Noname manuscript No.
(will be inserted by the editor)

Generation and interpretation of parsimonious predictive models for load forecasting in smart heating networks

Pre-print.

The final authenticated version is available online at
https://link.springer.com/article/10.1007/s10489-021-02949-4

Alberto Castellini · Federico Bianchi · Alessandro Farinelli

Received: date / Accepted: date

Abstract Forecasting future heat load in smart district heating networks is a key problem for utility companies that need such predictions for optimizing their operational activities. From the statistical learning viewpoint, this problem is also very interesting because it requires to integrate multiple time series about weather and social factors into a dynamical model, and to generate models able to explain the relationships between weather/social factors and heat load. Typical questions in this context are: *"Which variables are more informative for the prediction?"* and *"Do variables have different influence in different contexts (e.g., time instant or situations)?"* We propose a methodology for generating simple and interpretable models for heat load forecasting, then we apply this methodology to a real dataset, and, finally, provide new insight about this application domain. The methodology merges multi-equation multivariate linear regression and forward variable selection. We generate a (sparse) equation for each pair day-of-the-week/hour-of-the-day (for instance, one equation concerns predictions of Monday at 0.00, another predictions of Monday at 1.00, and so on). These equations are simple to explain because they locally approximate the prediction problem in specific times of day/week. Variable selection is a key contribution of this work. It provides a reduction of the prediction error of 2.4% and a decrease of the number of parameters of 49.8% compared to state-of-the-art models. Interestingly, different variables are selected in different equations (i.e., times of the day/week), showing that weather and social factors, and autoregressive variables with different delays, differently influence heat predictions in different times of the day/week.

Keywords Time series forecasting · Multivariate models · Variable selection · Smart grids · District Heating Networks

Alberto Castellini, Federico Bianchi, Alessandro Farinelli
University of Verona, Italy
E-mail: alberto.castellini@univr.it, federico.bianchi@univr.it, alessandro.farinelli@univr.it

1 Introduction

Smart grids have recently gained strong interest from the artificial intelligence community because the optimization of their functioning, productivity and maintenance requires non-trivial intelligent tools [23,14]. A specific branch of smart grids concerns *district heating networks* (DHNs) [28], in which centralized heating plants generate the heat, and a pipe system distributes it through exchangers to residential and commercial buildings. The high efficiency and pollution control of smart DHNs make these technologies strategic for future sustainable development. Moreover, the optimization of this technology can take strong advantage from the large amount of data available in smart cities [7,32].

In this work, we focus on specific artificial intelligence techniques for improving DHNs performance, namely, predictive models for heat load forecasting. They allow to predict in advance the amount of (hourly) heat a power plant should produce in the future, depending on weather conditions (e.g., temperature), social factors (e.g., the day of the week or holidays), past heat loads and possibly other sources of information. These tools can be implemented in several ways, using different statistical and machine learning models, such as, AutoRegressive Integrated Moving Average (ARIMA) models [6], Hidden Markov Models (HMM) [25], neural networks, [16] and other frameworks. We use a multi-equation linear autoregressive model, that, despite its simplicity, achieves very good performance. The main idea behind the proposed model is to split the prediction problem in many small sub-problems that can be linearly approximated with good accuracy. The main advantage of this approach is interpretability, since the mathematical equations by which each sub-problem is modeled can be interpreted by humans. New insight can be acquired by analyzing the parameters of these equations and the performance of the model.

Our original contribution concerns performance and interpretability improvement over a methodology presented in [3], by means of the introduction of a variable selection layer. This yields a reduction of the prediction error of 2.4% and a reduction of the number of parameters of 49.8% in the best model. Exploiting this reduction we provide new insight on the heat load process by an in-depth analysis of model parameters and performance, that reveal informative patterns and properties about the relationships between weather/social factors and heat load. In particular, we answer questions as, *"Which variables are more informative to predict the future heat load in specific days of the week and hours of the day?"* and *"In which days of the week and hours of the day the hourly heat load is more difficult to predict?"*, and again, *"How does a non-accurate sensor affect prediction performance?"*. Since experiments have been performed using a real dataset, answers provide insight about a true case study. The insight we provide is data-driven and it can be useful for utility companies that face similar questions to design, use or maintain their smart DHNs.

The dataset of heat load measurements has been generated by a DHN located in Verona (Italy) and managed by an Italian utility company called AGSM[1]. It contains hourly heat loads produced by three heating plants in 2014, 2016, 2017 and 2018. The overall prediction model is made of 168 linear regression equations, one for each pair day-of-the-week/hour-of-the-day (e.g., an equation is specialized

[1] `https://www.agsm.it/`

on the load prediction of Monday at 00.00, another in that of Monday at 01.00, and so on, until Sunday at 23.00). The multi-equation model performs predictions with different time horizons using all equations together, however we focus on Short Term Load Forecasting (STLF) [19,17], and aim at predicting the load of the next 48-hours. The *variable selection method* here proposed is an iterative forward stepwise technique which we apply separately to single equations, obtaining a potentially different set of variables for each equation. We analyze the performance of the overall model and of single equations trained on two datasets of different size, to show the impact of the training set size on the performance. Finally, we deepen the analysis of model parameters and get insight about the importance of different variables in different time instants.

The main contributions of this work to the state-of-the-art are summarized in the following:

- we improve the model generation pipeline proposed in [3] by adding a variable selection layer which produces a reduction of the prediction error of 2.4% and a reduction of the number of model parameters of 49.8%;
- we perform an in depth analysis of prediction performance and model parameters on a day-of-the-week/hour-of-the-day basis, identifying useful relationships between load and weather/social factors, and useful patterns for operational activities on the DHN;
- we analyze the sensitivity of model performance to the absence or inaccuracy of some weather/social factors.
- we perform a comparative analysis showing that, despite its simplicity, the proposed approach outperforms state-of-the-art methods in terms of both prediction performance and interpretability.

The rest of the manuscript is organized as follows. Section 2 presents the state-of-the-art on load forecasting and highlights the differences between methods in the literature and our approach. Section 3 describes the dataset. Section 4 defines the model generation and variable selection techniques, and the performance measures. Results are analyzed in Section 5 and conclusions are drawn in Section 6.

2 Related works

We have identified three research topics that have relationships with our work. From the more specific to the more general, we have: *i)* heat load forecasting in DHNs, *ii)* electric load forecasting and short term load forecasting, *iii)* time series forecasting. In the following we provide a short but comprehensive description of the main works of each topic, and we highlight the differences between those works and our work.

i) Heat load forecasting in DHNs. In [12,13] seasonal ARIMA models are compared to a multi-equation linear regression model based on 168-hour historical demand pattern and weather factors (i.e., temperature and wind speed). The best model identified in those works is a single-equation linear model using as predictors, the external temperature, holidays and wind speed in the last week. In tests performed on our dataset, multiple-equation models outperform single-equation

models. In [10] three different machine learning models are compared on a real-world case, based on hourly data of a DHN located in Aarhus (Denmark). The comparison shows that support vector regression based on weather factors and calendar events performs better than other models in 1538h forecasting horizons. In [15,31] eXtreme Gradient Boosting (XGBoost), linear regression and Deep Neural Networks are compared for thermal load forecasting in DHN. A methodology based on the integrated use of regularized regression and clustering for generating predictive models of future heating load in DHNs is also presented in [8]. We finally presented a first multi-equation model for heat load forecasting in [3].

