A machine learning approach for forecasting hierarchical time series

Paolo Mancuso, Veronica Piccialli, Antonio M. Sudoso

University of Rome Tor Vergata

Abstract

In this paper, we propose a machine learning approach for forecasting hierarchical time series. Rather than using historical or forecasted proportions, as in standard top-down approaches, we formulate the disaggregation problem as a non-linear regression problem. We propose a deep neural network that automatically learns how to distribute the top-level forecasts to the bottom level-series of the hierarchy, keeping into account the characteristics of the aggregate series and the information of the individual series. In order to evaluate the performance of the proposed method, we analyze hierarchical sales data and electricity demand data. Besides comparison with the top-down approaches, the model is compared with the bottom-up method and the optimal reconciliation method. Results demonstrate that our method does not only increase the average forecasting accuracy of the hierarchy but also addresses the need of building an automated procedure generating coherent forecasts for many time series at the same time.

Keywords: Hierarchical Time Series, Forecast, Machine Learning, Neural Network

1. Introduction

A hierarchical time series is a collection of time series organized in a hierarchical structure that can be aggregated at different levels [7]. As an example, Stock Keeping Unit (SKU) sales aggregate up to product subcategory sales, which further aggregate to product categories. In order to support decision-making at different levels of the hierarchy, a challenging task is the generation of coherent forecasts. Forecasts of the individual series must sum up in a proper way across the levels preserving the hierarchical structure.

For example, forecasts of regional sales should sum up to give forecasts of state sales, which should in turn sum up to give a forecast for the national sales.

In the literature two main lines of research are pursued: bottom-up and top-down approaches. Top-down approaches involve forecasting first the top-level series and then disaggregating by means of historical [5] o forecasted proportion [2] to get forecasts for the lower level series. On the other hand, the bottom-up approach produces first forecasts for the bottom-level time series and then aggregates them in order to get the forecasts for the higher level time series. Both classes of methods have their own advantages, since top-down approaches perform well when the top-level series is easy to forecast, whereas the bottom-up method presents the advantage that the pattern of each series is accurately identified and consequently forecasted without loss of information. However, the bottom-up approach ignores correlations among the series, and this may lead to poor aggregate forecasts with respect to top-down approaches [17]. In general, a bottom-up approach should be preferred whenever the forecasts are employed to support decisions that are mainly related to the bottom rather than the top of the hierarchy, whereas a top-down approach performs better when the bottom-level series are too noisy [4]. The objective to reconcile forecasts at all levels of the hierarchy, from the top to the bottom, has lead researchers to investigate the impact that the association between bottom-level series produce on the aggregation [15]. Analytical approaches to the forecast reconciliation problem have been proposed by Hyndman et al. (2011, 2014) [9, 6] and by Wickramasuriya et al. (2019) [19]. These methods not only ensure that forecasts are coherent but also lead to improvements in forecast accuracy. However, a shortcoming of these methods is that they involve two stages, with forecasts first produced independently for each series in the hierarchy, and then optimally combined to satisfy the aggregation constraint.

In this paper, we propose a new top-down approach for forecasting hierarchical time series. We formulate the disaggregation problem as a non-linear regression problem and we solve it with a deep neural network that jointly learns how to disaggregate and generate coherent forecasts across the levels of the hierarchy. Our approach is successful at capturing the relations between the series to disaggregate thanks to the neural network's ability to extract meaningful features from the aggregate series and combine them with the dynamics of the individual series. We test our method on two real-world datasets with completely different characteristics: the first one comes from

sales data and has noisy and intermittent bottom-level series, the second one comes from electricity demand data and has more regular bottom-level series. Our numerical experiments show that in both cases our method increases the average forecasting accuracy of the hierarchy outperforming state-of-the-art approaches. The rest of the paper is organized as follows. Section 2 discusses the concept of hierarchical time series and the methods of hierarchical forecasting. Section 3 contains the detail of the proposed machine learning algorithm. Section 4 describes the basic forecasting methods employed in the hierarchical models and the experimental setup. Section 5 discusses the datasets and the numerical experiments conducted to evaluate the proposed method. Finally, Section 6 concludes the paper.

2. Hierarchical Time Series

In a general hierarchical structure with K levels, level 0 is defined as the completely aggregated series. Each level from 1 to $K-2$ denotes a further disaggregation down to level $K-1$ containing the most disaggregated time series. In a hierarchical time series, the observations at higher levels can be obtained by summing up the series below. Let $\boldsymbol{y_t^k}$ be the vector of all observations at level $k = 1, \ldots, K-1$ and $t = 1, \ldots, T$, then we define the vector of all observations of the hierarchy:

$$\boldsymbol{y_t} = \begin{pmatrix} y_t^0 \\ \boldsymbol{y_t^1} \\ \vdots \\ \boldsymbol{y_t^{K-1}} \end{pmatrix},$$

where y_t^0 is the observation of the series at the top and the vector $\boldsymbol{y_t^{K-1}}$ contains the observations of the series at the bottom of the hierarchy. The structure of the hierarchy is determined by the summing matrix $\boldsymbol{S}$ that defines the aggregation constraints:

$$\boldsymbol{y_t} = \boldsymbol{S}\boldsymbol{y_t^{K-1}}.$$

The summing matrix $\boldsymbol{S}$ is a $0-1$ matrix of size $M \times m_{K-1}$, where m_k is the number of series at level k and $M = \sum_{k=0}^{K-1} m_k$ is the total number of series in the hierarchy.

Given observations at time $t = 1, ..., T$ and the forecasting horizon h, the aim is to forecast each series at each level at time $t = T+1, ..., T+h$.

The current methods of forecasting hierarchical time series are: top-down, bottom-up, middle-out and optimal reconciliation [7]. The main objective of such approaches is to ensure that forecasts are coherent across all the levels of the hierarchy. Regardless of the methods used to forecast the time series for the different levels of the hierarchy, the individual forecasts must be reconciled to be useful for any subsequent decision making. Forecast reconciliation is the process of adjusting forecasts to make them coherent. By definition, a forecast is coherent if it satisfies the aggregation constraints defined by the summing matrix.

2.1. Bottom-up Approach

The bottom-up approach focuses on producing the h-step-ahead base forecasts for each series at the lowest level $\hat{\boldsymbol{y}}_{\boldsymbol{h}}^{\boldsymbol{K-1}}$ and aggregating them to the upper levels of the hierarchy according to the summing matrix. It can be represented as follows:

$$\tilde{\boldsymbol{y}}_{\boldsymbol{h}} = \boldsymbol{S}\hat{\boldsymbol{y}}_{\boldsymbol{h}}^{\boldsymbol{K-1}},$$

where $\tilde{\boldsymbol{y}}_{\boldsymbol{h}}$ is the vector of coherent h-step-ahead forecasts for all series of the hierarchy. An advantage of this approach is that we directly forecast the series at the bottom-level and no information is lost due to aggregation. On the other hand, bottom-level series can be quite noisy and more challenging to model and forecast. This approach also has the disadvantage of having many time series to forecast if there are many series at the lowest level.

