Robust and automatic data cleansing method for short-term load forecasting of distribution feeders

Nathalie Huyghues-Beaufond[1], Simon Tindemans[2],Paola Falugi[1]
Mingyang Sun[1], Goran Strbac[1]

1. Department of Electrical and Electronic Engineering, Imperial College London, South Kensington Campus, London SW7 2AZ, UK
2. Department of Electrical Sustainable Energy, Delft University of Technology, Mekelweg 4, 2628 CD Delft, The Netherlands

Abstract

Distribution networks are undergoing fundamental changes at medium voltage level. To support growing planning and control decision-making, the need for large numbers of short-term load forecasts has emerged. Data-driven modelling of medium voltage feeders can be affected by (1) data quality issues, namely, large gross errors and missing observations (2) the presence of structural breaks in the data due to occasional network reconfiguration and load transfers. The present work investigates and reports on the effects of advanced data cleansing techniques on forecast accuracy. A hybrid framework to detect and remove outliers in large datasets is proposed; this automatic procedure combines the Tukey labelling rule and the binary segmentation algorithm to cleanse data more efficiently, it is fast and easy to implement. Various approaches for missing value imputation are investigated, including unconditional mean, Hot Deck via k-nearest neighbour and Kalman smoothing. A combination of the automatic detection/removal of outliers and the imputation methods mentioned above are implemented to cleanse time series of 342 medium-voltage feeders. A nested rolling- origin-validation technique is used to evaluate the feed-forward deep neural network models. The proposed data cleansing framework efficiently removes outliers from the data, and the accuracy of forecasts is improved. It is found that Hot Deck (k-NN) imputation performs best in balancing the bias-variance trade-off for short-term forecasting.

Keywords: distribution systems, outlier detection, binary segmentation, Kalman smoothing, multi-step forecasts

1. Introduction

Relatively little attention has been given to the short-term load forecasting problem of primary substations, probably because of load forecasts were not essential to secure the operation of passive distribution networks. With the increasing uptake of intermittent generations, distribution networks are becoming active since power flows can change direction in a rather volatile fashion. High shares of solar PV and wind generation are connected at all voltage levels in distribution networks, resulting in substantial uncertainty in their planning and operation routine [1]. The volatility of power flows introduces operational constraints on voltage control, system fault levels, thermal constraints, systems losses and high reverse power flows [2]. Greater observability of the networks is required to maintain a safe overall system and to maximise the utilisation of existing assets. Distribution networks operators (DNOs) are compelled to broaden their visibility of the networks to time horizons that include not only real-time information but also hour-ahead, day-ahead up to week-ahead forecasts. With this change in paradigm, short-term load forecast technology is becoming an essential tool that can assist distribution network operators and planners in identifying and anticipating any future critical operational conditions.

The electric power industry uses the term short-term load forecasting (STLF) to refer to the estimation of the system demand over a time horizon ranging from less than one hour to one week. Various techniques were developed starting in the mid-sixties to predict system load power grids. These techniques fell in the realm of statistical methods and rely on the assumptions of stationarity and linearity of the underlying process. Nowadays, short-term load forecasting statistical models are built upon two main modelling frameworks, the seasonal ARIMA (SARIMA) class models [3] and the exponential smoothing class models [4]. Meanwhile, short-term load forecasting based on artificial intelligence (AI) approaches took on momentum with artificial neural networks (ANN) receiving the largest share of attention due to their universal ability to learn complex nonlinear functions. With increasingly large datasets made available, forecasting models can be trained with powerful learning architectures such as forward deep neural network (FDNN), long short-term memory (LSTM) [5] and convolutional neural network (CNN) [6].

The data acquisition of large scale real-world data is prone to errors;

anomalies in data sets can lead to erroneous forecasting outcomes. Hence, data cleansing is an essential first step in data-driven learning techniques [7]. Data cleansing attempts to address data quality issues by detecting and correcting errors in data sets. In [8], data cleansing relates to the methods performed on data to enhance the quality and reliability of the data. The authors in [9] propose four quality dimensions for data: accuracy, completeness, consistency, and timeliness. Cleansing procedures seek to enhance the quality dimensions of a given data set.

Data cleaning is a labour-intensive and time-consuming task for the following reasons: 1) to select a suitable cleaning method is not trivial 2) to generalise or automate a cleansing procedure is challenging, 3) there is a risk to introduce new errors in the data. The definition of data cleansing is strongly dependent on the process under analysis. Since data cleansing activities require specific domain knowledge and expertise, errors detection and correction techniques are intrinsically either manual or semi-automatic. Most data cleansing procedures incorporate domain knowledge and statistical techniques. Domain knowledge contributes to setting rules or constraints that the data must satisfy. Statistical techniques are used to screen the data, identify patterns, detect inconsistencies and outliers, and, eliminate contamination. A comprehensive data cleansing procedure defines error types, identifies and corrects the uncovered errors and, measures improvement in the data quality. Data cleaning operations encompass data exploration, data formatting, missing values imputation, eliminating duplicates, and outliers detection [10].

In [11], historical data for one year period, with 5% of missing data, were used to predict consumption on weekdays only. Missing values were handled by listwise deletion. Listwise deletion consists of removing all timestamped rows for which one or more observations are missing. In the context of time series, listwise deletion produces an irregular spaced time series which can affect the structural dependencies of the series. The authors overcame the issues as mentioned earlier by creating multiple sections of time series bounded by the missing observations. An interesting study is also proposed that sets the problem of multistep forecasting strategies rigorously by formulating the day-ahead load forecast of commercial buildings with both recursive and direct strategies.

In [12] the authors were explicitly concerned with the impact of missing data estimation on the accuracy of solar irradiance short-term forecast, identifying the imputation methods that generate the best estimates of solar

irradiance missing values. The study has identified interpolation, weighted moving average, and Kalman filtering as the most suitable imputation strategies for solar irradiance dataset. The authors did not consider structural breaks in the investigation but suggested the topic as a direction for future research. In [13], Rahman *et al.* address the problem of training medium to long-term residential and commercial building electricity consumption forecasting models in the presence of small and large gaps (segments) of missing observations in the hourly training dataset. Small gaps are imputed using linear interpolation, while segment imputation is performed using a scheme based on LSTM models. The algorithm identifies the segment of missing values then estimates the missing observations as the weighted average of predictions produced by training two LSTM models: one with the data before the segment and the other with data after the segment.

One of the few studies investigating the impact of outliers and level-shifts on one day ahead forecast of system load can be found in [14]. The authors proposed a robust filtering algorithm based on the Kalman filter, which allows outliers to be filtered and replaced with estimates generated by the filter. The robustification of the filter is achieved by using the one-sided Hampel function, which filters only large negative residuals identified as the most dangerous contamination for the predictive model.

Akouemo *et al.* proposed in [15] two data cleansing procedures tested on natural gas consumption series. Their implementations are based on autoregressive with exogenous terms (ARX) models and ANN models. This approach is further discussed in Section 5.2.

In [16], the treatment of bad real-time load readings is raised. Wrong measurements are said to be caused by thunderstorms or communication transmission outages. These outliers are detected and corrected based on specified upper/lower limits defined by offset tolerances for the typical load profiles. Chen *et al.* [17] describe an investigation of forecast improvement of high voltage substation load where data quality enhancement is at the centre of the study. The article reports up to 20% of bad data and inaccurate measurements in the substation load historical data. Two outlier detection strategies are used. The first outlier detection method uses thresholds built upon Chebyshevs inequality, while the second compares typical daily and weekly patterns extracted by Fourier Transform from the partially cleaned data and compare the typical load curve to the raw data. Removed outliers are imputed using a linear transformation of the typical daily pattern. Forecasts are produced for raw and preprocessed testing datasets, and accuracy

are reported for both data. It was applied to a scenario with a limited test set and in the absence of network configuration events.

In [18], Ding *et al.* focus on providing a steps-by-steps model design procedure to proficiently train and test ANN-based STLF for medium and low voltage distribution feeders. The problem of missing values is handled by replacing missing observations with data from a similar day. In this study, 24-h ahead forecasts for two MV distribution feeders are produced using a recursive forecasting strategy. Forecast accuracy is evaluated with the mean average percentage error (MAPE) metric and reported as 15.5% and 10.3%. Outlier detection, structural breaks or level-shifts were not considered.

Today, short-term load forecasters start to be deployed at large scale, and hundreds of primary substation load time series data require to be modelled and forecast. In [2], Huyghues-Beaufond *et al.* provide an example of a real-world solution where a large number of medium voltage (MV) distribution feeders forecasts are used for look-ahead contingency analysis studies. Real-world time series modelling is known to be a challenging task and MV distribution feeders time series are no exception. First, there are practical challenges associated with the manipulation of time series data (i.e. timestamps format issues, duplicate data points, timezone and daylight saving issues, diverging sampling, etc.). In addition, primary substation load profiles are mixtures of industrial, commercial and residential customers [19]. Feeder data also have typical time series characteristics, such as a slow trend due to load growth over the years, several seasonal effects, annual cycle and pronounced dips around holidays periods [20]. Beside intricate seasonal patterns, the data structure might change over time due to load transfer requirements or network reconfiguration operations [21]. Network reconfiguration is bound to happen from time to time since it is essential for 1) securing the operation of the network and 2) ensuring reliable energy delivery to the end consumer. Structural breaks in feeder time series affect the level of the data and occasion level-shifts; these features are present in historical data, and they will arise in future data. Thus, level-shifts have a double contribution to decreasing STLF accuracy. Their presence in the training and testing datasets affects, respectively, the estimation of the model parameters and impact on the accuracy of forecasts [14]. Another difficulty arises when one or multiple level-shifts occur in feeder series during the data preprocessing stage, particularly during the cleaning process, during which outliers are detected and removed from raw data, and missing observations are estimated and imputed. Level-shifts affect outlier detection effectiveness if the detection

procedure does not explicitly account for them..

This paper describes a detailed study of data cleansing and short-term forecasting for a large-scale MV distribution feeder dataset. To the best of authors' knowledge, the proposed study has not been conducted before and does not currently exist in the literature. The paper also introduces a robust, automatic and computationally efficient data cleansing approach for STLF of distribution feeders and performs and extensive analysis. The main contributions are:

- We provide an in-depth discussion of the challenges associated with producing accurate short-term load forecasts at the medium voltage distribution level using real-world data.
- All studies are performed on a real-world data set comprising 342 MV feeders, with measurements spanning two years and one half. The analysis simultaneously addresses outlier detection, missing value imputation, level-shift detection and short-term forecasting.
- An unsupervised outlier detection framework for multi-seasonal univariate time series data is proposed. It combines binary segmentation and an adapted version of the boxplot labelling rule to detect outliers in the presence of unknown numbers of structural breaks in the data. It is robust and has low computational complexity. On average, it is 65 times faster than standard hypothesis test. The description of the method also includes a general formulation for the stopping criterion of the binary segmentation algorithm that is suitable for fitting L1-norm models.
- The performance of three missing data imputation techniques (Unconditional mean, Kalman smoothing and Hot Deck) is compared in combination with the outlier detection framework.
- An adaptation of Nested Cross-Validation, named Nested Rolling-Origin Validation (NROV), to time series data is proposed to tune the parameters and evaluate the models' generalisation performance. A comparison has been performed against the Nested Adjusted k-fold validation (NAkfoldV).
- The accuracy of short-term load forecasts is quantified and compared across the full ensemble of MV distribution feeders. Consistency of

performance across different feeders and its dependence on outlier and imputation methods are analysed.