The main difference between the cited literature and the present work is that here we investigate the contribution of variable selection on prediction performance and model interpretability. Moreover, in addition to standard weather variables (i.e., temperature and wind speed), we also consider the impact of relative humidity, rainfall and some engineered variables based on temperature. Furthermore, our sensitivity analyses provide practical insight about the informativeness of different factors on load prediction. urwork presented in [3] is also significantly extended by the feature selection mechanism that improves both performance and interpretability.

ii) Electric load forecasting and short term load forecasting. Electric load forecasting has several similarities with heat load forecasting because in both cases the quantity predicted is energy delivered by a central producer to consumers through a network. The consumers are also very similar in general (i.e., private and commercial buildings), hence weather and social factors that influence the prediction can be similar as well. Two key surveys on this topic are [17] and [22]. The first, dates back to the 1980s and focuses on statistical methods, while the second is more recent and considers also current machine learning models trained on high-frequency data collected by metering infrastructure. The works in this context which most resemble ours are [30,26]. In particular, we used some engineered variables related to temperature that they used in their linear regression models. However, in those paper dummy variables are used in a single equation approach to deal with factors as years, seasons, calendar events and weekdays, while we use a multiple equation approach. An hybrid technique is proposed in [20] using a least squares support vector machine for annual power load forecasting. Two very recent works about short term load forecast [11,29] propose approaches based on, respectively, deep residual networks with convolutional structure and weighted support vector regression combined with a modified grasshopper optimization algorithm to optimally select parameters. All these approaches have however interpretability limitations due to the complexity of the modeling frameworks used.

iii) Multivariate time series forecasting. First works on statistical models for time series forecasting date back to the seventies [6]. The majority of that models were univariate, namely, they predict future values of a variable given past values of the same variable. The availability of large amounts of data starting from the early 2000s fostered the development of multivariate models using information from both endogenous and exogenous variables to make predictions [24]. The advent of machine learning brought new types of model, such as, support vector machines, Gaussian processes and neural networks [1]. Feature selection and representation learning [2] has recently become also fundamental for dealing with multivariate

data. Recent reviews have evaluated machine learning and deep learning methods for time series forecasting [21], and competitions on large amount of data [24] stimulate continuous improvement of new methods.

A complete analysis of these approaches is impossible, hence we mention only a few of them from the main categories, and then we summarize the main differences with the approach proposed in the paper. XGBoost [9] is a fast implementation of gradient boosted decision trees that has recently dominated applied machine learning competitions. Time series forecasting can be tackled using this approach by transforming the problem to a supervised learning problem using sliding-window representation of the time series. Gaussian processes (GPs) are also a sound method for time series forecasting [27] which considers data uncertainty and provides also confidence intervals on predictions. Shrinkage methods for linear regression, such as, ridge [18], lasso and least angle regression have been recently used for predicting highly dimensional time series [24] achieving good performance. Finally, convolutional neural networks (CNN) and long-short term memory (LSTM) networks have gained popularity in recent times [21], since they achieve good performance on large training sets. In a previous work [4] we also tested CNNs, GPs and shrinkage methods on our dataset, achieving lower performance than in this paper, and also with a lower model interpretability.

3 Dataset

The dataset used for training our models contains hourly time series collected from four sources. The first are three plants managed by AGSM that provide hourly heat loads. The utility company provided three separated datasets for years 2014, 2016, 2017 and 2018. We merged them into a single signal representing the overall heat produced (hourly) by the plants and entered into a network located in the city center of Verona (Italy). The buildings served by the network are both residential and commercial. The second source is a temperature sensor located in Verona city center and managed by AGSM. It provides precise values of the hourly air temperature nearby the DHN. Since only the temperature signal was available from this data source we acquired also weather data from a nearby meteorological station located in the Villafranca airport, which is located about 12.5 Km out of the city center of Verona. Historical time series from this station are publicly available in the Reliable Prognosis 5 website[2]. From this website we acquired hourly time series of relative humidity, wind speed and rainfall. As a third data source we used Holidays[3], a Python module that offers functions related to calendars, legal/religious holidays and working-days. From this library we got information about Italian holidays.

The six signals described so far, namely, heat load l, air temperature T, relative humidity RH, wind speed W, rainfall R and holidays H (a binary variable generated by setting working days to 0 and holidays to 1), were engineered to obtain new variables with high predictive power in our application domain. The final list of variables, inspired from similar applications in the literature [26], is reported in Table 1. They are the heat load (used as a dependent variable in our

[2] `https://rp5.ru/`

[3] `https://pypi.org/project/holidays/`

predictive model) and nineteen variables (used as independent variables) derived from the six main signals described above.

Table 1 List of variables

Variable	Description
l	Heat load [MW] (target)
l_i	$i \in [1, 7]$, l of i days ago
l_p	l_1 at 6:00 AM (peak hour)
T	Temperature [°C]
T^2	Square of T
$T_{ma(7)}$	Moving avg of T last 7 days
T_m	Maximum T of the day
T_m^2	Square of T_M of the day
T_{m-1}	T_M of the previous day
T_{m-1}^2	T_M^2 of previous day
RH	Relative humidity [%]
W	Wind speed [m/s]
R	Rainfall(1:rain, 0:no rain)
H	Holiday(1:yes, 0:no)

The DHN has two main states of functioning. One is related to the times of the year in which the heating can be used to warm up the buildings, and the other is related to the time of the year when the heating must be switched off. This is regulated by law in Italy and has a strong impact on the prediction of future heat loads, since in the second period the load is only related to the production of hot water for sanitary use and heat dispersion in the pipeline. We make our predictions only on time periods when the heating is switched on, namely, from 01/01/2014 to 15/05/2014, from 15/10/2014 to 31/12/2014, from 01/01/2016 to 11/05/2016, from 11/10/2016 to 14/05/2017, and from 16/10/2017 to 21/04/2018. The total number of observations is 18024.

Our method, described in the next section, requires three datasets, namely, one for training model parameters (called D_P in the following), one for selecting informative variables (D_V) and the last for testing prediction performance (D_T). Since we want to evaluate the influence of dataset size on model performance, in our experiments we split the entire dataset in two different ways to generate sub-datasets D_P, D_V and D_T (see Section 5.1).