2.2. Top-down Approaches

Top-down approaches first involve generating the base forecasts for the total series and then disaggregating these downwards to get coherent forecasts for each series of the hierarchy. The disaggregation of the top-level forecasts is usually achieved by using the proportions $\boldsymbol{p} = (p_1, ..., p_{m_{K-1}})^{\mathsf{T}}$, which represent the relative contribution of the bottom-level series to the top-level aggregate. The two most commonly used top-down approaches are the Average Historical Proportions (AHP) and the Proportions of the Historical Averages (PHA). In the case of the AHP, the proportions are determined as follows:

$$p_i = \frac{1}{T}\sum_{t=1}^{T}\frac{y_{t,i}^{K-1}}{y_t^0}, \quad i = 1, \ldots, m_{K-1}.$$

In the PHA approach, the proportions are determined in the following manner:

$$p_i = \frac{\sum_{t=1}^{T} \frac{y_{t,i}^{K-1}}{T}}{\sum_{t=1}^{T} \frac{y_t^0}{T}}, \quad i = 1, \ldots, m_{K-1}.$$

For these two methods, once the bottom-level h-step-ahead forecasts have been generated, these are aggregated to generate coherent forecasts for the rest of the series of the hierarchy by using the summing matrix. Given the vector of proportions $\boldsymbol{p}$, top-down approaches can be represented as:

$$\tilde{\boldsymbol{y}}_{\boldsymbol{h}} = \boldsymbol{S}\boldsymbol{p}\hat{y}_h^0.$$

Top-down approaches based on historical proportions usually produce less accurate forecasts at lower levels of the hierarchy than bottom-up approaches because they don't take into account that these proportions may change over time. To address this issue, instead of using the static proportions as in AHP and PHA, Athanasopoulos et al. (2009) propose in [2] the Forecasted Proportion (FP) method in which proportions are based on forecasts rather than on historical data. It first generates an independent base forecast for all series in the hierarchy, then for each level, from the top to the bottom, the proportion of each base forecast to the aggregate of all the base forecasts at that level are calculated. For a hierarchy with K levels we have:

$$p_i = \prod_{k=0}^{K-2} \frac{\hat{y}_{t,i}^k}{\hat{\sigma}_{t,i}^{k+1}}, \quad i = 1, ..., m_{K-1}.$$

where $\hat{y}_{t,i}^k$ is the base forecast of the series that corresponds to the node which is k levels above node i, and $\hat{\sigma}_{t,i}^{k+1}$ is the sum of the base forecasts below the series that is k levels above node i and directly in contact with that series.

2.3. Middle-out Approach

The middle-out method can be seen as a combination of the top-down and bottom-up approaches. It combines ideas from both methods by starting from a middle level where forecasts are reliable. For the series above the middle level, coherent forecasts are generated using the bottom-up approach by aggregating these forecasts upwards. For the series below the middle level, coherent forecasts are generated using a top-down approach by disaggregating the middle level forecasts downwards.

2.4. Optimal Reconciliation

In [9] Hyndman et al. (2011) propose a novel approach that provides optimal forecasts that are better than forecasts produced by either a top-down or a bottom-up approach. Their proposal is based on independently forecasting all series at all levels of the hierarchy and then using a linear regression model to optimally combine and reconcile these forecasts. Their approach is based on a generalized least squares estimator that requires an estimate of the covariance matrix of the errors that arise due to incoherence. In [19] Wickramasuriya et al. (2019) show that this matrix is impossible to estimate in practice and they propose a state-of-the-art forecast reconciliation approach, called Minimum Trace (MinT) that incorporates the information from a full covariance matrix of forecast errors in obtaining a set of coherent forecasts. MinT minimizes the mean squared error of the coherent forecasts across the entire hierarchy with the constraint of unbiasedness. The resulting revised forecasts are coherent, unbiased and have minimum variance amongst all combination forecasts. An advantage of the optimal reconciliation approach is that allows for the correlations between the series at each level using all the available information within the hierarchy. However, it is computationally expensive compared to the other methods introduced so far because it requires to individually forecast all the time series at all the levels of the hierarchy.

3. Neural Network Disaggregation

As observed in [7], standard top-down approaches have the disadvantage of information loss since they are unable to capture the individual time series characteristics. Departing from the related literature, to the best of our knowledge, we propose a new top-down approach which first generates a good forecast for the aggregated time series at some level of the hierarchy and then disaggregates it downwards, without loss of information, by means of a machine learning algorithm. In order to explain the proposed algorithm, we focus on two consecutive levels with the top-level time series being at node j of level k and the bottom-level series at level $k+1$ (see Figure 1).

Let m_j^{k+1} be the number of series at level $k+1$ connected to the parent node j at level k, then we model the disaggregation procedure as a non-linear regression problem:

$$\boldsymbol{y_t^{k+1,j}} = f(y_{t,j}^{k,p}, y_{t-1,j}^{k,p}, \ldots, y_{t-l,j}^{k,p}, \boldsymbol{x_{t,1}}, \ldots, \boldsymbol{x_{t,m_j^{k+1}}}) + \boldsymbol{\epsilon},$$

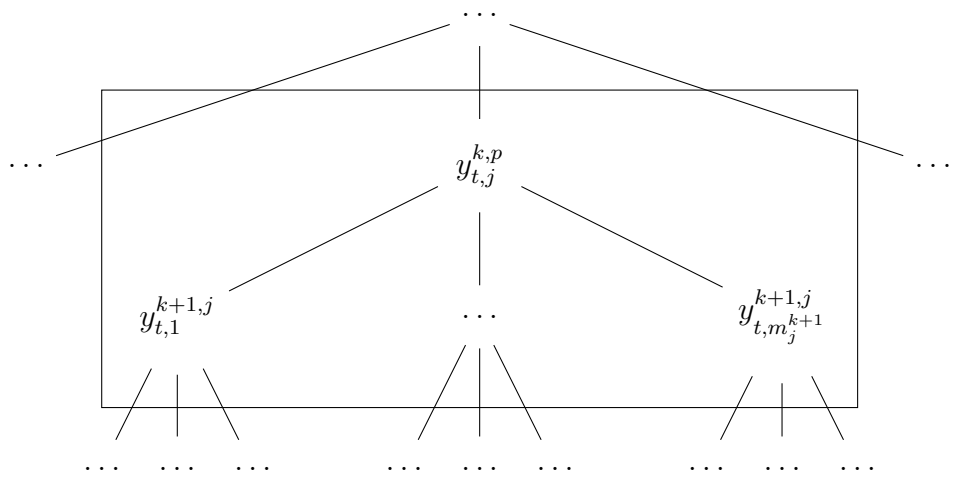

Figure 1: A top-level series at level k and the bottom-level series at level $k+1$.

where $\boldsymbol{y_t^{k+1,j}}$ is the vector of size m_j^{k+1} containing the series at level $k+1$, $y_{t,j}^{k,p}$ is the aggregate time series corresponding to the node j at level k connected to the parent node p at level $k-1$, l is the number of the lagged time steps of the aggregated series, $\boldsymbol{x_{t,i}}$ is a vector of the external regressors for each series at level $k+1$, f is a non-linear function learned, in our case, by a feed-forward neural network and $\boldsymbol{\epsilon}$ is the error term.

Given any aggregate time series $y_{t,j}^{k,p}$ and the vector of series $\boldsymbol{y_t^{k+1,j}}$, the algorithm is made up of two steps: in the first one the best forecasting model for the aggregated time series is chosen, and the neural network is trained with the real values of the training set of the two levels time series; in the second step, forecasts for the aggregated time series are fed to the neural network in order to obtain forecasts for all the lower level time series. The flow chart of the proposed algorithm is shown in Figure 2. More in detail, the two steps are the following:

Step 1 In the training phase, the best forecasting model F^* for the time series $y_{t,j}^{k,p}$ is chosen on the basis of the training set. At the same time, the neural network is trained taking as input the training set of $y_{t,j}^{k,p}$ with lagged time steps and the explanatory variables $\boldsymbol{x_{t,i}}$ relative to the training set of $\boldsymbol{y_t^{k+1,j}}$. The output are the true values of the disaggregated time series $\boldsymbol{y_t^{k+1,j}}$. In order to simplify the notation, from now on we refer to the produced model as NND (Neural Network Disaggregation).