The rest of this paper is organised as per the flow chart provided in Fig. 1: Sec.2, we discuss the proposed outlier detection framework and its main components. In Sec. 3, missing data imputation techniques used for the experiment are introduced. Sec.4 discusses the models' selection and evaluation methodology, as well as the strategy used to generate multistep ahead forecasts. Sec.5 presents the case study where the forecasts results are discussed. We compare 24-steps-ahead forecasts performances for eight combinations of the data cleansing strategies. Finally, in Sec.6 and 7, we offer discussions and suggestions for extensions and conclusions

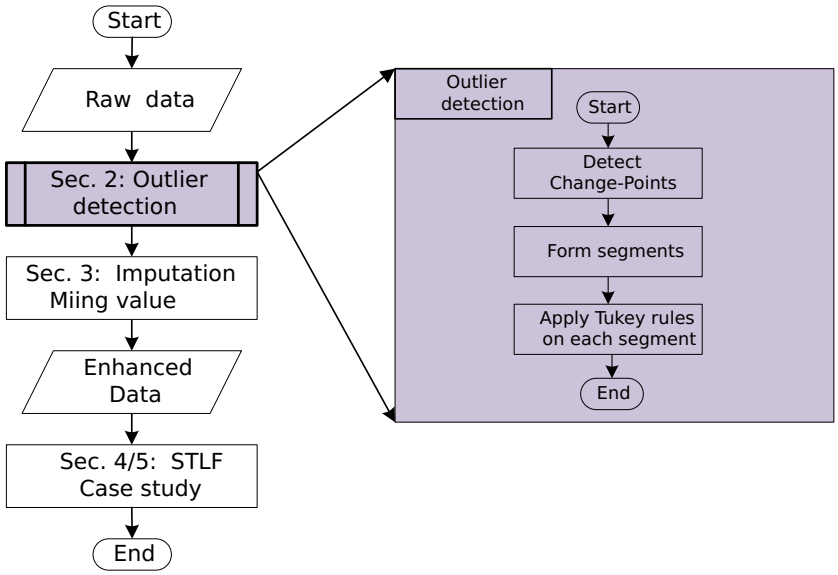

Figure 1: Flow chart of paper organisation

2. Outliers Detection in Univariate Time Series

Real-world data are frequently noisy and corrupted with outliers. Outliers relate to gross measurement errors, blunders and measurement errors [22]. Depending on the context, the proportion of gross errors in data is between 0.1% to 10% [23]. In [22], outliers are defined as data points that are significantly inconsistent with the remaining data. Many authors have reported

outliers in empirical data to bias parameters estimation, leading to model misspecification and reduced forecast accuracy. In the following sections, we then introduce the two components of the proposed outlier detection technique, namely Tukey's boxplot labelling rule and a robust implementation of the binary segmentation algorithm.

2.1. Robust detection of outliers in univariate time series

There are two standard approaches for dealing with outliers in regression problems, the regression diagnostics and the robust regression. A diagnostic approach identifies and removes the outliers from the data first and then fit the model to cleaned data, whereas a robust approach fits first a model to the entire dataset and then identifies the outliers as those data points which present large residuals. Robust regression approaches are sequential, and model parameters are re-estimated once the identified outliers are removed. In a recent publication, Akouemo and Povinelli [15] adopt a robust regression approach for the treatment of outliers in daily natural gas data, based on the analysis of the extreme values of forecasting residuals. This approach can be found in the statistical literature of outliers detection in time series data [24].

Statistical techniques are parametric, relying either on prior knowledge of the data distribution or on estimating unknown parameters of an assumed family of statistical distributions. The salient downsides of parametric strategies are 1) their model dependency, 2) the estimation of the model parameters is biased by outliers, 3) they often assume stationarity of the model. Robust outlier detection procedures are classified together with nonparametric and distribution-free procedures [25]. The density-based techniques with the kNN (k nearest neighbour), the LOF (local outlier factor) and Tukey's rule are among the most popular.

Tukey's rule relates to the robust method applied to identify outliers via boxplots visually. Potential outliers are flagged on the basis of upper and lower hinges that are related to quartiles of a batch of measurements rather than distributional assumptions. The first quartile $q1$ (25% percentile) and third quartile $q3$ (75% percentile) are computed to estimate the width of the central part of the data. The interquartile range ($iqr = q3 - q1$) has a breakdown point[1] of 25% [26], indicative of high robustness against outliers.

[1]The breakdown point is the smallest percentage of outliers that can cause an estimator

In Tukey's method, an observation is classed as an outlier when its value lies outside the outer fences, defined using the parameter r such that data points below $(q1 - r \times IQR)$ or above $(q3 + r \times IQR)$ are viewed as being too far from the median. The value $r = 1.5$, referred as the main resistant rule by Tukey and its performances is discussed in [27]. It is frequently used as a value that balances false positives and false negatives. The resistant rules $r = 2$ and $r = 3$ were also proposed later for heavy tail distribution [27]. We have tried the above resistant rules, but the main resistant rule $r = 1.5$ was the most suitable rule for our application.

2.2. Robust off-line change-point detection in univariate time series

Change-points are those points in a data sequence where statistical properties such as mean, median, variance or distribution change significantly. Among multiple change-points detection techniques, the Binary Segmentation (BS) is selected for its conceptual simplicity and its low computational complexity $\mathcal{O}(n \log n)$ [28]. The BS is a forward selection algorithm introduced by Scott and Knott in [29]. Let $\mathbf{y} = \{y_{1:n}\}$ denote a sample of observations from a nonstationary random process assumed to be piecewise stationary with k change-points at $\boldsymbol{\tau} = \{\tau_1, \ldots, \tau_k\} \subset \{1, \ldots, n\}$, the sequence of change-points is ordered such that $\tau_i < \tau_j$ if, and only if $i < j$. The dummy variables $\tau_0 = 0$ and $\tau_{k+1} = n$ are implicitly available. The segmentation refers to the automatic decomposition of $\mathbf{y} = \{y_{1:n}\}$ into $k + 1$ weakly stationary segments with the ith segment $s_i := \{y_{\tau_{i-1}+1:\tau_i}\}$.

At first, the entire dataset is searched for one change-point, typically via a cost function to be minimised. Once a change-point is found, the data are split into two sub-segments, defined by the identified change-point. Then, a similar search is performed on either sub-segment, possibly resulting in further splits. The recursion continues until a given criterion is satisfied. Here, to identify multiple unknown change-points in the data, the method adopts a general form where a contrast function $V(\boldsymbol{\tau}, \mathbf{y})$ (that penalizes a high number of change-points in order to avoid overfitting) is minimized with respect to k and $\boldsymbol{\tau} = \{\tau_1, \ldots, \tau_k\}$. As discussed in [30], we assume the penalty term to be linear in $k = |\boldsymbol{\tau}|$. Under this assumption, the cardinality

to take arbitrary large aberrant values.

constrained problem to be solved can be written as

$$\min_{\boldsymbol{\tau},|\boldsymbol{\tau}|} V(\boldsymbol{\tau},\mathbf{y}) + \beta|\tau| \text{ with } \beta > 0 \tag{1}$$

$$V(\boldsymbol{\tau},\mathbf{y}) := \sum_{i=0}^{k} c(y_{\tau_{i-1}+1:\tau_i}) \tag{2}$$

$$c(y_{\tau_{i-1}+1:\tau_i}) := \sum_{t=\tau_{i-1}+1}^{\tau_i} |y_t - \overline{y}_{\tau_{i-1}+1:\tau_i}| \tag{3}$$

where $c(y_{\tau_{i-1}+1:\tau_i})$ is the sum of absolute deviations for each t from the empirical median $\overline{y}_{\tau_{i-1}+1:\tau_i}$ of sub-signal $y_{\tau_{i-1}+1:\tau_i}$. The parameter β controls the balance between model complexity and goodness of fit. Low values of β favour overfitting with too many change-points and high values of β discard most true change-points. The cost function $c(.)$ in Eq. (3) measures the homogeneity of the sub-signal $s_i = \{y_{\tau_{i-1}+1:\tau_i}\}$. Thus, the cost is expected to be low when the sub-signal does not contain any change-points and large when it does.

Various cost functions are found in the existing literature among which piecewise linear models.

We use the Least Absolute Deviation (LAD) that was proposed by Bai in [31] for the estimation of level-shift points in autoregressive signals and noisy distributions. He considered the L1-norm because of its robustness against fat tails distribution. [31]

The Binary Segmentation approach iteratively inserts change-points in segments $s_i = \{y_{\tau_{i-1}+1:\tau}\}$ of the entire signal $\mathbf{y} = \{y_{1:n}\}$. The elementary operation is the single change-point method, it tests if a split of the segment exists such that the cost function over the two sub-segments plus the penalty term is smaller than the cost function across the entire signal $\mathbf{y} = \{y_{1:n}\}$. Under the linear assumption, the penalty term $\beta|\tau|$ is reduced to β for a single change-point search and the algorithm tests whether it exists a time index $\tau \in \{1 \dots n\}$ that satisfies

$$c(y_{1:\tau}) + c(y_{\tau+1:n}) + \beta < c(y_{1:n}) \tag{4}$$

If no change-point is detected, no additional change point is created and the algorithm stops. In the literature, penalty terms have been proposed and justified either from theoretical assumptions or inferred from data [30, 32].

In our application, we applied the two most common penalties used in the literature, namely the Akaike and Schwartz penalties, but in combination with the L1-norm cost function, the binary segmentation failed to detect any change-points. After experimentation, the value $\beta = [4 \times \log(n)]$ was selected as a suitable choice that allows the binary segmentation algorithm to approximate the number of change-points for a wide range of feeder data. We carried a sensitivity analysis on the parameter β to assert that its value was suitable for feeder data with and without level-shifts. In addition, the proposed choice of β has been successfully tested for invariance properties against scaling and shifting of the data.

2.3. Proposed Outlier Detection Procedure

In [22], Aggarwal asserts that the most effective methods for outlier detection are dataset specific and make use of contextual information to develop strategies tailored to the data in hands. The proposed strategy integrates the seasonal load features, namely, typical days of the week and the yearly cycle, in the outlier detection procedure. It is a single-step automatic procedure which identifies all outliers in a "segment" at once as opposed to the recent recursive method proposed in [33], which requires many model fits.

The method proposes to adapt Tukey's univariate rule method to detect and remove outliers from piecewise stationary segments. Time indexes of detected change-points bound segments if any exists; otherwise the full dataset is processed. A segment must contain at least a full day to be processed.

The method compares observations to suitable upper bound and lower bounds at each time step. Let S be a segment of raw data to be processed. S is divided into $\mathcal{S}_p^{w,s}$, where p is the hour of day, p depends on the granularity of the data, i.e. $p = 24$ for hourly data, $p = 48$ for half-hourly data, w is the typical day, weekday (WD) or weekend (WE) and s is the season. We implement Tukey's method to construct one Upper Bound (UB) vector and one Lower Bound (LB) vector, one for each typical days at all seasons s in S. Let $[\boldsymbol{L}_{\boldsymbol{b}_s}^{(1)}, \boldsymbol{U}_{\boldsymbol{b}_s}^{(1)}] \in \mathbb{R}^{p\times 2}$, the UB_s and LB_s be vectors for WD and $[\boldsymbol{L}_{\boldsymbol{b}_s}^{(2)}, \boldsymbol{U}_{\boldsymbol{b}_s}^{(2)}] \in \mathbb{R}^{p\times 2}$ the UB_s and LB_s vectors for WE. We compute the 5th and the 95th percentiles $q5_s^{(i)}[j]$ and $q95_s^{(i)}[j]$ respectively, and $iqr_s^{(i)}[j]$ with $i \in [1, 2]$, $j \in [1, \ldots, p]$, then we update the UB_s and LB_s vectors for both typical days as follow:

$$\boldsymbol{L}_{\boldsymbol{b}_s}^{(i)}[j] = q5_s^{(i)}[j] - 1.5 \times iqr_s^{(i)}[j] \tag{5}$$

$$\boldsymbol{U}_{\boldsymbol{b}_s}^{(i)}[j] = q95_s^{(i)}[j] + 1.5 \times iqr_s^{(i)}[j] \tag{6}$$

Once Tukey's hinges are computed for WD's and WE's for each season, daily observations in S are compared against the $UB_s^{(i)}$ and $LB_s^{(i)}$ vectors and outliers are flagged then removed from the data. Data are classified as outliers/non-outliers based on whether or not they fall outside the given bounds.

