4) comparative analysis of the meta-learner based ensemble strategy with backtest based ensemble strategy.

8 ACKNOWLEDGMENTS

Authors would like to acknowledge the work of many Bosch and Amazon AWS colleagues that contributed to the product, Phil Gaudreau, Jeff Thompson, Patrick Emmerich, Alexey Deynega, Lisa Marion Garcia, Matthew Jones, Ilkay Nazli Celik Gulsoy, Jingchen Zeng, Laszlo Scherman, Shane Rai, Manuel Culebras, Shivani Mehendarge, Brian Reed and Meg Tennant.

REFERENCES

[1] 2021. Detecting time series outliers. https://robjhyndman.com/hyndsight/tsoutliers/
[2] 2022. Amazon Forecast. https://aws.amazon.com/forecast/
[3] Alexander Alexandrov, Konstantinos Benidis, Michael Bohlke-Schneider, Valentin Flunkert, Jan Gasthaus, Tim Januschowski, Danielle C Maddix, Syama Rangapuram, David Salinas, Jasper Schulz, et al. 2019. Gluonts: Probabilistic time series models in python. *arXiv preprint arXiv:1906.05264* (2019).
[4] Apostolos Ampountolas. 2021. Modeling and Forecasting Daily Hotel Demand: A Comparison Based on SARIMAX, Neural Networks, and GARCH Models. *Forecasting* 3, 3 (2021), 580–595. https://doi.org/10.3390/forecast3030037
[5] George Athanasopoulos, Roman A Ahmed, and Rob J Hyndman. 2009. Hierarchical forecasts for Australian domestic tourism. *International Journal of Forecasting* 25, 1 (2009), 146–166.
[6] Konstantinos Benidis, Syama Sundar Rangapuram, Valentin Flunkert, Bernie Wang, Danielle Maddix, Caner Turkmen, Jan Gasthaus, Michael Bohlke-Schneider, David Salinas, Lorenzo Stella, et al. 2020. Neural forecasting: Introduction and literature overview. *arXiv preprint arXiv:2004.10240* (2020).
[7] Maximilian Christ, Nils Braun, Julius Neuffer, and Andreas W. Kempa-Liehr. 2018. Time Series FeatuRe Extraction on basis of Scalable Hypothesis tests (tsfresh – A Python package). *Neurocomputing* 307 (2018), 72–77. https://doi.org/10.1016/j.neucom.2018.03.067
[8] Amazon Forecast. [n.d.]. CNN-QR Algorithm. https://docs.aws.amazon.com/forecast/latest/dg/aws-forecast-algo-cnnqr.html

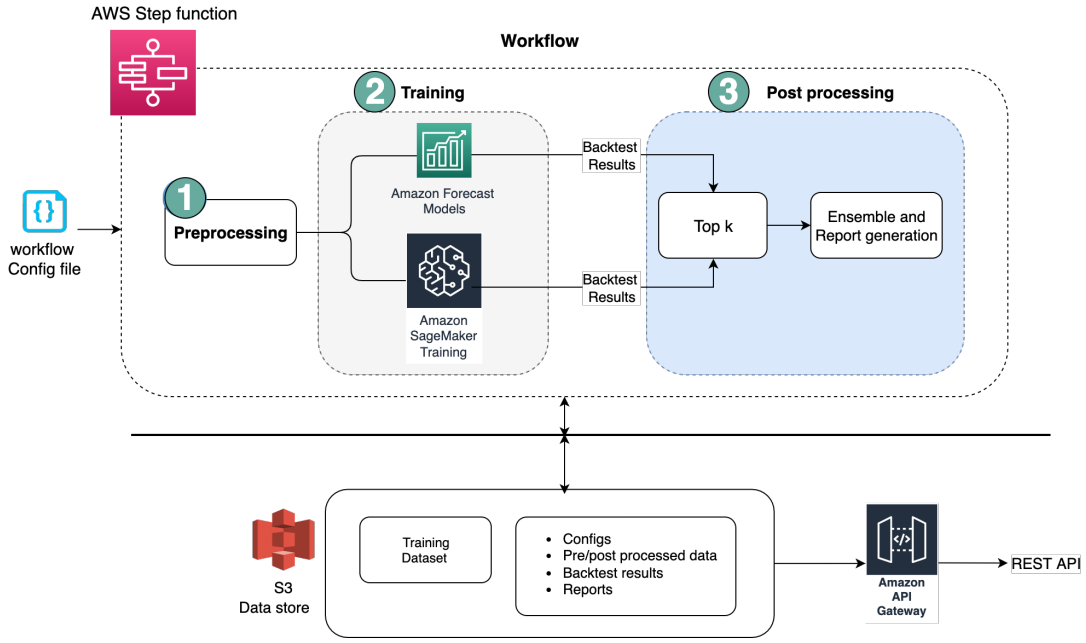

Figure 7: End-to-end forecasting workflow developed on AWS. Model training runs in parallel for efficiency. The whole workflow is implemented as a state-machine using AWS Step Function.

[9] Bosch Global. 2022. Annual report. https://www.bosch.com/company/annual-report/

[10] Prasanth Lade Goktug T. Cinar. 2020. Leveraging Apache Spark to Develop AI-Enabled Products and Services at Bosch. (2020). https://databricks.com/session_na20/leveraging-apache-spark-to-develop-ai-enabled-products-and-services-at-bosch Databricks Spark+AI Summit.