Step 2 In the disaggregation or test phase, forecasts $\hat{y}_{t,j}^{k,p}$ relative to the time period of the test set are generated by the model F^*. Finally, these forecasts are fed to the trained NND in order to produce the disaggregated forecasts $\hat{\boldsymbol{y}}_{\boldsymbol{t}}^{\boldsymbol{k+1,j}}$ for the test set.

In general, the learned function f generates base forecasts that are not coherent since they do not sum up correctly according to the structure of the hierarchy. In order to ensure that forecasts are reconciled across the hierarchy, we want f to output a set of forecasts that are as close as possible to the base forecasts, but also meet the requirement that forecasts at upper levels in the hierarchy are the sum of the associated lower level forecasts. From an optimization perspective, we want to introduce an equality constraint to the regression problem in such a way that we can still use backpropagation to train the network. More in detail, we are looking for the network weights such that the Mean Squared Error (MSE) between the true values and the predictions is minimized and, in addition, we want the following constraint to hold:

$$y_{t,j}^{k,p} = \mathbf{1}^{\mathsf{T}} \boldsymbol{y}_{\boldsymbol{t}}^{\boldsymbol{k+1,j}} = \mathbf{1}^{\mathsf{T}} \hat{\boldsymbol{y}}_{\boldsymbol{t}}^{\boldsymbol{k+1,j}} = \hat{y}_{t,j}^{k,p},$$

where $\mathbf{1}$ is the vector of all ones of size m_j^{k+1}.

We impose the aggregation constraint by adding a term to the MSE loss function that penalizes differences between the sum of lower level observations and the sum of the lower level forecasts:

$$\begin{aligned} L(\boldsymbol{y}_{\boldsymbol{t}}^{\boldsymbol{k+1,j}}, \hat{\boldsymbol{y}}_{\boldsymbol{t}}^{\boldsymbol{k+1,j}}) = \frac{1}{T}\Bigg[(1-\alpha)\sum_{t=1}^{T} ||\boldsymbol{y}_{\boldsymbol{t}}^{\boldsymbol{k+1,j}} - \hat{\boldsymbol{y}}_{\boldsymbol{t}}^{\boldsymbol{k+1,j}}||^2 + \\ +\alpha\sum_{t=1}^{T}(\mathbf{1}^{\mathsf{T}}\boldsymbol{y}_{\boldsymbol{t}}^{\boldsymbol{k+1,j}} - \mathbf{1}^{\mathsf{T}}\hat{\boldsymbol{y}}_{\boldsymbol{t}}^{\boldsymbol{k+1,j}})^2\Bigg], \end{aligned} \tag{1}$$

where $\alpha \in (0,1)$ is a parameter that controls the relative contribution of each term in the loss function. There is a compromise between minimizing the MSE and satisfying the aggregation constraint. A too small value of α will result in the corresponding constraint being ignored, producing, in general, not coherent forecasts whereas a too large value will cause the MSE being ignored, producing coherent but possibly inaccurate base forecasts. The idea is to balance the contribution of both terms by setting $\alpha = 0.5$. In principle,

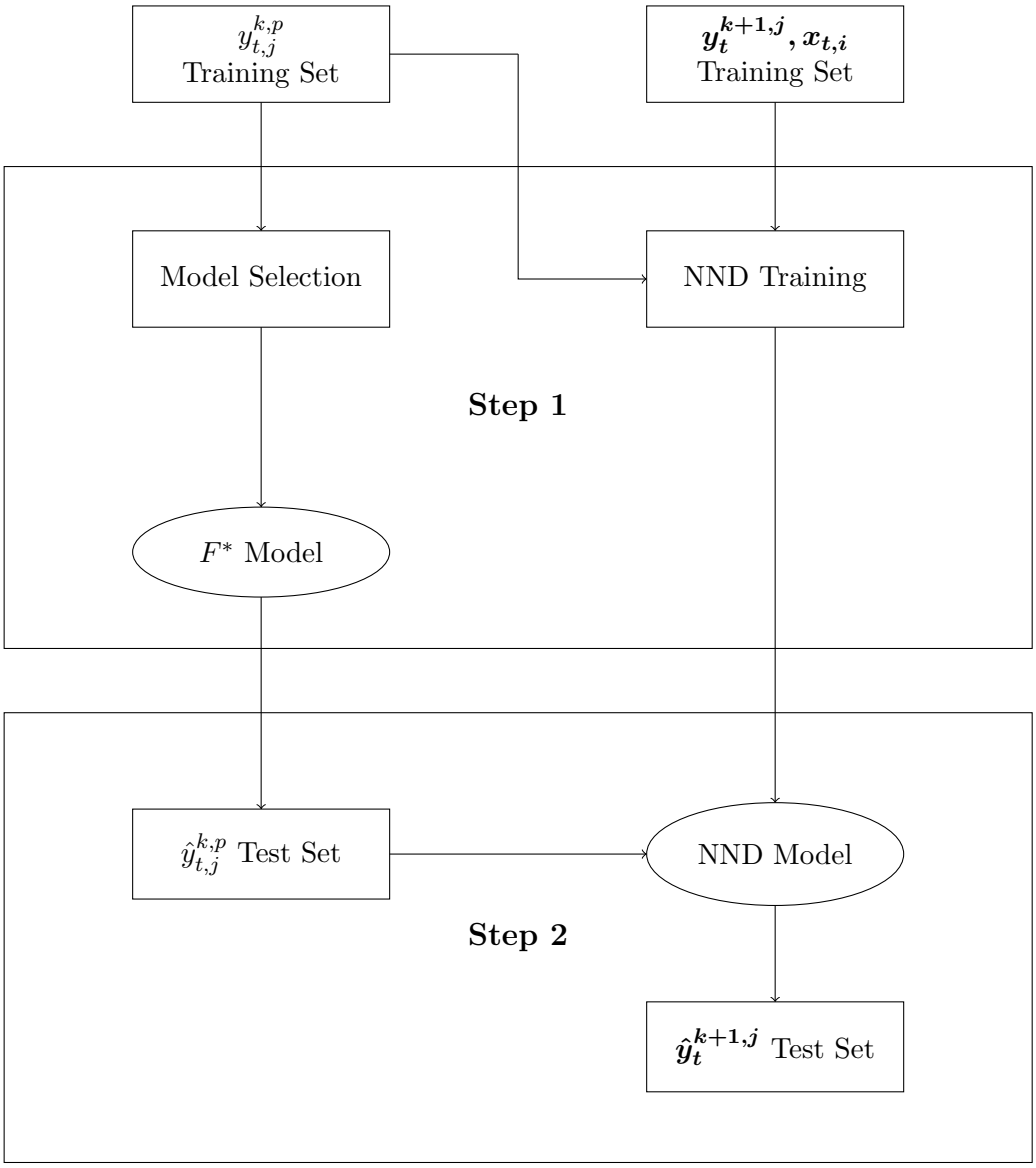

Figure 2: Decomposition of the aggregated forecast through a neural network: Neural Network Disaggregation (NND).

the parameter α may be tuned on each instance. However, if the forecasts were perfect the coherence constraint would be almost satisfied, since the true values are coherent by construction. This explains why even with $\alpha = 0$, the violation of coherence is in practice relatively small and with $\alpha = 0.5$ the violation is basically zero in all our experiments. For this reason, we did not investigate the tuning of parameter α and kept it fixed to 0.5.

Top-down approaches distribute the top-level forecasts down the hierarchy using historical or forecasted proportions of the data. In our case, explicit proportions are never calculated since the algorithm automatically learns how to disaggregate forecasts of the top-level series to the bottom-level series without loss of information. Furthermore, our method is flexible enough to be employed in the forecasting process of the whole hierarchy in two different ways:

1. *Standard top-down:* a forecasting model F^* is developed for the aggregate at level 0, and a single disaggregation model NDD is trained with the series at level 0 and $K-1$. Therefore, forecasts for the bottom-level series are produced by looking only at the aggregated series at level 0. Then, the bottom-level forecasts are aggregated to generate coherent forecasts for the rest of the series of the hierarchy.
2. *Iterative top-down:* the forecasting model F^* for an aggregate at level k is the disaggregation model NDD trained with the series at level $k-1$ and k, for each $k = 1, \ldots, K-1$. At level 0, instead, F^* is the best model selected among a set of standard forecasting methods. Forecast for all the levels are then obtained by feeding forecasts to the disaggregation models at each level.