The results of the segmentation and cleaning process are illustrated in Fig.2. The top figure exhibits the raw data from *Feed 3* (see Table 5) prior to application of the outlier cleansing framework to the data. This feeder data presents multiple structural breaks and multiple outliers; piecewise stationary segments 1 to 4 are indicated with red horizontal arrows. The bottom figure illustrates the data after the outliers being removed. Table 1 reports the count and the percentage of removed outliers in each segment. The proposed cleansing procedure consists of the combination of the binary segmentation algorithm and the Tukey rule. Both procedures have a complexity of $\mathcal{O}(n \log n)$ which gives to our procedure a complexity of $\mathcal{O}(n \log n)$. Fig. 3 and Fig. 4 illustrate the performance of the outlier detection procedure for a half-hourly and hourly resolution. The boxplots show the statistics of the running time and the percentage of removed outliers relative to 342 MV feeders. In practice, however, the running time rarely exceeds one minute, which is easily doable for the size of the datasets. On average less than 1% errors were detected in the training datasets with a maximum of observations removed not exceeding 2%. For distributions close to normal, the masking (false negatives) and swamping (false positive) effects on the detection error using the Tukey rule should not exceed 0.6% as per the study carried out by Hoaglin *et. al* in [27]. The forecasting performances associated with this feeder following outlier removal and missing values imputation process are presented in Table 5 and Table 6 and discussed in the case study section.

Table 1: Number of outliers removed per segment for *Feeder 3*

	segment 1	**segment 2**	**segment 3**	**segment 4**
sequence	[0, 6150]	[6151, 9380]	[9381, 18450]	[18451,21641]
No of outliers	70	18	87	23
(%) outliers	0.32	0.08	0.4	0.1

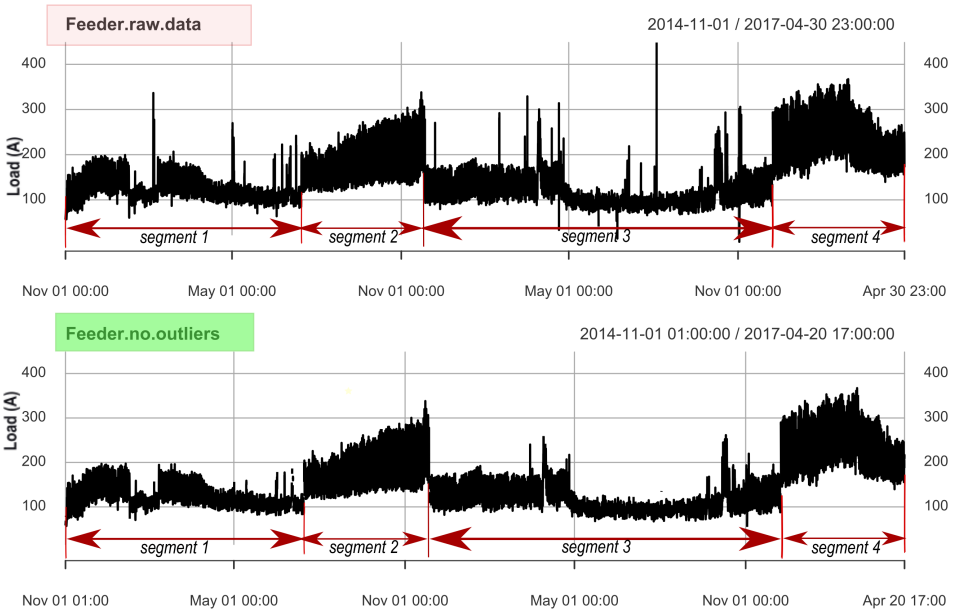

Figure 2: Outliers detection/removal with BS-Tukey for *Feeder 3* data which contain multiple structural breaks - top plot(before detection), bottom plot(after detection).

3. Missing Observations Imputation

As standing assumption for the rest of this work, we assume the missingness mechanism in feeders load data is Missing at Random (MAR) as per Little and Rubin [34] classification system. This assumption suggests that we can ignore the precise mechanism underlying the missing data, and missing data and observed data are assumed to come from the same distribution. The imputation techniques used in this paper to handle missing observations in the MV feeder series are listed below.

3.1. Simple imputation: Mean Substitution

The simplest univariate imputation technique is mean substitution. Mean substitution is a heuristic method that substitutes missing observations by the unconditional mean of the observed data. Mean imputation is naive and should be cautiously used since it can severely distort the empirical distribution of the data and insert bias in analytic or statistical inference, especially if the data is nonstationary [35, 36].

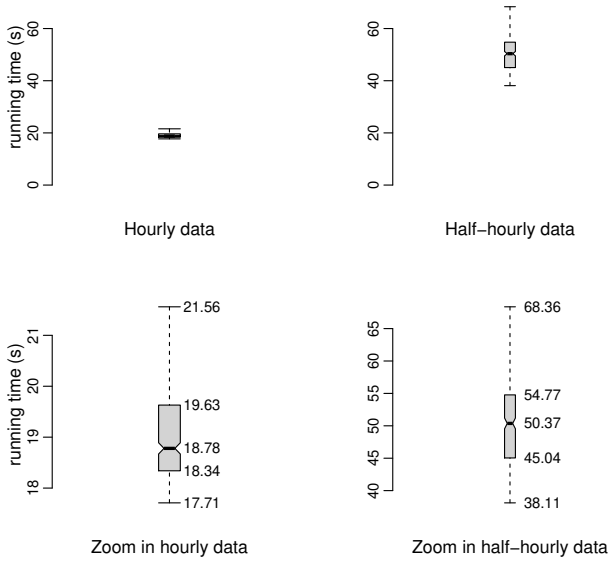

Figure 3: Running time statistics of the automatic outlier detection across 342 feeders for half-hourly data (43282 samples each) and hourly data (21641 samples each)

3.2. Hot Deck imputation: k-Nearest Neighbour

Hot Deck class of imputation techniques is widely used because it makes only minimal assumptions on the data. The procedure replaces missing values (recipient) by values extracted from responding covariates (donors) that most resembles the recipient. The algorithm widely used for matching donors to recipients is the k-Nearest Neighbour (kNN). The imputed value is either a single observation drawn from another variable (1-NN) or the weighted average of k observations drawn from k variables (k-NN) [37]. In standard kNN imputation, the similarity between recipient and donors is measured with the Euclidean or Manhattan distance [38].

In this work, the `optimistic` knn algorithm available in the python's `fancyimpute` library was used to impute the feeders' data. The imputation procedure is illustrated in Fig. 5. Feeder data sets are represented as a matrix $M \in \mathbb{R}^{n \times m}$, where each column is a time series of n regularly spaced measurement values. There are m such columns (features), one for each feeder. Missing observations are imputed on a row-by-row basis; the k-NN algorithm selects each row's k nearest neighbours (i.e. times with similar

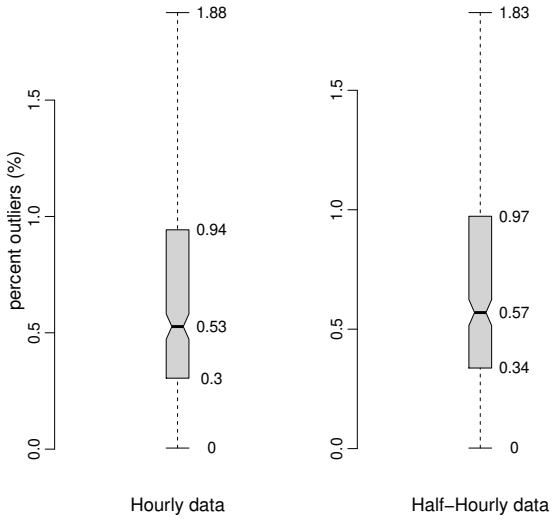

Figure 4: Statistics of the percentage outliers removed by the automatic outlier detection across 342 feeders for hourly data (43282 samples each) and hourly data (21641 samples each)

measurements) and computes their weighted average to impute the missing observations. The nearest neighbours of the ith row are identified as being the k rows with the smallest normalized Euclidean distances

$$d(i,j) = \frac{1}{n_{0(i,j)}} \sum_{h \in \Omega_{i,j}} (O_{i,h} - O_{j,h})^2 \tag{7}$$

where $O_{j,h}$ is the observed value for feeder h at timestamp j and the set $\Omega_{i,j}$ is defined as the set of common features between i and j (i.e. the feeders for which data is available at both time stamps) with $n_{0(i,j)} \doteq |\Omega_{i,j}| \leq m$. Note that $d(i,j) = d(j,i)$ holds for all $i, j \in \{0, \ldots, n-1\}$. Let us define the set $\mathcal{D}_i^k$ of k indices with the smallest distance from i as $\mathcal{D}_i^k \doteq \{j \neq i : d(i,j) \leq d_i^k,\ j = 0, \ldots, n-1\}$ with $d_i^k = \min_{d_x}\{d_x \ : \ \left(\sum_j \mathbb{I}_{d(i,j) \leq d_x}\right) \geq k\}$ in which $\mathbb{I}[.]$ denotes the indicator function. The imputed value $\hat{x}_{(i,h)}$ of feeder h at time

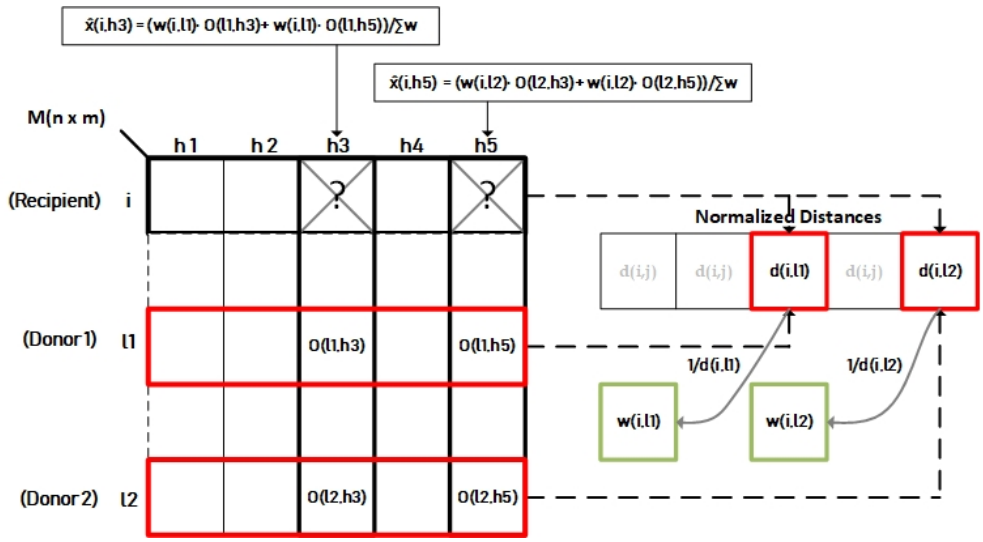

Figure 5: k-NN imputation technique illustration for $k = 2$. h_m = feeders, i, l_k = time

stamp i is given by

$$\hat{x}_{(i,h)} = \frac{\sum_{l \in \mathcal{D}_i^k} w_{i,l} O_{l,h}}{\sum_{l \in \mathcal{D}_i^k} w_{i,l}} \tag{8}$$

$$w_{i,l} = \frac{1}{d(i,l)} \tag{9}$$

The weight $w_{i,l}$ controls the influence of the observed values $O_{l,h}$ in the computation of $\hat{x}_{(i,h)}$.