4 Method

After describing the problem and presenting an overview of the proposed framework, we introduce the methodology used to select variables and generate the predictive model. The section continues with the definition of the performance measures used to evaluate the model, and it ends presenting an open source Python software dealing with the complete data analysis and model generation pipeline.

4.1 Problem definition and system overview

The problem we address is short-term heat load forecasting in DHNs. In particular, we aim to make 48-hour predictions of the heat load produced by a plant depending on weather conditions, social factors and past values of the heat load. From a statistical and machine learning viewpoint this is a time series prediction problem with multiple exogenous and endogenous variables. The goal is to reduce as much as possible the prediction error using the information in the variables as better as possible. Figure 1 depicts the main elements of the DHN (i.e., the heating plant, buildings and the pipeline), it shows weather and social factors that we consider in our analysis and the goal of minimizing the prediction error. Our methodology aims to generate model M in Figure 1. The model is a function taking past and (predicted) future values of exogenous and endogenous variables to produce heat load prediction at the next time instant. To get 48-hour predictions, the model is run 48 times feeding back each time the current load prediction as an endogenous variable to generate the next prediction.

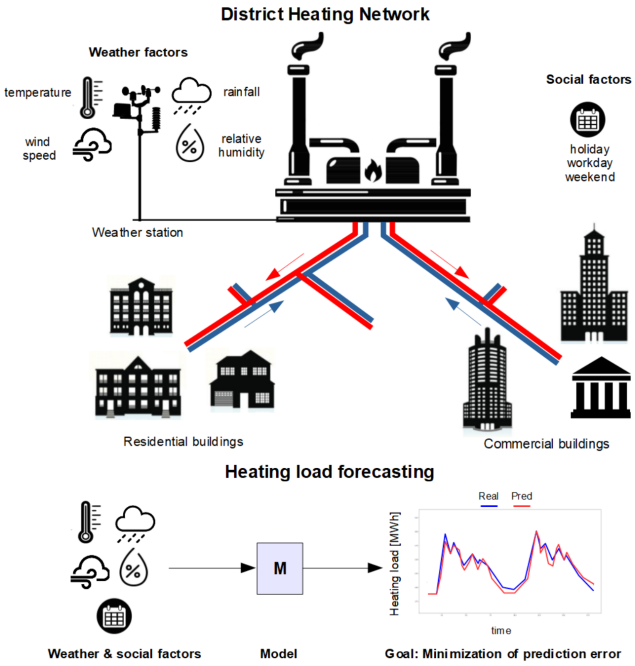

Fig. 1 Problem definition: short-term heat load forecasting in DHNs.

4.2 Model generation with integrated variable selection

The procedure which generates the predictive model is graphically depicted in Figure 2 and formally defined in Algorithm 1. The inputs are the two datasets D_P (i.e., the dataset for training model parameters) and D_V (the dataset for variable

selection). As a first step (lines 3 and 4 in Algorithm 1) we extract rows related to day i and hour j from both datasets D_P and D_V obtaining sub-datasets $D_P^{(i,j)}$ and $D_V^{(i,j)}$, $i \in \{1,\dots,7\}, j \in \{0,\dots,23\}$. Then we start to select the set of variables for each equation $E_{(i,j)}$ using the related datasets $D_P^{(i,j)}$ and $D_V^{(i,j)}$ (see line 7). The set of variables selected in each equation $E_{(i,j)}$ is independent of that of other equations because it depends only on observations in datasets $D_P^{(i,j)}$ and $D_V^{(i,j)}$ (see the *for* cycle in line 2). This has also the positive effect to make the process parallelizable.

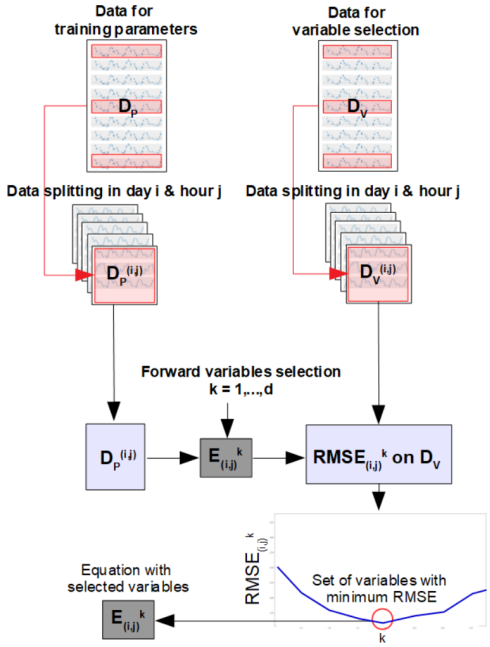

Fig. 2 Overview of the proposed methodology for variable selection and model generation.

Given a specific day i and hour j we first train, using dataset $D_P^{(i,j)}$, all possible models with a single variable selected from the set of all variables $Z = \{v_1,\dots,v_d\}$ (line 10). The cardinality of these models is equal to the total number of variables (d) in the first iteration. We then evaluate the performance of each model (line 11) by computing the average Root Mean Squared Error ($\overline{RMSE}_{1h}$) (see details in Section 4.4) between load predictions with horizon of 1-hour and the related true loads in dataset $D_V^{(i,j)}$. In Algorithm 1 this is called $\overline{RMSE}$ instead of $\overline{RMSE}_{1h}$ for simplifying the notation. The best model is subsequently selected (line 13), the related variable v^k added to the set of variables for that equation (line 16) and removed from Z (line 18). This process is then iterated (line 7). At the second iteration we generate all models with two variables, keeping v^k always in the model, then we train parameters using dataset $D_P^{(i,j)}$, and select the best model according to its RMSE on $D_V^{(i,j)}$. The iterations are performed until a model containing all d variables is obtained for equation $E_{(i,j)}$.

Algorithm 1: ModelGenWithVarSelection

Input: D_P: Dataset for training model parameters
D_V: Dataset for variable selection
Output: M_{vs}: Model with selected variables
$Z_{(i,j)}, i = 1, \ldots, 7, j = 1, \ldots, 24$: Selected vars