The difference between the two approaches is that in the standard top-down, bottom-level forecasts are generated with only one disaggregation model, whereas in the iterative version, a larger number of disaggregation models is trained, one for each series to be disaggregated. To be more precise, to disaggregate the m_k series at level $k = 0, \ldots, K-2$, exactly m_k disaggregation models are trained in parallel.

We also notice that this algorithm can be easily plugged into a middle-out strategy: a forecasting model is developed for each aggregate at a convenient level, and the disaggregation models are trained and tested to distribute these forecasts to the series below. For the series above the middle level, coherent forecasts are generated using the bottom-up approach.

Regarding the choice of the neural network architecture, our objective is to include in the model the relationship between explanatory variables derived from the lower level series, and the features of the aggregate series that describe the structure of the hierarchy. For this reason, we propose a deep neural network that is capable of accepting and combining multiple types of input, including cross-sectional and time series data, in a single end-to-end model. More precisely, the model is made up of two branches: the first branch is a simple Multi-Layer Perceptron (MLP) designed to handle the explanatory variables $\boldsymbol{x_{t,i}}$ such as price, promotions, day of the week, or in general, special events affecting the time series of interest; the second branch is a one-dimensional Convolutional Neural Network (CNN) that extracts feature maps over fixed segments of the aggregate series $y_{t,j}^{k,p}$. CNNs can create useful representations of time series automatically, they are highly noise-resistant models, and they are able to extract very informative, deep features, which are independent from time [13]. Features extracted from the two subnetworks are then concatenated together to form the final input of the multi-output regression model (see Figure 3). The output layer of the model is a standard regression layer with linear activation function where the number of units is equal to the number of the series to forecast.

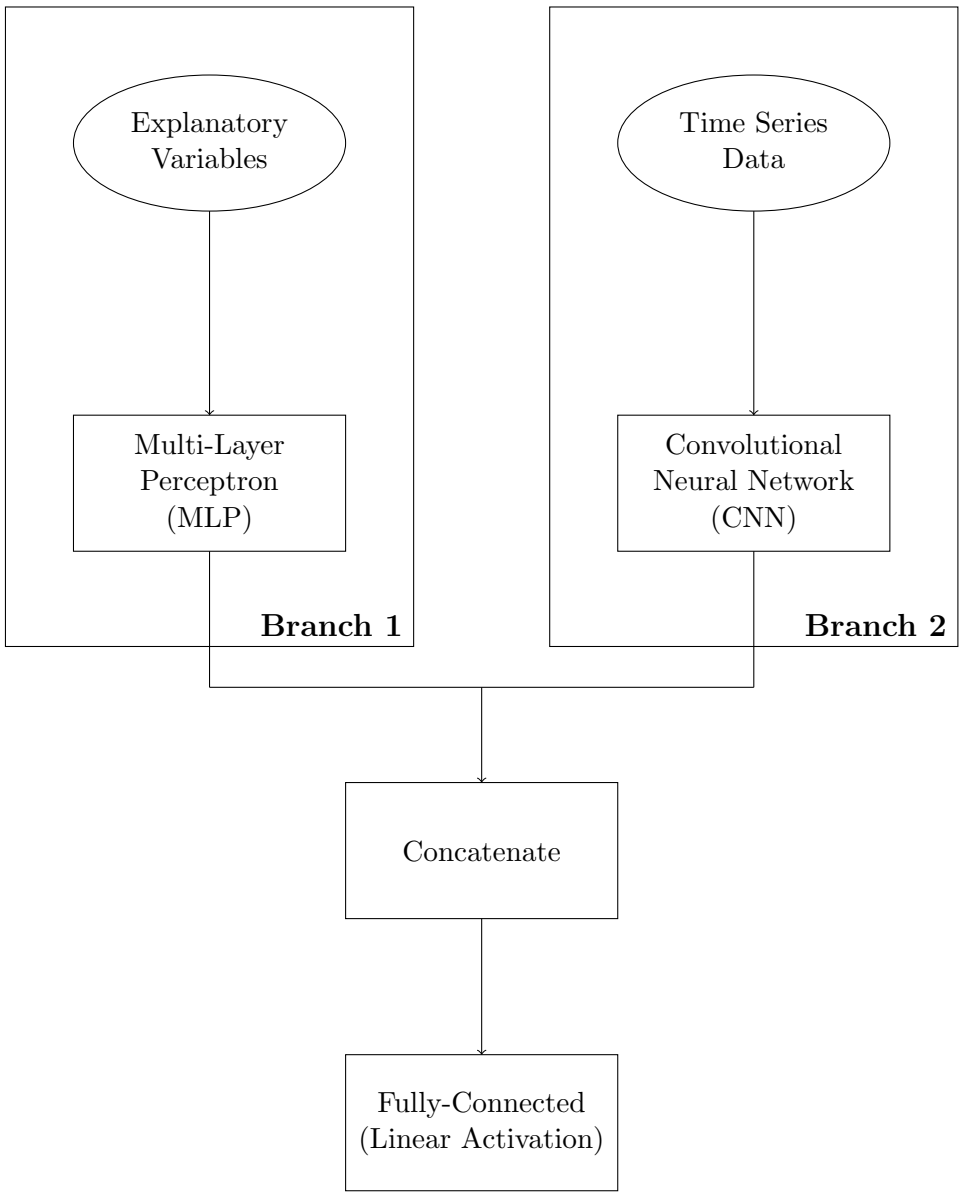

Figure 3: Our model has one branch that accepts the numerical data (left) and another branch that accepts time series data (right).

4. Experimental Setup

In this section, we resume first the forecasting models we use to generate base forecasts for the hierarchical approaches, then we describe our strategy to select the best forecasting model and the implementation details.

4.1. Forecasting Models

In order to describe the methods, let $(y_1, \ldots, y_T)$ be an univariate time series of length T and $(y_{T+1}, \ldots, y_{T+h})$ the forecasting period, where h is the forecast horizon. We consider the following models:

1. *Naive*
2. *Autoregressive Integrated Moving Average (ARIMA)*
3. *Exponential Smoothing (ETS)*
4. *Non-linear autoregression model (NAR)*
5. *Dynamic regression models*: univariate time series models, such as linear and non-linear autoregressive models, allow for the inclusion of information from past observations of a series, but not for the inclusion of other information that may also affect the time series of interest. Dynamic regression models allow keeping into account the time-lagged relationship between the output and the lagged observations of both the time series itself and of the external regressors. More in detail, we consider two types of dynamic regression models:
   (a) *ARIMA model with exogeneous variables (ARIMAX)*
   (b) *NAR model with exogenous variables (NARX)*

In the literature it has been pointed out that the performance of forecasting models could be improved by suitably combining forecasts from standard approaches [18]. An easy way to improve forecast accuracy is to use several different models on the same time series, and to average the resulting forecasts. We consider two ways of combining forecasts:

1. *Simple Average:* the most natural approach to combine forecasts is to use the mean. The composite forecast in case of simple average is given by $\hat{y}_t = \frac{1}{m}\sum_{i=1}^{m} \hat{y}_{i,t}$ for $t = T+1, ..., T+h$ where h is the forecast horizon, m is the number of combined models and $\hat{y}_{i,t}$ is the forecast at time t generated by model i.