3.3. Kalman Smoothing Imputation

The Kalman Filter is a recursive optimal linear filter, which is based upon the representation of a dynamic system in a state space form. Let $y_t \doteq \{y_t\}_{t=1}^n$ be a univariate nonstationary time series. It is supplemented by unobserved variables represented in a state vector $\{\boldsymbol{\alpha_t}\}$ where $\{y_t\}$ and $\{\boldsymbol{\alpha_t}\}$ are jointly Gaussian processes. We adopt a simple local linear trend model as suggested in [39, 40], taking the following form:

$$y_t = \mu_t + \varepsilon_t \tag{10}$$

$$\mu_t = \mu_{t-1} + \beta_{t-1} + \eta_t \tag{11}$$

$$\beta_t = \beta_{t-1} + \zeta_t \tag{12}$$

where $\varepsilon_t \sim N(0, \sigma_\varepsilon^2)$, $\eta_t \sim N(0, \sigma_\eta^2)$ and $\zeta_t \sim N(0, \sigma_\zeta^2)$ are white noise disturbances mutually uncorrelated . The local trend model can be cast in state space form as follows

$$\begin{aligned} y_t &= \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} \mu_t \\ \beta_t \end{bmatrix} + \varepsilon_t \\ \boldsymbol{\alpha}_t = \begin{bmatrix} \mu_t \\ \beta_t \end{bmatrix} &= \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \mu_{t-1} \\ \beta_{t-1} \end{bmatrix} + \begin{bmatrix} \eta_t \\ \zeta_t \end{bmatrix} \end{aligned} \tag{13}$$

Parameters are estimated by maximizing the Gaussian likelihood function of the chosen model. For this work, we use the Kalman smoother imputation available in `pykalman` library (version 0.9.2) with python 3.6.

The algorithm is used in four variants. The first variant consists of smoothing the entire raw univariate time series so that the filter performs two actions: it smooths the entire signal (all observed data) and interpolates the missing observations. In the second variant, the outliers are flagged and removed using the proposed outlier detection technique, and the remaining cleaned data are smoothed while all missing observations are interpolated by the smoother. In the third and fourth variants, missing values and removed outliers are replaced by interpolated values. No smoothing is performed on healthy raw data.

4. Methodology

4.1. Inputs Selection and Model Architectures

Deep neural network models were trained to forecast load for each feeder. Network architectures were optimized by grid search via a nested cross-validation procedure which is discussed in the following section. Table 2 and Table 3 provide details on the input layer configuration and network architecture and training algorithm specifications respectively. All algorithms have been implemented in Python 3.5 and 3.6. All studies were carried out on Windows 10 operating system with an Intel Xeon CPU E5-2630 2.4GHz processor. The mean absolute percentage error (MAPE) given by Eq. 14 is computed as the forecast performance metric. Model performance evaluation and the forecasting results are discussed in the following sections.

$$\text{MAPE} = \frac{1}{n} \sum_{t=1}^{n} \left| \frac{\hat{y}_t - y_t}{y_t} \right| \times 100\% \tag{14}$$

Weather variables were added to the input layer based upon the localization of the substation that hosts each feeder and the proximity of the substation to one of the eleven weather stations for which we had weather forecast data. Even though weather and feeder load scatter plots revealed a low correlation between the variables, which agrees with recent studies that report 80% of heating and cooling systems in the UK are supplied with gas energy, temperature and humidity weather forecast appeared to improve model performance during stepwise feature selection. High dimensional input space leads to an excessive number of input weights and poor performance [41]. Thus, to reduce the input space, dummy calendar variables were encoded with sinusoidal functions as shown in Table 2 rather than using one-hot encoding as in [13]. The network input vector $\boldsymbol{x}$ contains 13 inputs reported in Table 2. Each input was standardized with z_{score} such that each x_i was computed as $\hat{x}_i := (x_i - \mu_i)/\sigma_i$, where μ is the input mean and σ is its standard deviation.

Table 2: Network input layer (*)

Variables	Inputs
Load lags (current)	y_{t-1} or $\hat{y}_{t-1}$
	y_{t-24}, y_{t-48}
Weather forecast	temp(t+h), hum(t+h)
Calendar cycles	$\cos(2\pi\mathfrak{h}/24)$, $\sin(2\pi\mathfrak{h}/24)$
	$\cos(2\pi\mathfrak{d}/7)$, $\sin(2\pi\mathfrak{d}/7)$
	$\cos(2\pi\mathfrak{m}/12)$, $\sin(2\pi\mathfrak{m}/12)$
Day-type ($\mathfrak{D}_{\mathfrak{T}}$)	Weekday/Weekend = 0|1
	Holidays = 0|1

(*) Notes : t is the current time, I denotes the current and $\hat{\text{I}}$ denotes one-step-ahead forecast, $\mathfrak{h} \in H = \{0, \ldots, 23\}$ for time of day, $\mathfrak{d} \in D = \{0, \ldots, 6\}$ day of week with Monday = 0 and Sunday = 6, $\mathfrak{m} \in M = \{1, \ldots, 12\}$ month of year with January = 1 and December = 12.

Table 3: Specification of models' architecture and training algorithm

Tuning parameters method	Grid search
Inputs standardization	z-score
Hidden layers	$[1, 2, 4, 6, 8]$
Cells per layer	$[2, 4, 6, 8, 10]$
Batches	$[16, 32, 64]$
Activation	ReLU
Solver	Adam
Hyperparameters default settings	$\alpha = 0.001$, $\beta_1 = 0.9$ $\beta_2 = 0.999$,$\epsilon = 10^{-8}$
Cost function	MSE
Early stopping	True

4.2. Multistep-Ahead Forecasts Strategy

A recursive prediction strategy was used to generate 24-hour load forecasts (24 values) at midnight (12 am). Let $\mathbf{y} = \{y_{1:n}\}$ be a univariate feeder time series comprising n observations, and we aim to forecast the next 24 values of the time series. The underlying process is estimated by a model of the form m and an error term ϵ_t given by

$$y_t = m(\mathbb{y}_{t-1}, \mathbf{z}_t^{\mathcal{P}}; \boldsymbol{\theta}) + \epsilon_t \tag{15}$$

where $\epsilon_t \sim N(0, \sigma_t^2)$, $\boldsymbol{\theta}$ are the model parameters, $\mathbb{y}_t = [y_t, y_{t-23}, y_{t-47}]'$ and $\mathbf{z}_t^{\mathcal{P}}$ is the vector of the exogenous inputs (either known or forecast for time t) depending on the current hour, day, month and day type as reported in Table 2 and summarised with the parameter $\mathcal{P} := (\mathfrak{h}, \mathfrak{d}, \mathfrak{m}, \mathfrak{D}_{\mathfrak{T}})$. For simplicity, from hereon, we drop the dependence on θ in $m(\mathbb{y}_{t-1}, \mathbf{z}_t^{\mathcal{P}}; \boldsymbol{\theta})$ and use the shorthand notation $m(\mathbb{y}_{t-1}, \mathbf{z}_t^{\mathcal{P}})$. The recursive prediction consists of repeating one-step-ahead prediction several times using the previous forecast as input [11]. We compute forecasts recursively for $h = 0, \ldots, 23$ as

$$\hat{m}^{(h)}(y_{t-1:t-47}, \mathbf{z}_{t:t+h}^{\mathcal{P}}) = m([\hat{m}^{(h-1)}(y_{t-1:t-47}, \mathbf{z}_{t:t+h-1}^{\mathcal{P}}), y_{t+h-24}, y_{t+h-47}], \mathbf{z}_{t+h}^{\mathcal{P}}) \tag{16}$$

where the recursion is initialized by $\hat{m}^{(-1)}(y_{t-1:t-47}, \mathbf{z}_{t-1}^{\mathcal{P}}) := y_{t-1}$, and we use the conventions $y_{t-1:t-47} := [y_{t-1}, \ldots, y_{t-47}]$ and $\mathbf{z}_{t:t+h}^{\mathcal{P}} := [\mathbf{z}_t^{\mathcal{P}}, \ldots, \mathbf{z}_{t+h}^{\mathcal{P}}]$. We preprocessed training and testing datasets either by imputing missing values only or by detecting and removing outliers first followed by the imputation of all missing observations.

4.3. Datasets Preprocessing Strategy

We have preprocessed 342 MV feeders series with eight preprocessing strategies producing eight training/testing datasets. Each of the datasets was modeled with a FDNN architecture. Hence, we trained eight models per feeder. The first set of four training/testing datasets **r_m**, **r_knn**, **r_kf_imp**, **r_kf_smo** was obtained following the imputation of the raw data (**r**) without detecting and removing outliers; the missing values were imputed using either unconditional mean (**m**), 10-nearest neighbour (**knn**), Kalman imputation (**k_imp**), Kalman smoothing (**k_smo**). The other set of four training/testing datasets **no_m**, **no_knn**, **no_kf_imp**, **no_kf_smo** was processed by performing a detection/removal outliers procedure (**no**) to the raw data before imputing all missing values using the aforementioned imputation strategies.

4.4. Model Selection and Performance Evaluation

Common cross-validation techniques found in the STLF literature are out-of-sample (OOS) evaluation and k-fold cross-validation. In general, cross-validation techniques result in good model selection performances. However, particular care should be taken when the aim is to estimate the generalisation error of a model [42]. The out-of-sample method is a simple cross-validation technique that suffers from issues of high variance which can lead to overfitting in model selection due to *information leak* [43]. Hence, a resampling method, such as k-fold cross-validation, is more suitable. However, k-fold cross-validation implementation is not straightforward when it comes to time series forecasting. Because of the temporal dependence between errors in the training and testing datasets of time series data, training and testing sets are not independent, which invalidates the cross-validation results [44]. In addition, the traditional setting of k-fold cross validation uses future observations to predict the past.

To overcome the shortcomings of the standard k-fold cross-validation, two procedures are proposed to tune the parameters and evaluate the models' generalisation performance: the Nested Rolling-Origin-Validation (NROV) and the Nested Adjusted k-fold validation (NAkfoldV) illustrated in Fig. 6. Each procedure implements a pair of nested loops which offers an unbiased and robust model performance evaluation technique. Model selection and model fitting procedures are integral parts of the entire model evaluation process.

The Nested Rolling-Origin-Validation makes uses of the basic rolling-origin evaluation discussed in [45], also known as anchored walk-forward evaluation in financial optimisation. Feeder time series are partitioned multiple times in training, validation and testing sets. Each time, the training period is moved further ahead with its origin fixed at the beginning of the series. The advantages of NROV are 1) several out-of-sample errors referred to as *forecast origin* in [45] are obtained which gives a better understanding of how the models perform, 2) the strategy mimics the production scenario where forecasting models are retrained on new coming historical data, and 3) the procedure returns multiple optimum model architectures.

After each training period p with $p \in [1, \ldots, m]$, an optimum architecture is selected out of k pre-selected architectures. The selection is made based on the best one-step-ahead forecast performance achieved on the validation data of each period. The optimum architecture is retrained on the training and validation sets, and forecasts are produced on the test data (out-of-sample). At the end of the NROV, there are p optimum architectures available; in Fig. 6, *Model selection 1* compares the mean percentage error (MPE) achieved by each optimum architecture on the out-of-sample data and picks the best model that achieved the best performance.

Arguably, *model selection 1* carries the risk of biasing the model selection because different test sets are used to quantify the performance of the various optimal architectures. As a comparison, an alternative *Model selection 2* approach, NAkfoldV, is proposed, which applies an adapted k-fold strategy to split the training data into twelve equal splits. The procedure reserves the two last splits for validation and test purposes which are identical for each of the 10 periods. The best architecture is selected in the same way as for as the one used in *Model selection 1*.

The statistics for the performance of 24-steps ahead forecast across 342 MV feeders using NROV and NAkfoldV are shown in Fig. 7. The pre-processed data used for these model evaluation studies are the fully cleaned 10-nearest neighbours dataset (NO_KNN). The models' performance displays a comparable distribution of errors; thus, NROV and NAkfoldV procedures display similar model evaluation performance, and both methods offer a robust measure of forecast uncertainty through narrow confidence bands and a performance distribution close to normal. In the remainder, we use the NROV, because the procedure does not depend on a single choice of testing data (i.e. the most recent time window). Once the best models have been identified, they are retrained on the entire training data and forecasts are

produced on the new sets of test data. The data are introduced in the case study section.