1 $M_{vs} = \emptyset$;
2 **foreach** $i = 1, ..., 7,\ j = 1, ..., 24$ **do**
3 Extract $D_P^{(i,j)}$ from D_P;
4 Extract $D_V^{(i,j)}$ from D_V;
5 $Z = \{v_1, \ldots, v_d\}$; // Variables to test
6 $Z_{in} = \emptyset$ // Variables in eq. $E_{(i,j)}$
7 **foreach** $k = 1, \ldots, d$ **do**
8 **foreach** $v_h \in Z$ **do**
9 $Z_{in}^{k,v_h} = Z_{in} \cup v_h$;
10 Train $E_{(i,j)}^{k,v_h}$ with vars Z_{in}^{k,v_h} on $D_P^{(i,j)}$;
11 Compute $\overline{RMSE}_{(i,j)}^{k,v_h}$ of $E_{(i,j)}$ on $D_V^{(i,j)}$;
12 **end**
13 $E_{(i,j)}^{k} = \underset{E_{(i,j)}^{k,v_h}}{\arg\min}(\overline{RMSE}_{(i,j)}^{k,v_h})$; // Best eq.
14 $\overline{RMSE}_{(i,j)}^{k} = \min(\overline{RMSE}_{(i,j)}^{k,v_h}) \mid v_h \in Z$;
15 $v^k = \underset{E_{(i,j)}^{k,v_h}}{\arg\min}(\overline{RMSE}_{(i,j)}^{k,v_h})$; // Best variable
16 $Z_{(i,j)}^{k} = Z_{in} \cup v^k$
17 $Z_{in} = Z_{(i,j)}^{k}$
18 $Z = Z \setminus v^k$
19 **end**
20 $Z_{(i,j)} = \underset{Z_{(i,j)}^{k}}{\arg\min}(\overline{RMSE}_{(i,j)}^{k})$; // Best var.
21 Train $E_{(i,j)}$ with vars $Z_{(i,j)}$ on $D_P^{(i,j)} \cup D_V^{(i,j)}$;
22 $M_{vs} = M_{vs} \cup E_{(i,j)}$;
23 **end**
24 **return** $M_{vs},\ Z_{(i,j)}, i = 1, \ldots, 7, j = 1, \ldots, 24$

Since the training set $D_P^{(i,j)}$ and the test set $D_V^{(i,j)}$ are different, the $\overline{RMSE}_{1h}$ on $D_V^{(i,j)}$ usually reaches a minimum (line 20) when a subset of all variables is selected, because a too large number of variables produces overfitting, and a consequent increase of $\overline{RMSE}_{1h}$. Variable selection is performed for each equation $E_{(i,j)}$ achieving a final multi-equation model M_{vs} containing all equations. The variable selection procedure follows a forward selection strategy, it is greedy therefore it does not ensure to find the best solution. The optimal solution would however need to test all possible combinations of variables for each equation, which has a prohibitive time complexity when the number of variables grows. Our original contribution is in the integration of forward variable selection into the generation of the multi-equation model. Algorithm 1 provides all technical details about this integration. The time complexity of the proposed methodology is $O(|E_{(i,j)}| \cdot d^2 \cdot |D_P| \cdot |D_V|)$, where $|E_{(i,j)}|$ is the number of equations (see line 2 in Algorithm 1); d^2 refers to the iterations in the two for cycles of lines 7 and

8) that make $\sum_{k=1}^{d} k = \frac{d(d+1)}{2}$ iterations; $|D_P|$ considers the order of the number of steps needed to train the parameters, which depends on the dimension of the training set; K_V considers the number of steps needed to compute the RMSE on the test set.

4.3 Baseline model

The baseline model to which we compare M_{vs} is that computed using the approach proposed in [3], where all variables are used in all equations. In the following, we will call this model M_{all}. We notice that also that model is multi-equation and it has an equation for each pair day-of-the-week/hour-of-the-day.

4.4 Performance measure

The quality of the predictions made by model M_{vs} is evaluated by four measures, namely, the average Root Mean Squared Error and the average Mean Absolute Percentage Error on a 48-hour horizon ($\overline{RMSE}_{48h}$ and $\overline{MAPE}_{48h}$) and on a 1-hour horizon ($\overline{RMSE}_{1h}$ and $\overline{MAPE}_{1h}$). We compute all these measures on the test set D_T. Given an horizon of r hours, we first compute all r-hour predictions in D_T using a sliding window that moves from the beginning to the end of the dataset. For each position of the sliding window we compute the RMSE and MAPE according to the following formulas [18]:

$$RMSE = \sqrt{\frac{1}{r}\sum_{t=1}^{r}(\hat{y}_t - y_t)^2} \tag{1}$$

$$MAPE = \frac{1}{r}\sum_{t=1}^{r}\left|\frac{y_t - \hat{y}_t}{y_t}\right| \tag{2}$$

where $\hat{y}_1, \ldots, \hat{y}_r$ are predictions and $y_1, \ldots, y_r$ related ground truth values. Notice that in Algorithm 1 the performance is computed using the $\overline{RMSE}_{1h}$ on dataset D_V. The unity of measure of both RMSE and MAPE in this paper is MWh since they measure the heat load error.

In the next section we also evaluate the *variable selection capability* of Algorithm 1. This is measured as the number of parameters of M_{vs}, called *# parameters* in the following. Finally, we measure the *time* performance of the training phase and model selection phase (i.e., Algorithm 1).

4.5 XM_HeatForecast

In this section we describe XM_HeatForecast[4] [5], a software that allows DHN operators to use the methodology presented in this work in real-world applications. XM_HeatForecast generates and updates predictive models, and uses them in real-time for heat load forecasting. The tool consists of four modules, namely, i) graphical user interface; ii) model generation and prediction; iii) analysis of performance; iv) variables selection. A description is reported in the following.

[4] The Python code of XM_HeatForecast is available at `https://github.com/XModeling/XM`

i) Graphical user interface (GUI). It supports operators in the monitoring of the entire process. When the software is launched the user can select different settings, namely, a forecasting horizon (24 or 48 hours), a mode (normal or variables selection), a prediction rate (daily or hourly prediction) and the size of the training dataset. The GUI is divided into four main sections: *a)* the heat load visualization panel, shows the last 24-hour or 48-hour predictions in real-time (top-left in Figure 3.a). It allows the visualization of weather factors such as T, RH, R and W with related predictions; *b)* an area to monitor the performance of models in terms of $\overline{RMSE}$ calculated on 24h-prediction or 48h-prediction and coefficient of determination R^2 (top-right); *c)* an area displaying model parameters to analyze values for each day-of-the-week/hour-of-the-day, and the contribution of each variables prediction (bottom-left); *d)* an area which visualizes the time series in the training set used in the last update of the model (bottom-right).