2. *Constrained Least Squares Regression:* in this setting, the composed forecast is not a function of m only as in the simple average but is a linear function of the individual forecasts whereby the parameters are determined by solving an optimization problem. The approach proposed by Timmermann in [18] minimizes the sum of squared errors under some additional constraints. Specifically, the estimated coefficients β_i are constrained to be non-negative and to sum up to one. The weights obtained are easily interpretable as percentages devoted to each of the individual forecasts. Given the optimal weights, the composed forecast is obtained as $\hat{y}_t = \sum_{i=1}^{m} \beta_i \hat{y}_{i,t}$ for $t = T+1, ..., T+h$. From the mathematical point of view the following optimization problem needs to be solved:

$$\begin{aligned} \min \quad & \sum_{t=T+1}^{T+h} (y_t - \sum_{i=1}^{m} \beta_i \hat{y}_{i,t})^2 \\ s.t. \quad & \beta_i \geq 0 \qquad\qquad i = 1, ..., m \\ & \sum_{i=1}^{m} \beta_i = 1 \end{aligned} \tag{2}$$

Differently from the simple average which does not need any training as the weights are a function of m only, with this method we need to allocate a reserved portion of forecasts in order to train the meta-model.

In particular, we consider two following composite models:

1. Combination of ARIMAX, NARX and ETS forecasts obtained through the simple mean.
2. Combination of ARIMAX, NARX and ETS forecasts obtained by solving the constrained least squares problem.

We choose to combine the two dynamic regression models with the exponential smoothing in order to take directly into account the effect of the explanatory variables and the presence of linear and non-linear patterns in the series.

4.2. Model Selection

Following an approach widely employed in the machine learning literature, we separate the available data into two sets, training (in-sample) and

test (out-of-sample) data. The training data $(y_1, \ldots, y_N)$, a time series of length N, is used to estimate the parameters of a forecasting model and the test data $(y_{N+1}, \ldots, y_T)$, that comes chronologically after the training set, is used to evaluate its accuracy.

To achieve a reliable measure of model performance, we implement on the training set a procedure that applies a cross-validation logic suitable for time series data. In the expanding window procedure described in [7], the model is trained on a window that expands over the entire history of the time series and it is repeatedly tested against a forecasting window without dropping older data points. This method produces many different train/test splits and the error on each split is averaged in order to compute a robust estimate of the model error (see Figure 4). The implementation of the expanding window procedure requires four parameters:

- *Starting window:* the number of data points included in the first training iteration.
- *Ending window:* the number of data points included in the last training iteration.
- *Forecasting window:* number of data points included for forecasting.
- *Expanding steps:* the number of data points added to the training time series from one iteration to another.

For each series, the best performing model after the cross-validation phase is retrained using the in-sample data and forecasts are obtained recursively over the out-of-sample period. The above procedure requires a forecast error measure. We consider the Mean Absolute Scaled Error (MASE) proposed by Hyndman and Koehler in [11]:

$$MASE = \frac{\frac{1}{h}\sum_{i=T+1}^{T+h} |y_i - \hat{y}_i|}{\frac{1}{T-m}\sum_{t=m+1}^{T} |y_t - y_{t-m}|},$$

where the numerator is out-of-sample Mean Absolute Error (MAE) of the method evaluated across the forecast horizon h, and the denominator is the in-sample one-step ahead Naive forecast with seasonal period m.

Figure 4: Expanding window procedure.

We also consider the Symmetric Mean Absolute Percentage Error (SMAPE) defined as follows:

$$SMAPE = \frac{2}{h} \sum_{i=T+1}^{T+h} \frac{|y_i - \hat{y}_i|}{|y_i| + |\hat{y}_i|}.$$

The SMAPE is easy to interpret, it has an upper bound of 2 when either actual or predicted are zero or when actual and predicted are opposite signs. However, the significant disadvantage of SMAPE is that it produces infinite or undefined values where the actual values are zero or close to zero. The MASE and SMAPE can be used to compare forecast methods on a single series and, because they are scale-free, to compare forecast accuracy across series. For this reason, we average the MASE and SMAPE values of several series to obtain a measurement of forecast accuracy for the group of series.

4.3. Implementation

Time series models described above are implemented by using the "forecast" package in R [10]. Hierarchical time series forecasting is performed by using the "hts" package in R [8]. The proposed disaggregation method is implemented in Python with TensorFlow, a large-scale machine learning framework [1]. Regarding the training details of the NND, early stopping is employed as a form of regularization to avoid overfitting since it stops the training as soon as the error on the validation set starts to grow [3]. The neural network is trained with Adam optimizer [14], a mini-batch stochastic

gradient descent algorithm. Grid search is used to perform the hyperparameter optimization which is simply an exhaustive search through a manually specified subset of points in the hyperparameter space of the neural network. The configuration of the hyperparameters is evaluated on the validation set and optimal values are chosen the size of the mini-batch, number of layers and units for the MLP subnetwork, and convolutional filters and kernel size for the CNN subnetwork. The training time of a single disaggregation model requires order of minutes on a commercial GPU depending on the network dimension and on the granularity of the dataset.

5. Numerical Experiments

In this section, we aim to evaluate the effectiveness of our approach, by comparing it with all the hierarchical methods described in Section 2. In order to be as fair as possible in the comparison, we perform model selection among the set of forecasting methods described in Section 4 whenever a base forecast is required. More in detail, this means that different methods may be used for each time series of the hierarchy we are trying to forecast (bottom-level series for the bottom-up approach, aggregate series for all the top-down, all the time series for the optimal reconciliation approach). As for the metrics used for comparison, we use both the MASE and the SMAPE where possible (i.e. where no zeros are present). We consider two datasets, coming from two completely different problems. In both cases, starting from the aggregated series at some level, we aim to exploit our method to increase the forecasting accuracy of the hierarchy using the characteristics of the aggregate series and explanatory variables.

5.1. Datasets

1. **Sales Data:** we analyze sales data gathered from an Italian supermarket. The dataset consists of 118 daily time series, representing the demand of pasta from 01/01/2014 to 31/12/2018. Besides univariate time series data, the quantity sold is integrated by information on the presence or the absence of a promotion (no detail on the type of promotion on the final price is given). Demand time series can be naturally arranged to follow a hierarchical structure. Here, the idea is to build a 3-level structure: at the top of the hierarchy, there is the total or the store-level series obtained by aggregating the brand-level series. At the second level there are the brand-level series (like for instance Barilla)

Level	Number of series	Total series per level
Store	1	1
Brand	4	4
Item	42, 45, 10, 21	118

Table 1: Hierarchy for the sales data.

Level	Number of series	Total series per level
Grid	1	1
Meter	24	24

Table 2: Hierarchy for the electricity data.

obtained by aggregating the individual demand at the item-level and at the third level there are the most disaggregated time series representing the item-level demand (for example the demand of spaghetti Barilla). The completely aggregated series at level 0 is disaggregated into 4 component series at level 1. Each of these series is further subdivided into 42, 45, 10 and 21 series at level 2, the completely disaggregated bottom-level representing the different varieties of pasta for each brand (see Table 1).

2. **Electricity Data**: we analyze a public electricity demand dataset that contains power measurements and meteorological forecasts relative to a set of 24 power meters installed in low-voltage cabinets of the distribution network of the city of Rolle in Switzerland [16]. The dataset contains measurements from 13/01/2018 to 19/01/2019 at the resolution of 10 minutes and includes mean active and reactive power, voltage magnitude, maximum total harmonic distortion for each phase, voltage frequency and the average power over the three phases. We assume that the grid losses are not significant, so the power at the grid connection is the algebraic sum of the connected nodes. Based on the historical measurements, the operator can determine coherent forecasts for all the grid by generating forecasts for the nodal injections individually. We build a 2-level hierarchy in which we aggregate the 24 series of the distribution system at the meter-level to generate the total series at the grid-level (see Table 2).