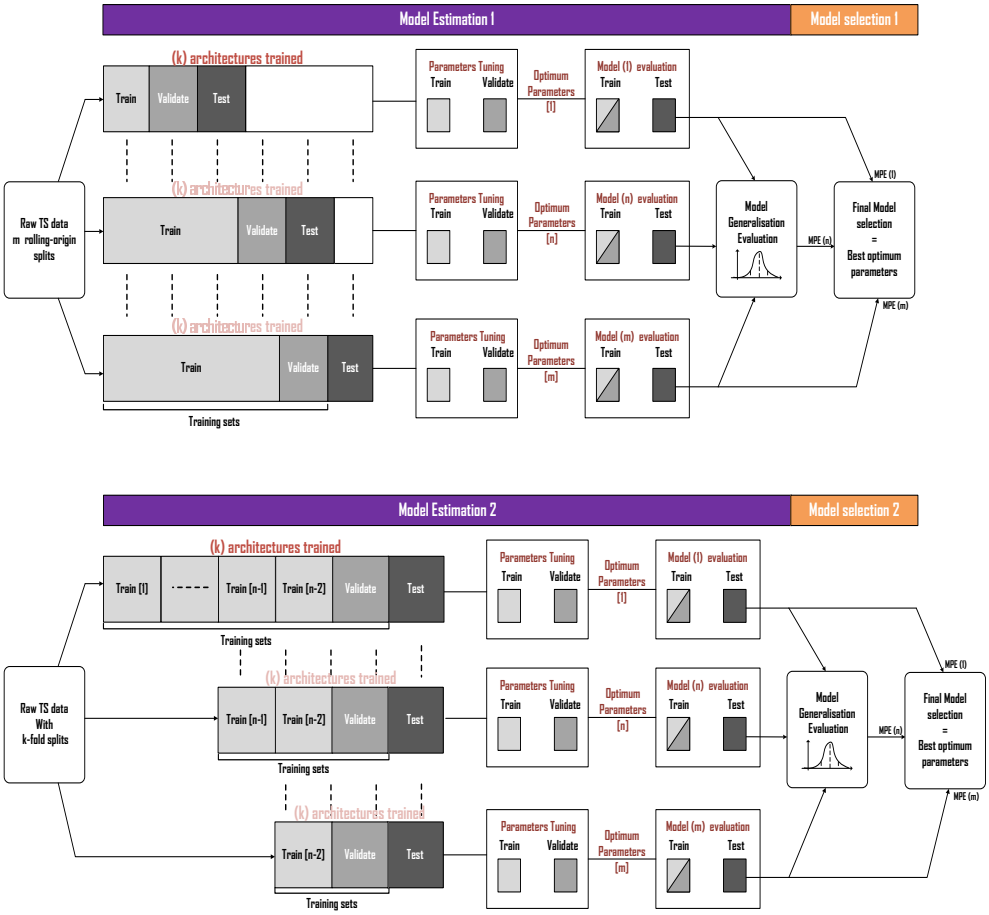

Figure 6: Nested Validation procedures - top: Nested Rolling-Origin Validation (NROV) - bottom: Nested Adjusted-k-fold Validation (NAk-foldV)

5. Case Study

5.1. The Data

UK Power Networks (UKPN), one of the six distribution network operators (DNO) in the United Kingdom have provided us with historical time series data for 342 MV/LV feeders alongside with next-day historical weather prediction data collected from eleven weather stations spread across the East

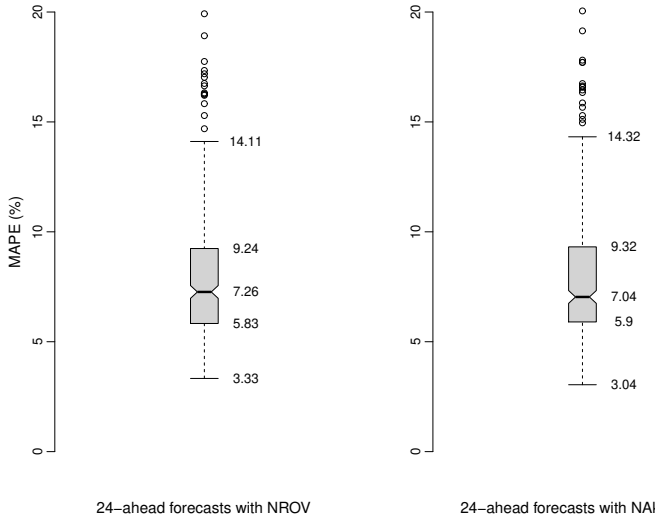

Figure 7: Statistics for the performance of 24-steps ahead forecast across 342 MV feeders using 1) Nested Rolling-Origin Validation (NROV) on the left and 2) Nested Adjusted-k-fold Validation (NAk-foldV) on the right (NO_KNN dataset)

Kent area in the South East region of England. Feeders are connected to 33kV and 11kV distribution substations. Load and weather data cover the period starting from January 2014 and ending in September 2017 at hourly granularity. Datasets are divided into training, and testing sets with the testing set starting in May 2017. The number of missing observations averages 1.08% across (training and testing data combined) with a standard deviation across feeders of 1.25%. In the following, the notation 1.08±1.25% will be used to report mean and variation across feeders. After outlier removal, the fraction of missing data increases to 2.0±1.41%.

5.2. Outlier detection: Tukey Rule vs Hypothesis Testing

Here, we compare the proposed outlier detection using Turkey's method with the algorithm proposed by Akouemo and Povinelli [15], which we will refer to as H-test. The H-test procedure is recursive; it fits uncleaned data to a given model and runs a hypothesis testing procedure on the forecast residuals to establish if an extremum is an outlier. When the probability of an extremum to be an outlier exceeds the level of significance set to $\alpha = 0.01$, the corresponding data point is removed from the dataset, imputed, and the

model is retrained on healthier data. The procedure repeats until no outliers are found in the residuals.

The method proposed in [15] was adapted in a few ways to make a direct comparison possible. The authors of [15] propose to fit the data to ANN and NARX models, instead, we used the same FDNN architecture and inputs vector described in the previous section to fit the data. Moreover, we have modified the procedure to accommodate the data in hands in three ways. First, we had to impute all missing values in the raw data prior to train the FDNN models. We chose to impute the raw data with unconditional mean. Secondly, to ensure a fair comparison, we added the change-points detection procedure to the H-test algorithm. For each of the detected segments, we trained a model and detect outliers in the one-step-ahead forecast residuals. Lastly, we imputed the outliers using the median of segments instead of an interpolation method since the feeder data exhibit groups of consecutive outliers but not isolated outliers as it was the case with the natural gas data used in [15]. We run the H-test augmented with change-points detection as follows; outliers were searched one-by-one and temporarily imputed with the median of the segment. Outlier indices were recorded during the search. Once the algorithm had found all the outliers, we substituted them with mean, and we trained the FDNN forecasters so that we could compare the forecasting results with **no_m** (no outliers + mean imputation) datasets for Tukey and H-test method.

We compare both procedures in terms of running time and detection accuracy, by applying them to load time series of two feeders: *Feed 1* and *Feed 3* (see Fig. 8). We first used both Tukey's procedure and the H-test to detect outliers in the training and testing data of the two aforementioned feeders. We then trained two forecasters per feeders with the cleaned data and compared the predictions from each model.

In Table 4, we report on the number of outliers found in the training and testing datasets by both procedures. We have recorded the running time for each outlier detection procedure. Running time and 24-h ahead forecasts MAPE results are also reported in the table. Note that Tukey's method on average is 65 times faster than H-test. If we compare the MAPE results for *Feed 1* between both procedures, the performance of dataset preprocessed with Tukey and H-test are similar although H-test omitted a number of outliers in the testing data which has slightly penalized the forecast accuracy. In *Feed 3* case, H-test was significantly less successful at identifying harmful outliers. The discrepancies in the number of found outliers by the H-test

Table 4: Outlier detection and MAPE(%) results : Tukey method vs Hypothesis testing (H-test)

	Feed 1		Feed 3	
	H-test	Tukey	H-test	Tukey
number of outliers in training data	199	191	52	198
Running time (min)	50.51	0.37	8.33	0.33
number of outliers in testing data	22	39	35	76
Running time (min)	2.23	0.055	3.60	0.061
MAPE(%)				
no_m_r	5.33	4.11	10.40	8.38
no_m_no	4.11	4.02	10.34	6.22

algorithm can be explained; the binary segmentation only approximates the number of change-points. Hence, the number of level-shifts detected in the data might not always be optimal. If the segmentation is not optimal, the residual normality assumption does not hold, and the H-test cannot perform well. This demonstrates the robustness of the Tukey method. In addition, the H-test method relies on the model's parameters estimation, therefore each time the method runs, the total number of found outliers varies as opposed to Tukey method that always flagged the same observations as outliers.

To conclude, both methods are relatively easy to implement, but H-test is less suitable for large data sets, due to computational requirements. Tukey's method is fast and robust and will be a better choice for voluminous data.

5.3. Forecast Performance

Let us first recall that FDNN or any deep neural networks architectures fail to process data with missing values. Here, results are provided for 1) model performance evaluation with out-of-sample evaluation and nested cross-validation 2) 24-steps-ahead forecast at feeder level 3) 24-steps-ahead forecast results at scale.

5.3.1. ***Results at individual feeder level***

To evaluate forecasting performance, we used the mean absolute percentage error metric referred to as MAPE in Eq. (14). We have computed the MAPE using two ground truth datasets; the raw testing data (**r**) and the testing data cleansed from outliers (**no**). The latter allows us to compute a 'clean' forecast accuracy because accuracy results are reduced when the

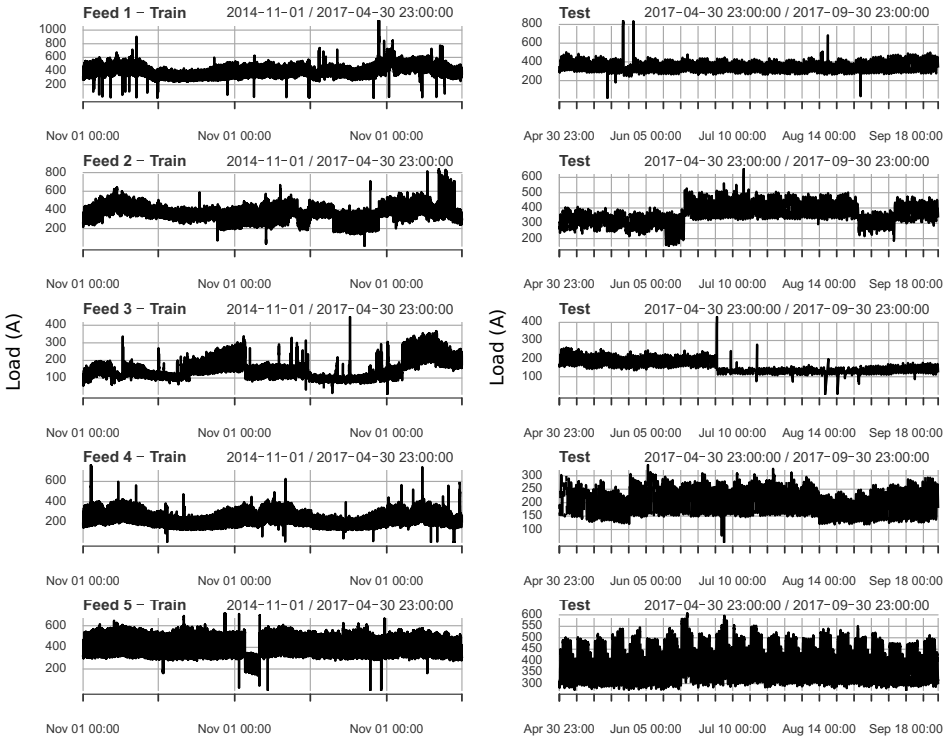

Figure 8: Time plots for the training and testing raw datasets of "Feed 1" to "Feed 5"

Table 5: 24-h ahead forecast MAPE(%) results for 5 feeders - Training/testing datasets preprocessed with imputation only (no outliers cleaning)

	Feed 1	Feed 2	Feed 3	Feed 4	Feed 5
r_m_r	5.34	**6.13**	**10.37**	5.77	**6.11**
r_m_no	4.06	**5.99**	**8.39**	5.67	**6.11**
r_knn_r	5.46	6.30	9.46	**6.19**	6.27
r_knn_no	4.17	6.17	7.25	**6.07**	6.27
r_kf_imp_r	**5.22**	6.17	9.70	5.72	6.31
r_kf_imp_no	**3.94**	6.06	7.56	5.61	6.30
r_kf_smo_r	6.17	6.44	**9.01**	**5.70**	6.14
r_kf_smo_no	4.88	6.31	**6.92**	**5.59**	6.14
missing values train (%)	0.80	1.05	1.28	1.17	0.30
missing values test (%)	0.24	0.32	0.22	0.508	0.07

Note : Training and testing data preprocessing strategies are indicated in black; ground truth data are indicated in blue (r = Raw data, no = No outliers, m = Unconditional mean imputation, kf_imp = Kalman filter imputation, kf_smo = Kalman smoother imputation, knn = 10-Nearest Neighbour)

ground truth data contain outliers. Note that training and testing data are always preprocessed with the same strategy.