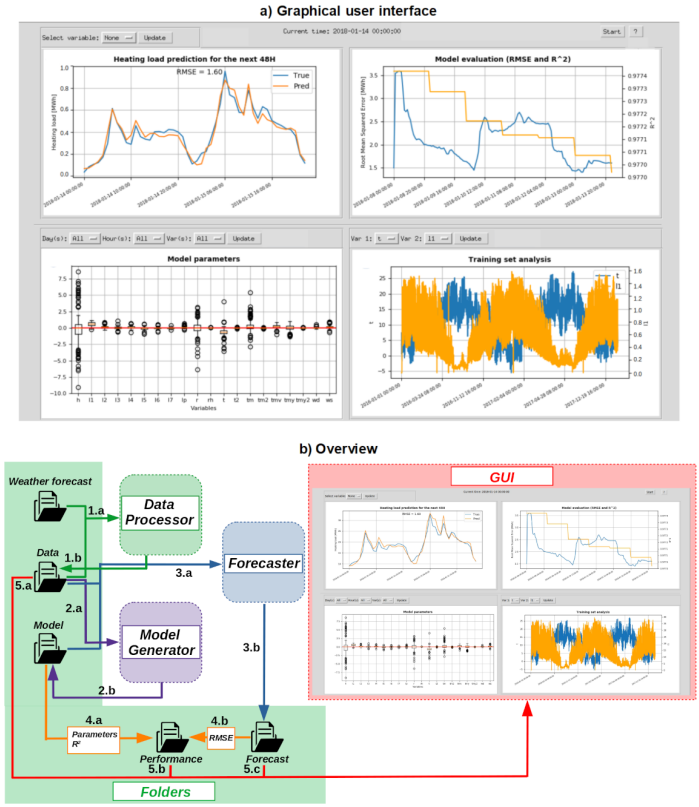

Fig. 3 XM_HeatForecast overview [5]: *a)* Main elements of the graphical user interface; *b)* Folder and module organization.

ii) Model generation and forecasting. An overview of the internal organization of the software is showed in Figure 3.b. The software trains predictive models given a dataset contained in the input folders as csv file (green and purple arrows in Figure

3.b). Model parameters are re-trained and saved automatically every 24 hours. Predictions of the next 24 or 48 hours are computed every hour by reading hourly weather forecast and historical data (green and blue arrows). The performance of the model is calculated and visualized in the GUI (orange and red arrows).

iii) Analysis of performance. The performance measures are calculated in two stages, namely, every 24 hours the software computes the $\overline{RMSE}$ and the R^2 of the re-trained model, and every hour it computes the RMSE of the last 24-hour or 48-hour prediction.

iv) Variables selection. It is an independent module introduced in the last version of the software which integrates the methodology presented in this work. The module must be executed before the forecasting phase. It generates a model using algorithm 1. The selected variables can be then used in the forecasting phase.

5 Results

In this section the result of our experimental tests are presented. We first describe the experimental setting and then compare the performance of models using all variables with that of models using only a selection of variables. Subsequently, we analyze model parameters, that provide valuable information about the importance of different variables in different days-of-the-week and hours-of-the-day.

5.1 Experimental setting

We compute two types of model. One, called M_{all}, using all variables of Table 1, and another one, called M_{vs}, using a subset of those variables, selected by the methodology presented in Section 4.2. Both types of model are trained on two datasets of different size. Dataset D_1 uses data from years 2016 and 2017 for training the model and data from 2018 for testing it. Dataset D_2 uses instead data from years 2014, 2016 and 2017 for training, and data from 2018 for testing. In M_{vs} the training set is split into two parts, namely, D_P is used for training the parameters and D_V for selecting the variables, as described in Sections 3 and 4.2. In particular, in D_1 we first use 2016 (i.e., 5184 samples) to train parameters and 2017 (i.e., 5112 samples) to select variables, and then we re-train the parameters of selected variables only with data of both 2016 and 2017 (this is why 2017 is in brackets in column D_P of dataset D_1, in Table 2). In D_2 the same procedure is performed initially using 2016 and 2017 (i.e., 10296 samples) for training parameters and 2014 (i.e., 5064 samples) for selecting variables. Afterwards, the model with selected variables is retrained using all data from 2014, 2016 and 2017 (in this case year 2014 is in brackets in column D_P, because data from that year are used to train parameters only in the last stage of the procedure). Performance are computed as RMSE and MAPE on the test set for predictions of 1 hour (i.e., $\overline{RMSE}_{1h}$ and $\overline{MAPE}_{1h}$) and 48 hours (i.e., $\overline{RMSE}_{48h}$ and $\overline{MAPE}_{48h}$). Comparison are performed in the next subsection. Model parameters are instead analyzed in Subsection 5.5. The code used for the experiments is written in Python version 3.6.9. Data cleaning and pre-processing uses *pandas* version 1.1.4. Models are generated

using *statsmodels*[5] version 012.1. Tests are run on a laptop with processor Intel Core i7 - 6500 CPU 2.50 GHz x 4, RAM 12 GB and operating system Ubuntu 18.04.5 LTS.

5.2 Global model performance

Model performance and dimensions are shown in Table 2. The first two lines concern, respectively, model M_{vs} and model M_{all} on dataset D_1, while the second two lines concern, respectively, model M_{vs} and M_{all} on dataset D_2. On dataset D_1 model M_{vs} reaches a reduction of $\overline{RMSE}_{1h}$ of 0.9% and a reduction of $\overline{MAPE}_{1h}$ of 2.4%. Errors $\overline{RMSE}_{48h}$ and $\overline{MAPE}_{48h}$ are instead reduced, respectively, of 0.8% and 1.3% with respect to M_{all}. Furthermore, this improvement has been reached with a strong simplification of the model, namely, the number of parameters is reduced of 51.8%, from 3360 to 1620. Model performance is further improved on dataset D_2 because it contains more observations for training the parameters. Model M_{vs} achieves in this case the best $\overline{RMSE}_{1h}$ of 0.992 MWh, with a reduction of 0.5% ($\overline{MAPE}_{1h}$ = 7.100 MWh (-5.7%)). The $\overline{RMSE}_{48h}$ has been even further improved to 1.295, with a 2.4% reduction ($\overline{MAPE}_{48h}$ = 7.397 MWh (-4.7%)) compared to M_{all}. The number of parameters of the best model is in this case 1688, about 49.8% less than M_{all}. The time needed to train models with variable selection (i.e., 1h 58m for D_1 and 2h 15m for D_2) is about two times that required to train the model without variable selection (i.e., 53m for D_1 and 1h 02m for D_2). The difference is related to the length of dataset D_V, since the performance of each variable combination is evaluated on that dataset.

Table 2 Model performance. *Data* is the name of the dataset; *Model* is the name of the model; D_P, D_V and D_T are the years in the dataset used for parameter learning, variable selection and testing, respectively; $\overline{RMSE}_{xh}$ and $\overline{MAPE}_{xh}$ are the performance measures; *#par* is the number of parameters in the model.