Summarizing, we have the first dataset with a three level hierarchy and the second one with a two level hierarchy. As for the experimental setup, we have to make some choices for each dataset:

1. **Sales Data:** for each series, as explanatory variables, we add a binary variable representing the presence of promotion if the disaggregation is computed at the item-level or a variable representing the relative number of items in promotion for each brand if the disaggregation is computed at the brand-level. In both cases, dummy variables representing the day of the week and the month are also added to the model. As for the number of lagged observations of the aggregate demand, we consider fixed-length time windows of 30 days with a hop size of 1 day. We consider 4 years from 01/01/2014 to 31/12/2017 for the in-sample period and the last year of data from 01/01/2018 to 31/12/2018 for the out-of-sample period. The experimental setup for the cross-validation procedure is as follows. The starting windows consists of the first three years of data from 01/01/2014 to 31/12/2016. The training window expands over the last year of the training data including daily observations from 01/01/2017 to 31/12/2018. The forecasting window is set to $h = 7$, corresponding to a forecasting horizon of one week ahead. At each iteration, the training window expands by one week to simulate a production environment in which the model is re-estimated as soon as new data are available and to better mimic the practical scenario in which retailing decisions are made every week. In order to evaluate the forecasting accuracy at each level, for this hierarchy we use the average MASE, as recommended by Hyndman in [12], since most of the item-level series are intermittent.
2. **Electricity Data:** for each series, we use the average power over the three phases as target variable and the temperature, horizontal irradiance, normal irradiance, relative humidity, pressure, wind speed and wind direction as explanatory variables. Dummy variables representing the day of the week and the hour of the day are also added to the model. As for the number of lagged observations of the aggregate power, we consider fixed-length time windows of 144 observations with hop size of 10 minutes. We consider 6 months from 13/01/2018 to 13/06/2018 for the training set and the last 6 months from 14/06/2018 to 13/01/2019 for the test set. The configuration of the cross-validation procedure is as follows. The starting window consists of the first three months of

data from 13/01/2018 to 13/03/2018. At each iteration, the training window expands by 24 hours over the last 3 months of the training data including observations from 14/03/2018 to 13/06/2018. The forecasting window is set to $h = 144$, corresponding to a forecasting horizon of 24 hours ahead. We evaluate the forecasting accuracy at each level by using the average MASE and the average SMAPE over all the series of that level, since there are no zero values in these time series.

5.2. Results

In Table 3 we compare the forecasting performance of our method at each level, in both its versions, standard top-down (NND1) and iterative top-down (NND2) with the bottom-up, average historical proportions, proportions of historical averages, forecasted proportions and the optimal reconciliation approach (MinT). We stress that for all the top down approaches, the performance at the most aggregated level are equivalent, and the differences only emerge at the lower levels of the hierarchy, where we really are interested in the comparison.

For the NND1, we directly forecast the demand at the item-level using the aggregate demand at the store-level and then we aggregate the item-level forecasts to obtain the brand-level forecasts. For the NND2, we train a disaggregation model that generates the brand-level forecasts starting from the store-level series and then one NND for each brand-level series to generate forecasts for each item demand of the brand they belong to. In total, for the entire hierarchy, we train one NND at the top-level and 4 NND in parallel at the brand-level.

The bad performance of the bottom-up method can be attributed to the fact that the demand at the most granular level of the hierarchy is often challenging to model and forecast effectively because it is too sparse and erratic. The majority of the item-level time series display sporadic sales including zeros and the promotion of an item does not always correspond to an increase in sales. By using traditional or combination of methods to generate base forecasts for the time series at the lowest level, we end up with flat line forecasts, representing the average demand, failing to account for the seasonality that truly exists but is impossible to identify between the noise. By focusing our attention at the highest or some intermediate level of the hierarchy, we have enough data to build decent models capturing the underlying trend and seasonality. In fact, the aggregation tends to regularize the demand and make it easier to forecast. The only level for which the

Average MASE							
Level	Bottom-up	AHP	PHA	FP	NND1	NND2	Optimal
Store	1.103	0.567	0.567	0.567	0.567	0.567	**0.559**
Brand	1.237	1.413	1.481	1.157	0.862	**0.838**	1.137
Item	1.057	0.934	0.943	0.891	**0.745**	0.811	0.893

Table 3: Average MASE for each aggregation level of sales data. In bold the best performing approach.

optimal reconciliation approach is the best is the top level. As we move down the hierarchy our approach significantly outperforms all the top-down approaches, the bottom-up method and the optimal reconciliation with the NND iterative top-down (NND2) performing best at the brand-level and the NND standard top-down (NND1) performing best at the item-level. This result is reasonable since the iterative top-down involves training and testing multiple disaggregation models and forecasts generated by one model are fed to the model below. As a side effect, when we go down, the forecasting error tends to propagate achieving less accurate performance at the bottom of the hierarchy. In Figure 5 and 6 we show the forecasts generated by the NND for the all the brand-level series. We perform pairwise t-tests to formally test whether forecasts produced by the hierarchical methods are different. The average historical proportions and the proportions of historical averages and are not significantly different (p-value $= 0.765$). The remaining methods are significantly different from each other (p-value $< 10^{-3}$ for each test).

In Table 4 and 5 we present the MASE and SMAPE at each level of the electricity dataset and for each method. We find that all the top-down approaches perform best at the grid-level. The good performance of the bottom-up method with respect to the classical top-down approaches can be attributed to the fact that the series have a strong seasonality, even at the bottom level. Our NND clearly outperforms all the top-down methods, the bottom-up and the optimal combination at the meter-level. In Figure 7 we show the forecasts generated by the NND for some of the meter-level series. In order to identify the pairs of forecasts which are significantly different from each other, we perform pairwise t-test. Our method is significantly different from each of the other methods (p-value $< 10^{-3}$ for each test). The optimal combination and the bottom-up methods are not significantly different (p-value $= 0.599$).