The achieved accuracies for 24h-ahead forecasts of five feeders are reported in Table 5 and Table 6. In both tables, the left-hand-side column indicates in black the data preprocessing strategies and in gray which ground truth data were used to compute the MAPE (i.e. **r_kf_imp_no** must be understood as the model is trained and tested with raw data imputed with the Kalman filter and the ground truth has been cleaned from outliers). Table 5 describes the accuracy results obtained for the forecasters trained and tested on raw data with the missing values imputed. Table 6 outlines the performances achieved by the forecasters trained and tested on data fully processed (outliers removal + imputation). The results highlighted in red indicate the worst forecasting performances between Table 5 and Table 6.

Fig. 8 shows the time plots of the training and testing raw datasets for each of the five feeders under consideration. The plots help highlight the presence of gross errors and level-shifts in the datasets. From these plots we see that *Feed 1* and *Feed 4* present no structural breaks in the training data and behave alike in terms of overall shape; both data contain outliers of varying amplitudes spread across the full dataset (training and testing), but *Feed 4* has two large outliers at the end of the training data. The training

Table 6: 24-h ahead forecast MAPE(%) results for 5 feeders-Training/testing datasets preprocessed with imputation and outlier cleaning

	Feed 1	Feed 2	Feed 3	Feed 4	Feed 5
no_m_r	5.31	6.35	8.38	**4.92**	5.70
no_m_no	4.02	6.10	6.22	**4.80**	5.69
no_knn_r	5.06	**6.12**	**8.24**	4.98	**5.31**
no_knn_no	3.76	**5.98**	**6.08**	4.87	**5.30**
no_kf_imp_r	**4.89**	6.36	9.05	5.57	5.98
no_kf_imp_no	**3.59**	6.13	6.96	5.47	5.98
no_kf_smo_r	4.89	6.37	8.42	5.14	5.96
no_kf_smo_no	3.58	6.25	6.30	5.03	5.96
missing values train (%)	1.68	1.82	2.75	2.73	2.02
missing values test (%)	1.82	1.70	1.0	1.58	0.32

Note : Training and testing data preprocessing strategy is indicated in black; ground truth data are indicated in blue (r = raw data, no = no outliers, m = unconditional mean imputation, kf_imp = Kalman Filter imputation, kf_smo = Kalman smoother imputation, knn = 10-Nearest Neighbour)

and testing data of *Feed 2* and *Feed 3* contain multiple structural breaks and gross errors of various amplitude. *Feed 5* presents only one significant structural break in the middle of the training data, but there are no significant data quality issues in its testing data. At the bottom of Table 5 and Table 6, the percentage of missing values (**mv**) in the feeders' training and testing data are reported. Note that missing values can only be logged before any imputation is performed.

We now discuss the forecasts accuracy results considering the five feeders one-by-one, and we compare the feeders' MAPE reported in Table 5 (imputed raw datasets) and Table 6 (cleaned and imputed datasets). We are interested in assessing how the data cleaning procedure improves the modelling performances; ergo, we compute the forecasting accuracy improvement as the difference between the best performances found in Table 6 and Table 5. Let us start with *Feed 5*. The feeder's raw data present a modest percentage of missing observations, and consequently, most of the missing values in Table 6 are due to outliers being removed. The forecasts results obtained with the raw data imputed with **mean** outperformed the three other techniques. Following the outliers cleaning procedure, forecasts accuracy has improved for all imputation techniques, but the **knn** imputation did a better job on cleaner data and surpassed all the other imputation methods. Because the

testing data had very few outliers, the MAPE results obtained with both type of ground truth data (**r** and **no**) are almost identical. With *Feed 5*, we achieve a 0.80% reduction of the MAPE.

Now we discuss *Feed 4* and note in the first instance that overall *Feed 4* performs better than *Feed 5* despite the level of missing values in *Feed 4* data being larger than in *Feed 5* data and despite the presence of many large gross errors in its series. This implies that the only presence of the level-shift in *Feed 5* training data has biased the estimation of the model parameters. Going back to *Feed 4*, the model's performances indicate that Kalman smoothing was the best imputation procedure used on the raw data. These good performances are a consequence of the filtering of some of the noisy data in both training and testing data. Note that imputing *Feed 4* raw training data with **knn** produced the worst forecasts possibly because contaminated observations have been used to impute missing observations. Overall, *Feed 4* forecasts accuracy has improved following outliers removal and best forecasts are achieved with **mean** imputation on cleansed data. We have reported the accuracy improvement based on the MAPE obtained with the cleaned ground truth data (**no**) since *Feed 4* testing data contained many outliers. *Feed 4* forecasts were improved by 0.8.

For *Feed 3* we obtained the worst forecasts performance among all five feeders. These poor results are due to the presence of multiple level-shifts in the training data and one in the testing data. In addition, the feeder's raw data contain the highest level of missing observations exacerbated by the outlier cleansing procedure. Similar to *Feed 4*, *Feed 3*'s model performed best on smoothed raw data while **knn** imputation outperformed **mean** and **Kalman** on cleaned data. With *Feed 3*, we also achieved 0.80% accuracy improvement. Forecast results for *Feed 2* show that outlier cleansing does not always help FDNN models to perform better. The results also emphasise what we have already discussed for *Feed 5* and *Feed 3*; level-shifts in the training data affect negatively the model fit consequently the performance of the forecasters. From *Feed 2* accuracy results, we see that the forecasts produced by the models trained on cleaned data only and imputed with **knn** or **Kalman smoothing** are approximately equal.

Last but not least, we discuss *Feed 1* modelling results. The forecaster trained on *Feed 1* data achieved the best forecasts among the five feeders with the lowest MAPE = 3.59% against 6.08% for *Feed 3*. As a reminder, all accuracy results reported in this section relate to 24-h ahead forecasts. We achieve 0.33% accuracy improvement for this feeder with **Kalman filter**

imputation outperforming the other strategies. We further discuss these results in the discussion section.

Next, we propose to visualise the carry-over effect of outliers on short-term forecasts. This effect occurs because lagged values of feeder load data are used as inputs into the forecast model, thus impacting the forecast performance. Fig. 9 illustrates the carry-over effect. The red areas in the plots highlight the contaminated observations in the testing data and their carry-over effect on predictions while the green areas outline the improved forecasts produced with the testing data cleaned from contaminated data. Level-shifts not only affect the model parameters estimation, but they also alter the forecast accuracy as illustrated in Fig. 9. Although in Fig. 9 the level does not jump significantly, the 5th day experienced a drop in level which which cannot be predicted by the forecaster. To conclude, we have identified several factors which deteriorate the performance of MV feeders short-term load forecaster: level-shifts, outliers in historical and future data and the missing values imputation strategy.

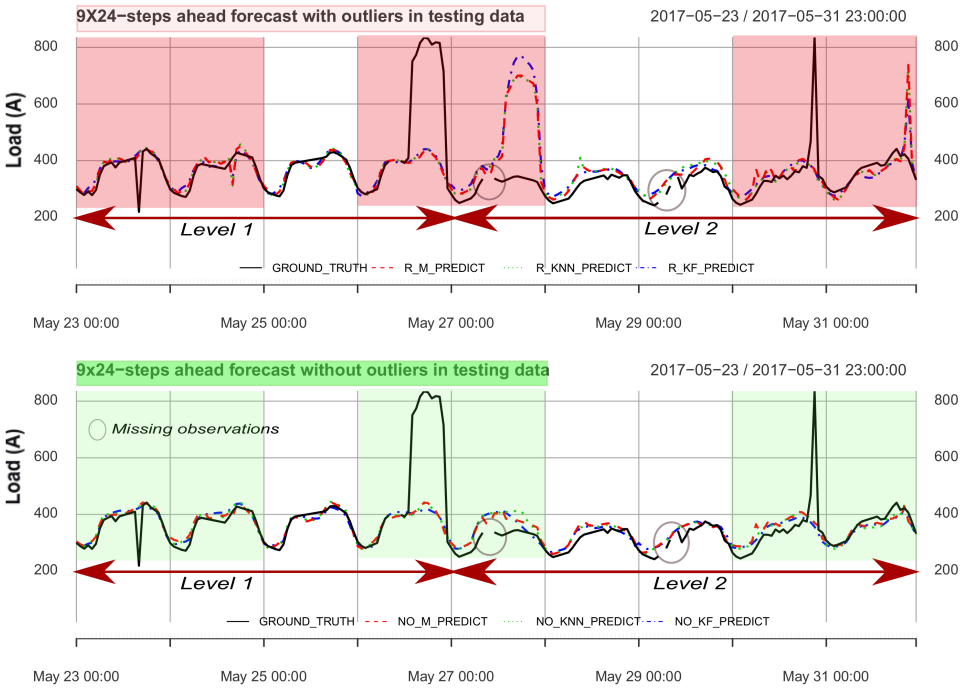

Figure 9: 9×24-steps-ahead forecasts. Testing data contain outliers (top) while outliers are removed and missing value are imputed

5.3.2. ***Forecast results at scale***

In this section, we discuss the accuracy of short-term load forecasting of 342 MV distribution feeders. The accuracy results are compiled in Tables 7, 8 and 10. Similarly to the previous section, the forecasts relate to 24-h ahead prediction of feeders loads. In Table 7, the forecasting models are identified by the strategies used to preprocess the training and testing datasets. For each model (8 models per feeder), we have computed two MAPEs: one uses raw ground truth data, the other uses cleaned ground truth data. The daily MAPEs were averaged across the full testing data so that we report a unique accuracy value per feeder for a total of 342 values per model (each model accuracy being evaluated twice). The MAPEs of each model are reported in terms of their distribution; histograms are displayed in Fig. 11.

In Table 7, we have summarised all distributions with mean and standard deviation. Because these statistics are highly sensitive to outliers, we have also computed the median and the Median Absolute Deviation (MAD) as robust alternatives to the mean and the standard deviation, respectively. The MAD is defined as

$$\text{MAD} = \text{median}|X_i - \tilde{X}| \quad \text{with } \tilde{X} = \text{median } X. \tag{17}$$

Table 7 is organised as follows: the first two columns report on the distribution of the MAPEs for the models that were trained on raw datasets (missing values are imputed) while the two last columns provide MAPEs' statistics for models trained on data fully preprocessed (outliers removal + missing values imputation). In the table, each block of four rows refers to a different imputation strategy. If we compare the mean and median of each model, they are quite dissimilar, which indicates significant skewness in the results (see Fig. 11). The rest of the analysis is consequently based on the median and the MAD. We have highlighted the robust statistics of the MAPEs in blue in the 2nd and the 4th columns of Table 7. These statistics correspond to the accuracy performance computed with cleaned ground truth data, and we refer to them in the following discussions.