Data	Model	D_P	D_V	D_T	$\overline{\text{RMSE}}_{1h}$ [MWh]	$\overline{\text{RMSE}}_{48h}$ [MWh]	$\overline{\text{MAPE}}_{1h}$ [MWh]	$\overline{\text{MAPE}}_{48h}$ [MWh]	#par
D_1	M_{all}	16/17	-	18	1.055	1.420	7.818	8.136	3360
	M_{vs}	16/(17)	17	18	**1.046** (**-0.9%**)	**1.409** (**-0.8%**)	**7.630** (**-2.4%**)	**8.034** (**-1.3%**)	**1620** (**-51.8%**)
D_2	M_{all}	14/16/17	-	18	0.997	1.327	7.533	7.765	3360
	M_{vs}	(14)/16/17	14	18	**0.992** (**-0,5%**)	**1.295** (**-2.4%**)	**7.100** (**-5.7%**)	**7.397** (**-4.7%**)	**1688** (**-49.8%**)

We further analyze the global performance of models M_{all} and M_{vs} by computing the (discretized) distribution of their *percent prediction errors* in predictions with 1-hour (Figure 4.a) and 48-hour (Figures 4.b) horizon. Orange bars are related to model M_{all} and blue bars concern model M_{vs}. For each hourly prediction (performed in a 1-hour or 48-hour horizon) we compute the percent error with respect to the true load value. These percent values are stored and then grouped into the buckets (i.e., percent intervals) listed in the x-axis of Figures 4.a and 4.b.

[5] https://www.statsmodels.org/stable/index.html

Table 3 Time performance. *Time for variable selection* is the time required by Algorithm 1. *Time for parameter training* is the time needed to train model parameters when variables are known. *Time for prediction* is the time required to make all 48-hour predictions in the test set (see test set definitions in Table 2).

Data	Model	Time for variable selection	Time for parameter training	Time for prediction	Total time
D_1	M_{all}	-	3m, 27s	49m, 33s	53m
	M_{vs}	1h, 5m	2m, 45s	50m, 15s	1h, 58m
D_2	M_{all}	-	5m	57m	1h, 2m
	M_{vs}	1h, 16m	4m	55m	2h, 15m

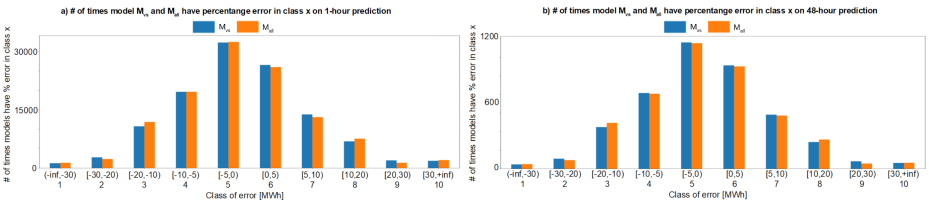

Fig. 4 Distribution of the percent RMSE prediction error [MWh].

Good predictions are in central buckets, which correspond to intervals $[-5\%, 0\%)$ and $[0\%, 5\%)$, while bad predictions are in the two sides, corresponding to intervals $(-\infty, -30\%]$ and $[30\%, +\infty)$. Both histograms in Figures 4.a and 4.b show a larger or equal amount of blue mass with respect to orange mass in the three central intervals (i.e., [-10%,-5%], [-5%,0%], [0%,+5%]). In the rest of the bars there seems to be no specific pattern among the two predictors.

Time performance is shown in Table 3. The total time is split in *Time for variable selection*, *Time for parameter training* and *Time for prediction* on dataset D_1 and D_2. The total time for generating the model (column *total time*) with variable selection is about one hour more than that for generating the full model. This difference is mainly due to the variable selection phase (column *time for variable selection*) that takes more than one hour to be completed. We remind that in both datasets D_1 and D_2 variables have been selected using data from a single year (i.e., 2017 in D_1 and 2014 in D_2). The parameter training phase takes around three minutes in dataset D_1, where data from two years are used, and around four minutes and a half in dataset D_2, where data from three years are used. The time required to preform all 2449 48-hour predictions in the test set (a 48-hour prediction is performed every hour) is around fifty minutes, namely every 48-hour prediction takes about 1.23 seconds.

5.3 Hourly model performance

We split errors $\overline{RMSE}_{1h}$ and $\overline{RMSE}_{48h}$ into 168 parts, one for each equation $E_{(i,j)}$ which refers to a specific day-of-the-week (i.e., i) and hour-of-the-day (i.e., j). The interest in this analysis is related to the possibility to identify temporal patterns in prediction errors. Figure 5 contains results of a quantitative analysis of these patterns. In particular, the heatmaps in Figures 5.a and 5.b show, respec-

tively, the breakdown in day-of-the-week/hour-of-the-day of error $\overline{RMSE}_{1h}$ and $\overline{RMSE}_{48h}$. Rows are hours of the day (from 0 to 23), columns are days of the week (from Monday to Sunday), and each cell contains the average RMSE for a day-of-the-week/hour-of-the-day. Predictions with horizon of 1 hour show larger error variability than predictions with horizon of 48 hours, as expected, because the error in the second case is averaged over 48 hourly predictions. The visual analysis of heatmaps in Figures 5.a and 5.b is enriched by the visualization of line charts in Figures 5.c and 5.d. They display, respectively, hourly $\overline{RMSE}_{1h}$ and $\overline{RMSE}_{48h}$ concatenated on a column basis (i.e., day-by-day). Moreover, they show also the hourly $\overline{RMSE}_{1h}$ and $\overline{RMSE}_{48h}$ of model M_{all} (see the orange line). Then, Figures 5.e and 5.f keep only the hour of the day in the x-axis and use different lines an colors for different days of the week, favouring a better comparison between errors of different days of the week at the same hour.