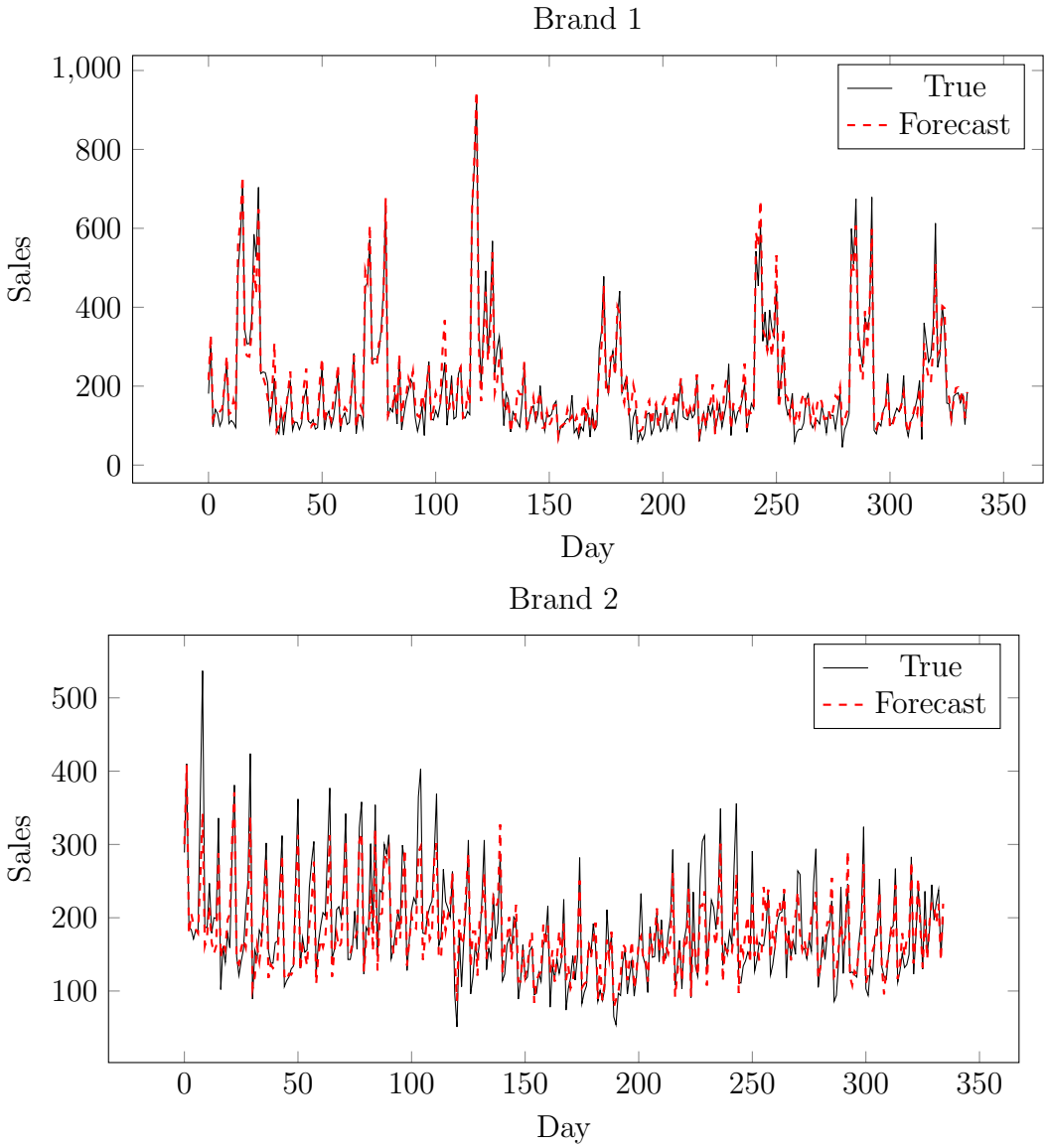

Figure 5: NND out-of-sample forecasts for Brand 1 and Brand 2.

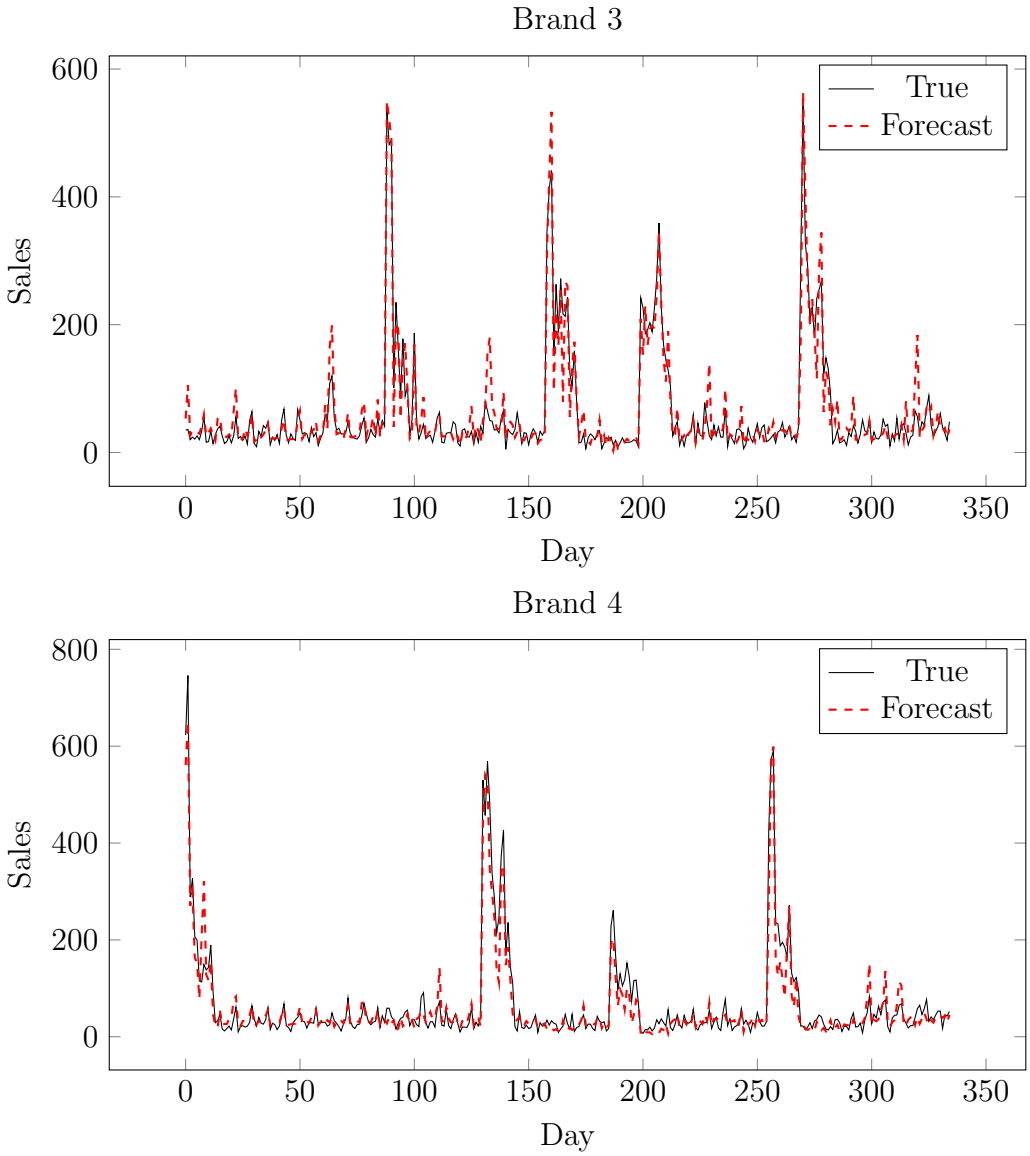

Figure 6: NND out-of-sample forecasts for Brand 3 and Brand 4.