Several outcomes can be drawn from the statistics in Table 7: the outlier cleaning procedure did improve the median performance of the forecasters regardless of which imputation strategy was adopted. Nonetheless, the removal of outliers tends to slightly increase the dispersion of the MAPEs (as evidenced by the MAD), which indicates an increase in the models' performance uncertainty. Overall, the imputation techniques that we have investigated perform alike on the MV feeders datasets. Surprisingly, the simple

Table 7: MAPE(%) statistics of 24-h ahead forecasts across 342 feeders

	r_m_r	**r_m_no**	**no_m_r**	**no_m_no**
mean	13.08	9.35	12.98	9.14
std	12.66	6.93	13.53	7.56
median	8.48	**7.50**	8.06	**6.99**
mad	7.89	**3.72**	8.21	**3.98**
	r_knn_r	**r_knn_no**	**no_knn_r**	**no_knn_no**
mean	13.15	9.46	12.85	9.04
std	12.57	6.84	12.99	6.85
median	8.42	**7.69**	8.04	**7.26**
mad	7.89	**3.69**	7.97	**3.73**
	r_kf_imp_r	**r_kf_imp_no**	**no_kf_imp_r**	**no_kf_imp_no**
mean	13.15	9.46	12.9	9.15
std	12.42	6.54	12.77	6.70
median	8.48	**7.63**	8.02	**7.33**
mad	7.83	**3.64**	7.84	**3.68**
	r_kf_smo_r	**r_kf_smo_no**	**no_kf_smo_r**	**no_kf_smo_no**
mean	13.13	9.44	13	9.24
std	12.49	6.67	12.71	6.61
median	8.35	**7.59**	8.09	**7.28**
mad	7.87	**3.70**	7.92	**3.71**

Note : Training/testing data preprocess strategy is in black - ground truth data are in blue with r = raw data, no = no outliers, m = unconditional mean imputation, kf_imp = Kalman Filter imputation, kf_smo = Kalman smoother imputation, knn = 10-Nearest Neighbour

unconditional mean imputation technique achieved a good score with the lowest median of the MAPEs.

To contrast between the performance of each cleansing strategy and better identify if a given data preprocessing can be more suitable, we provide in Table 8 the counts of feeders for which the computed MAPE does not exceed the following upper bounds: 5%, 7.5%, 10% and 15%. Feeders for which the MAPE exceed 15% are considered to be outliers; these represent approximately 8.80% of all feeders.

Our experimental results indicate that most of the feeders for which forecasts accuracy were notably improved had a *raw* performance not exceeding 7.5%. The imputation strategy for which outliers removal has been the most beneficial are **knn** and **mean** with 32 feeders and 18 feeders respectively that

were led under the 7.5% upper bound performance. Table 8 shows that the **mean** strategy presents more models which perform under the 5.0 % upper bound.

Table 8: Models performances - count of feeders per data preprocessing

	MAPE (%) upper bounds				
Data preprocessing	≤ 5	≤ 7.5	≤ 10.0	≤ 15.0	> 15
r_m_no	**19**	**171**	264	311	31
no_m_no	**34**	**189**	263	312	29
r_knn_no	**17**	**158**	258	312	30
no_knn_no	**26**	**190**	266	313	29
r_kf_imp_no	14	160	265	312	30
no_kf_imp_no	19	185	266	313	28
r_kf_smo_no	15	165	260	312	30
no_kf_smo_no	17	179	265	312	30

We have further investigated the 30 feeders for which the MAPE exceeded 15%. By inspecting the feeder data and forecasts, we have identified three main reasons for the lousy forecast performance. 1) The quality of the testing data is poor; hence, it may be difficult to assess the out-of-sample performance of the forecaster, and the results from the test are not reliable. 2) The model was fitted on poor quality training data, and the model was misspecified, the model has to be retrained on better quality data. 3) The features selection was inappropriate. Generally, any issues related to the model specification should be identified and solved during the cross-validation.

To investigate the performance outliers, we propose to compare the forecasts collected during the nested cross-validation phase (see illustration Fig. 6) to those obtained during the final testing phase. Throughout the cross-validation, we evaluated the performance of 10 potential model architectures for each of the preprocessed datasets. At each evaluation, we stored the forecasts generated from out-of-sample test data. In Table 10, the MAPE performance of the 30 outlying feeders differs significantly from training to testing. The table shows that only nine feeders performed with a MAPE exceeding 15% during the training; these other 21 feeders were tested on bad quality data. Table 10 also shows that two feeders performed badly during both the training and testing phase; the quality of the historical data cause

these performances. In definitive, only nine feeders require more in-depth investigation.

Table 9: Distribution of training and testing MAPE (binned) for 30 feeders with testing MAPE> 30%

	Upper bounds							
MAPE(%)	≤ 5	≤ 7.5	≤ 10	≤ 15	≤ 30	≤ 60	≤ 200	≤ 1000
Training	3	9	4	5	6	1	1	1
Testing	0	0	0	0	23	5	2	0

Table 10: MAPE's adjustment after full data cleansing

	Δmedian	ΔMAD
r_m −> no_m	0.49	0.26
r_knn −> no_knn	**0.43**	**0.04**
r_kf_imp −> no_kf_imp	0.30	0.04
r_kf_smo −> no_kf_smo	0.31	0.01

Note : In blue is the best trade-off between improving the accuracy across all feeders while maintaining the variability of MAPE to a low level

The results in Table 8 also show that automatic outliers detection has succeeded in improving the forecasts whenever it was possible without deteriorating the overall accuracy. The main drawback with removing outliers is the increased level of missing observations to be estimated. Thus, there exists a bias-variance tradeoff associated with the choice between cleaning contaminated observations and creating additional missing values to be estimated. We have summarised the outcomes in Table 10 where we show the variations in MAPE distributions between models trained and tested on raw data and models trained and tested on cleaned data. Although **mean** imputation exhibits the best performance improvement with 0.49% MAPE improvement, the statistics show that it is at the cost of an increase in the variability of the models' performance. Hence, **mean** displays the highest MAD increase even if the increased model variability remains low. A comparison of MAPE distributions is given in 11.

In conclusion, the **knn** algorithm enhanced with the proposed cleaning strategy achieves the best compromise between improving the forecasters' performance and keeping the uncertainty of the model as low as possible.

5.3.3. ***Residuals analysis***

A careful analysis of residuals helps to assess adequately the risks associated with the use of forecasts on decision making or control strategy. We propose to investigate the average performances of the 24-steps-ahead forecasts across the full forecasting horizons. We have clustered the hourly forecast residuals, and hourly percentage error results in 48 groups using all the forecasts. The top of Fig. 10 displays the distribution of the hourly MAPE using boxplots while we only plotted the median of hourly residuals of the forecasts at the bottom of the figure.

Because feeder loads vary in magnitude, we have normalised the residuals using the feeders' load median as base-load. We have used the datasets cleaned and imputed with **knn** for the analysis. The forecast horizons for which forecasts are the most uncertain are highlighted in yellow (top panel); these hours correspond to the morning peak-hours of the day. The corresponding forecast residuals (bottom panel) show that most of the time, the models underestimate the load. These results corroborate the analysis found in [14] where the authors stipulate that robust models penalise the prediction of peak demand by down-weighting their estimation.

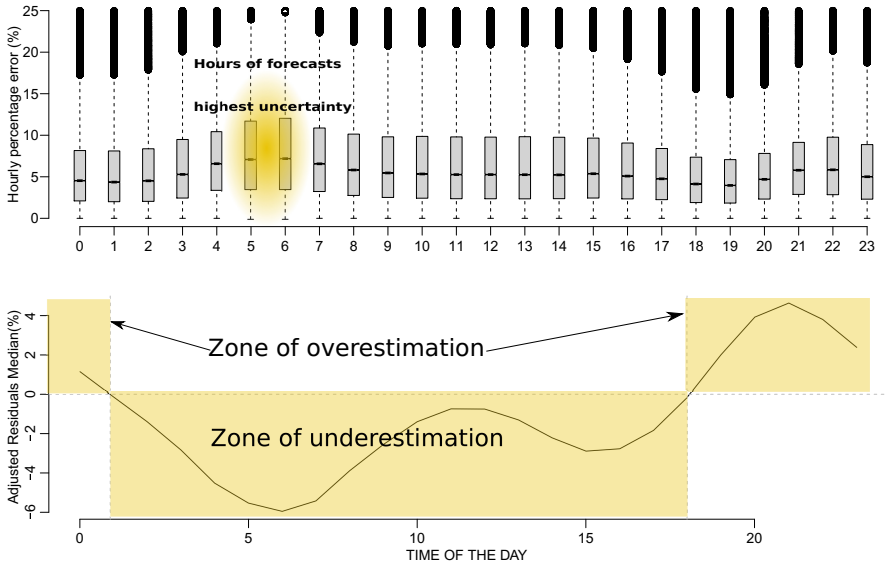

Figure 10: Hourly analysis of 24-steps-ahead forecasts across 342 MV/LV feeders for the NO_KNN training dataset. Hourly boxplots of the MAPE (top) and median normalised hourly forecast residuals (bottom)

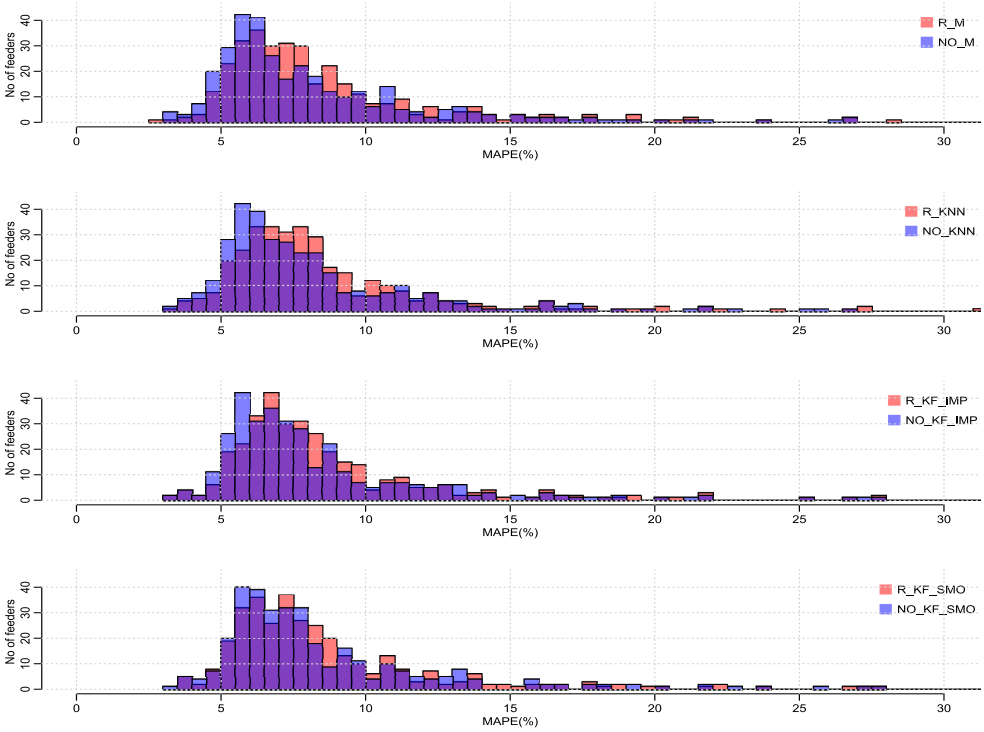

Figure 11: Histograms of the MAPEs distributions for raw imputed data and fully cleansed data

5.3.4. ***General outcomes***

The key outcomes drawn from our experiments and results are 1) there is no perfect missing values imputation technique that works well for any datasets since we found that mean imputation outperforms the Kalman smoothing method despite the latter being highly recommended in the literature given its optimality properties. 2) The **knn** is an effective imputation technique that generally performed well on our dataset but, to take full advantage of the method, the data must be cleaned from contaminated observations. 3) Structural changes lead to biased parameter estimates and forecasts. However, the forecasters handle level-shifts well by adapting quickly to changes. 4) The past 24-h lagged values are significantly weighted in the short-term forecasts problem, which creates the need for an on-line data cleaning procedure.

6. Discussion

We chose to implement the FDNN forecasters with the `Scikit-learn` Python library because it is computationally very efficient. `Scikit-learn` offers a class of deep learning models for regression problem that are fast to learn; however, the only loss function available to train the models is the mean squared error (MSE). `TensorFlow` is a powerful and flexible deep learning library. It offers a wide range of preprogrammed loss functions and users also have the option to implement customized functions. However, we have found `TensorFlow` to be computationally less efficient than `Scikit-learn` on this large scale problem [46]. Since we used the MSE loss function to train the models, further improvement in forecast accuracy might be achieved by using more robust loss functions such as the Mean Absolute Error (MAE), the Huber loss or Log hyperbolic loss.