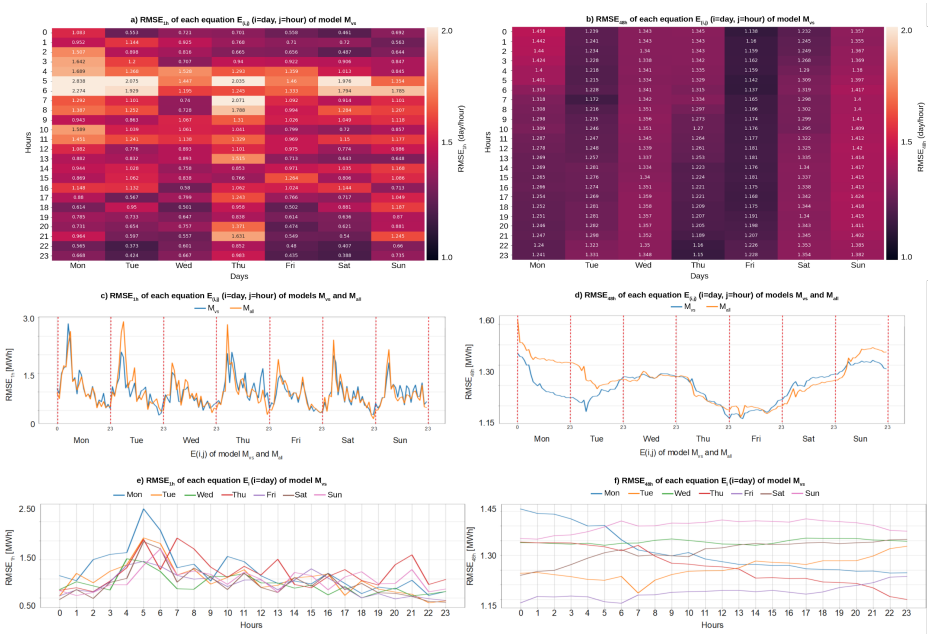

Fig. 5 Peformance of models M_{vs} and M_{all} calculated on test set D_T in the experiment on dataset D_1: a) $RMSE_{1h}$ of each equation $E_{(i,j)}$ (i.e., day-of-the-week/hour-of-the-day) on 1h-predictions (measured in MWh); b) $RMSE_{48h}$ of each equation $E_{(i,j)}$ (i.e., day-of-the-week/hour-of-the-day) on 48h-predictions (measured in MWh); c) Comparison of $RMSE_{1h}$ distributions over the weekday of models M_{vs} and M_{all} on 1h-prediction (measured in MWh); d) Comparison of $RMSE_{48h}$ distributions over the weekday of models M_{vs} and M_{all} on 48h-prediction (measured in MWh); e) $RMSE_{1h}$ over the hour-of-the-day for each day-of-the-week (equation E_i); f) $RMSE_{48h}$ over the hour-of-the-day for each day-of-the-week (equation E_i).

The analysis of Figures 5.a, 5.c and 5.e, shows that the hourly $\overline{RMSE}_{1h}$ has a main peak at 5.00, which corresponds to the peak-hour also for the heat load. The larger error of model M_{vs} is on Monday at 5.00, with a value of 2.838 MWh. Afterwards, there are Tuesday, Thursday and Saturday, that have an error of about 2.000 MWh at the same hour; then there is Sunday, with an error of 1.785 MWh

and, finally, Wednesday and Friday with peak errors of about 1.500 MWh. The hourly error is on average high (i.e., between 1.500 MWh and 2.900 MWh) in the early morning from 4.00 to 7.00, then it has medium values (between 0.800 MWh and 1.500 MWh) from 8.00 to 16.00, and low values (between 0.4 MWh and 1.000 MWh) in the evening/night from 17.00 to 3.00. We also observe a pattern related to the days of the week, with highest values (on average) on Monday, then a decrease on Tuesday and another decrease on Wednesday. Strangely, Thursday sees an increase of the error which is followed by another decrease on Friday. This makes Tuesday, Wednesday and Friday similar in terms of average daily error, while Monday and Thursday have higher values. The behaviour of Monday is motivated by the discontinuity with respect to the weekend, which makes the network producing more heat. Sunday has an increment of both peak and average hourly errors with respect to Friday, because of discontinuity of people behaviour in that day of the week. Also Sunday has a different error profile, but in this case the error increases mainly in the afternoon and evening with respect to all the other days. A final consideration is related to error differences between models M_{vs} and M_{all}. The model with selected variables (blue line) has better performance in all peaks except that of Monday, where errors are very similar. The two models have similar error profiles, with slightly smaller errors for M_{vs} in general, with a few exceptions, for instance on Thursday at 20.00 and 21.00, and on Sunday late morning and afternoon.

Figures 5.b, 5.d and 5.f focus on 48-hour predictions, which is a standard horizon for operational planning in utility companies. Error values are much smoother than in 1-hour predictions because they are averaged over two days. What emerges is an oscillatory error profile with weekly period, two maxima (one global and one local) and two minima (one global and one local). The global maximum is on Sunday, with highest value on Monday at midnight (i.e., 1.458 MWh), and the local maximum is on Wednesday with highest value at 17.00 (i.e., 1.359 MWh). In these instants 48-hour predictions have their highest average error. The global minimum is instead in the night between Thursday and Friday, with lower value of 1.137 MWh on Friday at 6.00. The local minimum is instead on Tuesday at 7.00 with a value of 1.172 MWh. Models M_{vs} and M_{all} perform similarly from Wednesday to Saturday, while M_{vs} clearly outperforms M_{all} in 48-hour predictions taken from Sunday to Tuesday.

5.4 Comparison with methods in the literature

In this section we compare our best model M_{vs}, generated using Algorithm 1, with some state-of-the-art methodologies, namely, CNN, LSTM and Least Shrinkage and Selection Operator (Lasso). For each of those methods we describe architecture, learning parameters and performance. All models have been tested on dataset D_2, and corresponding training and test sets described in Table 2. Table 4 summarizes the performance of all methods and allows to perform a comparative analysis.

CNN. We trained a CNN with architecture as in [4]. The input is a matrix having a column for each variable and a row for each time instant (168 rows in total). The first layer is composed of five neurons with *ReLu* activation function

and convolution of the input performed by a kernel having the same dimension of the input. The five features maps obtained are then passed to a dense layer, which computes their linear combination. The model is implemented using Keras[6] and trained on dataset D_2. Weights are initialized by *Xavier* normal initializer and learned by gradient descent with *Adam* and performing 20 epochs using batch size equals to 32. Early stopping procedure is used to avoid overfitting.

LSTM. We generated 15 different LSTM models by varying architecture and hyperparameters. In particular, we tested i) number of layers from 1 to 5, ii) number of neurons per layer from 24 to 168, iii) regularization factors between layers using dropout with rates between 0.5 and 0.8 . Then we selected the model having the best performance on the test set. The input of the model is a matrix having one column for each variable and one row for each time instant (168 rows in total). The first layer is composed of 168 recurrent units that use hyperbolic tangent (i.e., *tanh*) as activation function and sigmoid as recurrent activation step function. The output of the recurrent layer is linearly combined in a final dense layer. The model is trained and tested on dataset D_2, with Keras, for 20 epochs using batch size of 32 and early stopping procedure for best set of weights selection.

Starting from this (best) model we also tested some local changes to confirm its goodness. Specifically, we first tested a number of training epochs between 20 and 60. The training task is equipped with an early stopping procedure that speeds up the process and avoids overfitting. The training stage is stopped when the loss function converges and the performance on the validation set stops improving. We observe that in tests with 60 epochs the early stopping procedure converged after 45 epochs. Moreover, we performed new tests concerning: i) the addition of a dense layer with 48 neurons after the first LSTM layer having 168 recurrent units, ii) the addition of two dense layers with 48 and 24 neurons, respectively, after the first LSTM layer having 168 recurrent units, iii) regularization factors between layers using dropout with rates equals to 0.5, iv) the usage of GRU units instead of LSTM units. In all tests (i.e., combinations of network architecture and number of epochs) the performance did not improve with respect to the best model (see the performance of the best model in Table 4).