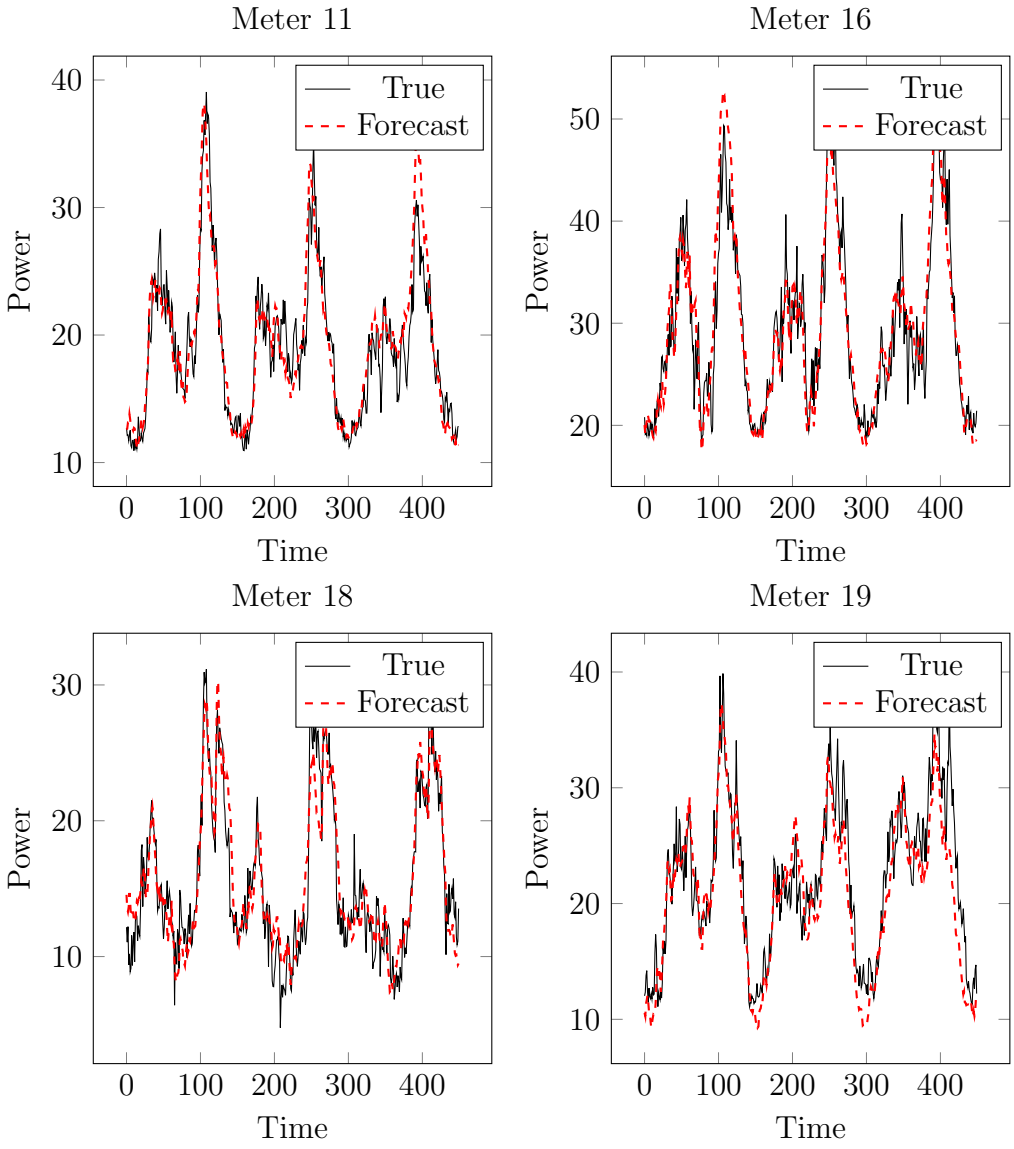

Figure 7: NND out-of-sample forecasts for meter 11, 16, 18 and 19 (first 72 hours).

Average MASE						
Level	Bottom-up	AHP	PHA	FP	NND	Optimal
Grid	1.095	**1.007**	**1.007**	**1.007**	**1.007**	1.087
Meter	1.089	1.371	1.365	1.143	**0.956**	1.112

Table 4: Average MASE for each aggregation level of the electricity dataset. In bold the best performing approach.

Average SMAPE (%)						
Level	Bottom-up	AHP	PHA	FP	NND	Optimal
Grid	7.347	**7.345**	**7.345**	**7.345**	**7.345**	7.349
Meter	16.267	19.245	19.219	16.634	**12.801**	16.323

Table 5: Average SMAPE for each aggregation level of the electricity dataset. In bold the best performing approach.

Summarizing, we get similar results on both datasets, and this is particularly significant due to the different characteristics of the two datasets. In the sales data, the bottom-level series are extremely noisy and hard to forecast (as confirmed by the poor performance of the bottom-up method). On the other hand, the electricity demand data display seasonality at the bottom-level, as confirmed by the good performance of the bottom-up method. In both cases, we generate coherent forecasts (in all our experiments the maximum violation of the aggregation constraint is less then 10^{-3}) and we improve the overall accuracy at any level of the hierarchy. This confirms the general viability of our approach.

6. Conclusions

In this paper, we propose a machine learning method for forecasting hierarchical time series. We formulate the disaggregation problem as a non-linear regression problem and we solve it with a deep neural network. The network architecture is able to jointly learn the structure of the hierarchy and generate coherent forecasts, thanks to the neural network's ability to extract meaningful features from the aggregate series and combine them with the dynamics of the individual series. Furthermore, differently from top-down approaches, our method allows to easily incorporate any external information that affect the time series to disaggregate with no loss of information. Results demonstrate that our method does not only increase the average

forecasting accuracy of the hierarchy but also addresses the need of building an automated procedure generating coherent forecasts for many time series at the same time. Our procedure fulfills the need of scalable algorithms to automate the process of forecasting hierarchical time series with the aim to increase the forecasting accuracy at any level. We stress that, differently from the recently proposed optimal forecast reconciliation approach, in our method, forecast reconciliation is performed inside the learning process without the need of generating base forecast for all the series of the hierarchy. As all the top-down approaches, our method highly relies on accurate top-level forecasts. However, this assumption is often satisfied in hierarchical time series since the top-level series are in general periodic and less noisy (since they are the sum of many sub-level components) compared to individual series at the bottom-level. In summary, our machine learning approach uses all the relevant information available in the hierarchical structure. This is important, as particular aggregation levels may reveal hidden features of the data, not easily identifiable at other levels, that are of interest to the user and needed to be modeled.

References

[1] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., & Shlens, J. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems.

[2] Athanasopoulos, G., Ahmed, R. A., & Hyndman, R. J. (2009). Hierarchical forecasts for australian domestic tourism. *International Journal of Forecasting*, *25*, 146–166.

[3] Caruana, R., Lawrence, S., & Giles, L. (2000). Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping. In *Proceedings of the 13th International Conference on Neural Information Processing Systems* NIPS'00 (pp. 381–387). MIT Press.

[4] Dunn, D. M., Williams, W. H., & Dechaine, T. L. (1976). Aggregate versus subaggregate models in local area forecasting. *Journal of the American Statistical Association*, *71*, 68–71.

[5] Gross, C. W., & Sohl, J. E. (1990). Disaggregation methods to expedite product line forecasting. *Journal of forecasting*, *9*, 233–254.

[6] Hyndman, R., & Athanasopoulos, G. (2014). Optimally reconciling forecasts in a hierarchy. *Foresight: The International Journal of Applied Forecasting*, (pp. 42–48).

[7] Hyndman, R., & Athanasopoulos, G. (2018). *Forecasting: principles and practice*. OTexts.

[8] Hyndman, R., Lee, A., Wang, E., & Wickramasuriya, S. (2018). *hts: Hierarchical and Grouped Time Series*. R package version 5.1.5.

[9] Hyndman, R. J., Ahmed, R. A., Athanasopoulos, G., & Shang, H. L. (2011). Optimal combination forecasts for hierarchical time series. *Computational statistics & data analysis*, *55*, 2579–2589.

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

[11] Hyndman, R. J., & Koehler, A. B. (2006). Another look at measures of forecast accuracy. *International journal of forecasting*, *22*, 679–688.

[12] Hyndman, R. J. et al. (2006). Another look at forecast-accuracy metrics for intermittent demand. *Foresight: The International Journal of Applied Forecasting*, *4*, 43–46.

[13] Ismail Fawaz, H., Forestier, G., Weber, J., Idoumghar, L., & Muller, P.-A. (2019). Deep learning for time series classification: a review. *Data Mining and Knowledge Discovery*, *33*, 917–963.

[14] Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization.

[15] Nenova, Z. D., & May, J. H. (2016). Determining an optimal hierarchical forecasting model based on the characteristics of the data set: Technical note. *Journal of Operations Management*, *44*, 62 – 68.

[16] Nespoli, L., Medici, V., Lopatichki, K., & Sossan, F. (2019). Hierarchical demand forecasting benchmark for the distribution grid.

[17] Shlifer, E., & Wolff, R. W. (1979). Aggregation and proration in forecasting. *Management Science*, *25*, 594–603.

[18] Timmermann, A. (2006). Forecast combinations. *Handbook of economic forecasting*, *1*, 135–196.

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