In our work, we have computed forecasts recursively. The recursive formulation is computationally more efficient than the direct method, which requires to train one model per forecasting horizon h. The advantage of using a recursive strategy, therefore, grows when a large number of time series and multiple forecast horizons are involved. If the model's parameters $\boldsymbol{\theta}$ are chosen adequately during the cross-validation phase, the one-step ahead prediction is unbiased. However, in [47], Taieb *et al.* show that the same unbiasedness property does not hold for forecast horizons $h \geq 2$, particularly when the model is nonlinear. The authors show that bias increases with the curvature of the nonlinear function. In future works, we should consider the impact on the bias of this recursive strategy. Additionally, if computational resources are available, one should consider using a direct strategy or a hybrid recursive-direct strategy to possibly improve forecast accuracy.

The Nested Rolling-Origin-Validation (NROV) returns multiple optimum models. An improved model selection procedure would involve ensemble learning as a model averaging technique, but, here again, it would require substantial computation resources. In addition, a multivariate and univariate imputation procedures have been used independently to estimate missing observations. A hybrid imputation technique that optimally combines both procedures should be investigated for better missing value estimates. There is substantial merit in investigating how a full structural model that includes a trend, two seasonal components and cyclical component could improve the performance of Kalman imputation.

7. Conclusions

In this paper, we have presented a comprehensive investigation into the problem of short-term load forecasting for large numbers of MV distribution network feeders. Data quality and good practices in model design and evaluation are key features to achieve good forecast performances. To enhance the quality of MV feeders data, we proposed a hybrid automatic outliers cleaning procedure that is fast, robust and easy to implement. It is suitable for large datasets that present structural breaks.

We have considered three primary imputation techniques for the estimation of missing observations: Unconditional Mean, Hot Deck (k-NN) and Kalman smoothing. We combined outlier detection and missing values imputation techniques to preprocess 342 MV feeders. We modelled the MV feeders data with feed-forward deep neural networks, applying a rigorous nested cross-validation methodology that is suitable for time series data, to evaluate the performance of the models. We have adopted a recursive forecasting strategy which has the advantage of reducing the problem of 24-step-ahead forecasting to the training of a single model.

We have observed that bias and variance trade-off exists in the data quality enhancement problem. Ideally, we would like to achieve low bias and variance simultaneously, but in practice, a decrease of the bias generates an increase of the variance and vice versa. Although no imputation methods outperform the others significantly, among the three imputation techniques that we have investigated, we recommend **Hot Deck (k-NN)** because of its better performances in balancing between the bias and the variance in the recursive forecasting setting. Moreover, the k-NN technique works best on data cleaned for large gross errors. We found that robust statistics such as median and MAD should be used to report on the overall performance of the forecasters.

The results described in this paper provide essential support to analysts in charge of developing and reporting large scale short-term forecasting projects at MV distribution network level. Besides, the proposed procedure allows to quickly identify the challenges in modelling time series with data quality issues and structural changes. We have also provided guidelines on how to overcome these challenges.

Acknowledgements

The authors thanks UK Power Networks, in particular the innovation team and Alex Jakeman for their support.

References

[1] Ali Ehsan and Qiang Yang. State-of-the-art techniques for modelling of uncertainties in active distribution network planning: A review. *Applied Energy*, 239(January):1509–1523, 2019.

[2] Nathalie Huyghues-Beaufond, Alex Jakeman, S.H. Tindemans, and Goran Strbac. Enhancing distribution network visibility using contingency analysis tools. In *IET International Conference on Resilience of Transmission and Distribution Networks*, pages 4 (6 .)–4 (6 .), 2018.

[3] Gordon Lowry, Felix U. Bianeyin, and Nirav Shah. Seasonal autoregressive modelling of water and fuel consumptions in buildings. *Applied Energy*, 84(5):542–552, 2007.

[4] Alysha M. De Livera, Rob J. Hyndman, and Ralph D. Snyder. Forecasting time series with complex seasonal patterns using exponential smoothing. *Journal of the American Statistical Association*, 106(496):1513–1527, 2011.

[5] Weicong Kong, Zhao Yang Dong, Youwei Jia, David J. Hill, Yan Xu, and Yuan Zhang. Short-Term Residential Load Forecasting based on LSTM Recurrent Neural Network. *IEEE Transactions on Smart Grid*, 10(1):841–851, 2019.

[6] Heng Shi, Minghao Xu, and Ran Li. Deep Learning for Household Load Forecasting A Novel Pooling Deep RNN. *IEEE Transactions on Smart Grid*, 3053(c):1–1, 2017.

[7] Ronald J Brachman and Tej Anand. The Process of Knowledge Discovery in databases. *Advanced in knowledge discovery and data mining*, pages 37–57, 1996.

[8] Erhard Rahm and Hong Hai Do. Data Cleaning: Problems and Current Approaches Erhard. *DOLAP: Proceedings of the ACM International Workshop on Data Warehousing and OLAP*, pages 49–56, 2007.

[9] Christopher Fox, Anany Levitin, and Thomas Redman. The notion of data and its quality dimensions. *Information Processing and Management*, 30(1):9–19, 1994.

[10] Oded Maimon and Lior Rokach. *The Data Mining and Knowlegde Discovery Handbook*. Tel-Aviv, tel-aviv u edition, 2005.

[11] Mengmeng Cai, Manisa Pipattanasomporn, and Saifur Rahman. Day-ahead building-level load forecasts using deep learning vs. traditional time-series techniques. *Applied Energy*, 236(October 2018):1078–1088, 2019.

[12] Haydar Demirhan and Zoe Renwick. Missing value imputation for short to mid-term horizontal solar irradiance data. *Applied Energy*, 225(March):998–1012, 2018.

[13] Aowabin Rahman, Vivek Srikumar, and Amanda D. Smith. Predicting electricity consumption for commercial and residential buildings using deep recurrent neural networks. *Applied Energy*, 212(December 2017):372–385, 2018.

[14] J. T. Connor. Robust neural network filter for electricity demand prediction. *Journal of Forecasting*, Vol 15:6(May):437–458, 1996.

[15] Hermine N. Akouemo and Richard J. Povinelli. Data Improving in Time Series Using ARX and ANN Models. *IEEE Transactions on Power Systems*, 32(5):3352–3359, 2017.

[16] J Y Fan and J D Mcdonald. A real-time implementation of short-term load forecasting for distribution power systems. *IEEE Transactions on Power Systems*, 9(2):988–994, 1994.

[17] Xinyu Chen, Chongqing Kang, Xing Tong, Qing Xia, and Junfeng Yang. Improving the accuracy of bus load forecasting by a two-stage bad data identification method. *IEEE Transactions on Power Systems*, 29(4):1634–1641, 2014.

[18] Ni Ding, Clementine Benoit, Guillaume Foggia, Yvon Besanger, and Frederic Wurtz. Neural network-based model design for short-term load forecast in distribution systems. *IEEE Transactions on Power Systems*, 31(1):72–81, 2016.

[19] Mukwanga Willy Siti, Dan Valentin Nicolae, Adisa A. Jimoh, and Abhisek Ukil. Reconfiguration and load balancing in the LV and MV distribution networks for optimal performance. *IEEE Transactions on Power Delivery*, 22(4):2534–2540, 2007.

[20] Henrique Steinherz Hippert, Carlos Eduardo Pedreira, and Reinaldo Castro Souza. Neural networks for short-term load forecasting : A review and evaluation. *IEEE Transactions on Power Systems*, 16(1):4333, 2001.

[21] Mesut E. Baran and Felix F. Wu. Network reconfiguration in distribution systems for loss reduction and load balancing. *IEEE Transaction on Power Delivery*, Vol. 34 No:1401, 1989.

[22] Charu C. Aggarwal. *Outlier Analysis*. Springer Science, springer edition, 2013.

[23] Frank R Hampel. The Influence Curve and Its Role in Robust Estimation. *Journal of the American Statistical Association*, 69(346):383–393, 1974.

[24] Ruey S Tsay. Outliers, Level Shifts, and Variance Changes in Time Series. *Journal of forecasting*, 7(May 1987), 1988.

[25] Irad Ben-gal. Outlier Detection. *Data Mining and Knowledge Discovery Handbook*, pages 131–146, 2005.

[26] Frank R Hampel. A general qualitative definition of robustness. *The Annals of Mathematical Statistics*, 42(6):1887–1896, 1971.

[27] David C. Hoaglin, Boris Iglewicz, and John W. Tukey. Performance of some resistant rules for outlier labeling. *Journal of the American Statistical Association*, 81(396):991–999, 1986.

[28] Charles Truong, Laurent Oudre, and Nicolas Vayatis. A review of change point detection methods. 2018.

[29] A. J. Scott and M. Knott. A Cluster Analysis Method for Grouping Means in the Analysis of Variance. *Biometrics*, 30(3):507, 1974.

[30] Charles Truong, Laurent Oudrez, and Nicolas Vayatis. Penalty learning for changepoint detection. *25th European Signal Processing Conference, EUSIPCO 2017*, 2017-Janua:1569–1573, 2017.

[31] Jushan Bai. Least absolute deviation regression. *Econometric theory*, pages 403–436, 1995.

[32] Yi Ching Yao. Estimating the number of change-points via Schwarz' criterion. *Statistics and Probability Letters*, 6(3):181–189, 1988.

[33] Hermine N. Akouemo and Richard J. Povinelli. Data Improving in Time Series Using ARX and ANN Models. *IEEE Transactions on Power Systems*, 32(5):3352–3359, 2017.

[34] R. J. Little and D. B. Rubin. *Statistical Analysis with Missing Data*. John Wiley & Sons, 1987.

[35] Roderick J A Little. Missing-Data Adjustments in Large Surveys. *Journal of Business & Economic Statistics*, 6(No.3), 1988.

[36] Andrew Gelman and Jennifer Hill. Missing-data imputation. *Data Analysis Using Regression and Multilevel/Hierarchical Models*, pages 529–543, 2007.

[37] Lorenzo Beretta and Alessandro Santaniello. Nearest neighbor imputation algorithms: a critical evaluation. *BMC Medical Informatics and Decision Making*, 16(S3):74, 2016.

[38] Yenduri Sumath and Iyengar S. Performance Evaluation of Imputation Methods for Incomplete Datasets. *International Journal of Software Engineering and Knowledge Engineering*, 17(01):127–152, 2007.

[39] F Palmer. Multiple imputation of time series: an application to the construction of historical price indexes. *Biltoki*, pages 1–10, 2005.

[40] A Harvey and S Peters. Estimation procedures for structural time series models. *Journal of Forecasting*, 9(2):89–108, 1990.

[41] Alex Graves. *Supervised Sequence Labeling with Recurrent Neural Networks*, volume 12. 2013.

[42] Sylvain Arlot and Alain Celisse. A survey of cross-validation procedures for model selection. *Statistics Surveys*, 4:40–79, 2009.

[43] Gavin C. Cawley and Nicola L. Talbot. On Over-fitting in Model Selection and Subsequent Selection Bias in Performance Evaluation. *Journal of Machine Learning Research*, 11:2079–2107, 2010.

[44] Christoph Bergmeir and Jos M. Benítez. On the use of cross-validation for time series predictor evaluation. *Information Sciences*, 191:192–213, 2012.

[45] Leonard J. Tashman. Out-of-sample tests of forecasting accuracy: An analysis and review. *International Journal of Forecasting*, 16(4):437–450, 2000.

[46] Shaohuai Shi, Qiang Wang, Pengfei Xu, and Xiaowen Chu. Benchmarking state-of-the-art deep learning software tools. *Proceedings - 2016 7th International Conference on Cloud Computing and Big Data, CCBD 2016*, pages 99–104, 2017.

[47] Souhaib Ben Taieb and Amir F. Atiya. A Bias and Variance Analysis for Multistep-Ahead Time Series Forecasting. *IEEE Transactions on Neural Networks and Learning Systems*, 27(1):62–76, 2016.