[11] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press. http://www.deeplearningbook.org.

[12] A.R. Hoshmand. 2009. *Business Forecasting: A Practical Approach*. " Routledge".

[13] Rob Hyndman, George Athanasopoulos, Christoph Bergmeir, Gabriel Caceres, Leanne Chhay, Mitchell O'Hara-Wild, Fotios Petropoulos, Slava Razbash, Earo Wang, and Farah Yasmeen. 2022. *forecast: Forecasting functions for time series and linear models*. https://pkg.robjhyndman.com/forecast/ R package version 8.16.

[14] Rob J. Hyndman, Roman A. Ahmed, George Athanasopoulos, and Han L Shang. 2011. Optimal combination forecasts for hierarchical time series. *Computational Statistics and Data Analysis* 55(9) (2011), 2579–2589.

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

[16] Rob J Hyndman and Yeasmin Khandakar. 2008. Automatic time series forecasting: the forecast package for R. *Journal of Statistical Software* 26, 3 (2008), 1–22. https://doi.org/10.18637/jss.v027.i03

[17] Pooja Kaunchi, Tushar Jadhav, Yogesh Dandawate, and Pankaj Marathe. 2021. Future Sales Prediction For Indian Products Using Convolutional Neural Network-Long Short Term Memory. In *2021 2nd Global Conference for Advancement in Technology (GCAT)*. 1–5. https://doi.org/10.1109/GCAT52182.2021.9587668

[18] Sungil Kim and Heeyoung Kim. 2016. A new metric of absolute percentage error for intermittent demand forecasts. *International Journal of Forecasting* 32, 3 (2016), 669–679. https://doi.org/10.1016/j.ijforecast.2015.12.003

[19] Shiyang Li, Xiaoyong Jin, Yao Xuan, Xiyou Zhou, Wenhu Chen, Yu-Xiang Wang, and Xifeng Yan. 2019. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. *Advances in Neural Information Processing Systems* 32 (2019).

[20] Bryan Lim, Sercan Ö. Arık, Nicolas Loeff, and Tomas Pfister. 2021. Temporal Fusion Transformers for interpretable multi-horizon time series forecasting. *International Journal of Forecasting* 37, 4 (2021), 1748–1764. https://doi.org/10.1016/j.ijforecast.2021.03.012

[21] Bryan Lim and Stefan Zohren. 2021. Time-series forecasting with deep learning: a survey. *Philosophical Transactions of the Royal Society A* 379, 2194 (2021), 20200209.

[22] Mark Masse. 2011. *REST API design rulebook: designing consistent RESTful web service interfaces*. " O'Reilly Media, Inc.".

[23] Pablo Montero-Manso, George Athanasopoulos, Rob J Hyndman, and Thiyanga S Talagala. 2020. FFORMA: Feature-based Forecast Model Averaging. *International Journal of Forecasting* 36(1) (2020), 86–92.

[24] Konstantinos Nikolopoulos, Sushil Punia, Andreas Schäfers, Christos Tsinopoulos, and Chrysovalantis Vasilakis. 2021. Forecasting and planning during a pandemic: COVID-19 growth rates, supply chain disruptions, and governmental decisions. *European journal of operational research* 290, 1 (2021), 99–115.

[25] Syama Sundar Rangapuram, Lucien D Werner, Konstantinos Benidis, Pedro Mercado, Jan Gasthaus, and Tim Januschowski. 2021. End-to-End Learning of Coherent Probabilistic Forecasts for Hierarchical Time Series. In *Proceedings of the 38th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 139)*, Marina Meila and Tong Zhang (Eds.). PMLR, 8832–8843. http://proceedings.mlr.press/v139/rangapuram21a.html

[26] Youness H. Sagheer A, Hamdoun H. 2021. Deep LSTM-Based Transfer Learning Approach for Coherent Forecasts in Hierarchical Time Series. Sensors (Basel). *21(13):4379. doi: 10.3390/s21134379. PMID: 34206750; PMCID: PMC8271891*. (2021).

[27] David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. 2020. DeepAR: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting* 36, 3 (2020), 1181–1191.

[28] Souhaib Ben Taieb, James W. Taylor, and Rob J. Hyndman. 2017. Coherent Probabilistic Forecasts for Hierarchical Time Series. In *Proceedings of the 34th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 70)*, Doina Precup and Yee Whye Teh (Eds.). PMLR, 3348–3357. https://proceedings.mlr.press/v70/taieb17a.html

[29] Yuichi Takano Tomokaze Shiratori, Ken Kobayashi. 2020. Prediction of hierarchical time series using structured regularization and its application to artificial neural networks. *arXiv:2007.15159* (2020).

[30] Ruofeng Wen, Kari Torkkola, Balakrishnan Narayanaswamy, and Dhruv Madeka. 2017. A Multi-Horizon Quantile Recurrent Forecaster. *arXiv: Machine Learning* (2017).

[31] R.I. Whitfield and A.H.B. Duffy. 2013. Extended revenue forecasting within a service industry. *International Journal of Production Economics* 141, 2 (2013), 505–518. https://doi.org/10.1016/j.ijpe.2011.11.015 Special Issue on Service Science.

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

[33] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. 2021. Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting. In *The Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021*. AAAI Press, online.