Since LSTM usually have good performance on time series prediction, we clarify that the motivation of their low performance compared to the multi-equation regression model lies in the structure of the regression model. In particular, by splitting the regression model into 168 equations (i.e., one equation for each day-of-the-week/hour-of-the-day) we introduce important prior knowledge about the problem into the model. In fact, the behaviour of the heating load in specific day-of-the-week/hour-of-the-day is simple and linear with respect to some factors (mainly the temperature). This is because day-of-the-week and hour-of-the-day contain information about the (important) social component of the prediction problem. The LSTM model, on the other hand, is unique, namely, it does not use one network for each day-of-the-week/hour-of-the-day, hence it should be able to infer the social component from data. The low performance of the LSTM model shows that inferring this information from data is very difficult, hence providing it a priory gives a strong advantage. One could ask why we did not computed a multiple LSTM model using a network for each day-of-the-week/hour-of-the-day. The answer is that neural networks need large amounts of data, and splitting the

[6] `https://keras.io/`

dataset by day-of-the-week/hour-of-the-day generates 168 small datasets containing on average 62 samples each, in our case, which is not enough for training a neural network. The simplicity of the regression model allows in this case to learn an accurate model with little data, achieving better performance than neural networks and also a very good interpretability.

Lasso. We generated a linear regression model using Lasso for variable selection. We used the *lassoCV* function from *sklearn* library[7]. The length of regularization path ϵ was set to $5e^{-3}$ and the number of αs along the regularization path to 100. The best model was selected by 10-fold cross-validation.

Performance comparison. The performance of different methods is shown in Table 4. We show average RMSE and average MAPE on 1 hour and 48 hour horizons, number of parameters and training time for model M_{vs} trained and tested on dataset D_2. The model M_{vs} generated by the approach proposed in this work achieves the best performance in terms of prediction quality. The methodology that selects the smaller number of variables is Lasso (13 variables) and the best training time is achieved using the CNN (41 minutes). The $\overline{RMSE}_{1h}$ reached by M_{vs} is 0.992 MWh, about 15% less than the performance of the second model in the ranking, which is the CNN with 1.143 MWh. Similarly, the $\overline{RMSE}_{48h}$ is 1.295 MWh for M_{vs} and 1.455 MWh for the CNN, with an increase of 12.4%. The number of parameters of model M_{vs}, namely 1688, is the second smallest. Only the lasso regression model has less parameters, namely 13, because it is a single equation model. Its prediction error is however almost double (e.g., $\overline{RMSE}_{48h} = 2.553$ MWh). This shows that the multi-equation structure of our model is fundamental for reducing the prediction error. Neural network based models have much more parameters than our model. In particular, the CNN model has 18491 parameters (more than 10 times the parameters of M_{vs}) and the LSTM has 126505 parameters (almost 100 times the parameters of M_{vs}). The fact that the 1688 parameters of our model are also split in 168 equations, with an average of 10 parameters per equation, highlights also the sparseness and the consequent good interpretability of our model comparing to neural network based models. The training time of our method is the highest (i.e., 1 hour and 20 minutes) but we remind that, as displayed in Table 2, the majority of this time is used only the first time to perform variable selection (which needs 1 hour and 16 minutes, see Table 2); instead only 4 minutes are needed to compute the parameters if the set of variables have already been defined. Considering this, the training time is comparable to that of the other methodologies, and in any case acceptable for applications in which the model is re-trained only seldom (e.g., once a day).

Table 4 Comparison of global performance with the state-of-the-art

Model	$\overline{\text{RMSE}}_{\text{1h}}$ [MWh]	$\overline{\text{RMSE}}_{\text{48h}}$ [MWh]	$\overline{\text{MAPE}}_{\text{1h}}$ [MWh]	$\overline{\text{MAPE}}_{\text{48h}}$ [MWh]	# par	Time for training	Time for prediction
$\mathbf{M_{vs}}$	**0.992**	**1.295**	**7.100**	**7.397**	1688	1h, 20m	55m
CNN	1.143	1.455	9.700	10.074	18491	20m	**21m**
Lasso	2.553	2.498	17.492	16.401	**13**	**5m**	40m
LSTM	4.953	5.944	33.000	33.163	126505	39m	42m

[7] https://scikit-learn.org/stable/

In Figure 6 we show the hourly performance of models in terms of $RMSE_{1h}$. The hourly error profile of our model M_{vs} is represented by a blue line. It has values lower than the other methods in almost all the time instants (i.e., x-axis). Only the CNN model has a similar profile but with slightly larger values on average.

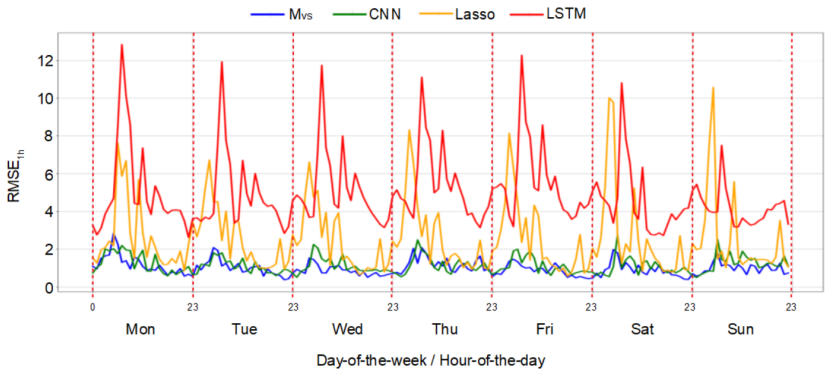

Fig. 6 $RMSE_{1h}$ of the four models M_{vs}, CNN, Lasso and LSTM for each day-of-the-week/hour-of-the-day.

5.5 Analysis of model parameters

In this section we analyze and compare the parameters of models M_{vs} and M_{all}. This analysis provides valuable knowledge about the role of different variables in the prediction. Some questions that we answer here are, for instance, "Which variables are most informative for predicting future heat loads?" and "Are these variables selected in all equations or only in some of them?", and also "Which variables are more informative for the prediction made at 6.00? And for those made on Sunday?" or "Does relative humidity influence the heat load? How much?". We can extract such information from model parameters. The section is split in six main parts, focusing on: *i)* variables selected in each equation; *ii)* variables selected in each day of the week; *iii)* variables selected in each hour of the day; *iv)* number of variables in each equation; *v)* sensitivity of model performance to historical, meteorological and social factors; *vi)* sensitivity of model performance to non-accurate sensor signals.

Variables selected in each equation. The heatmaps in Figure 7.a and Figure 7.b show, respectively, the variables selected in each equation of model M_{vs} and M_{all}. Rows represent the 19 available variables and columns the 168 equation $E_{(i,j)}$. Cell colors are red for negative parameters (i.e., the variable has been selected by the equation and the coefficient is negative), green for positive parameters, blue for non-selected variables and grey for selected variables with null coefficient (this case occurs only for variable H in Figure 7.b, since it is a binary value). Rows are sorted by selection rate, namely, on top there are variables selected by several equations and in the bottom variables selected by few equations. The horizontal

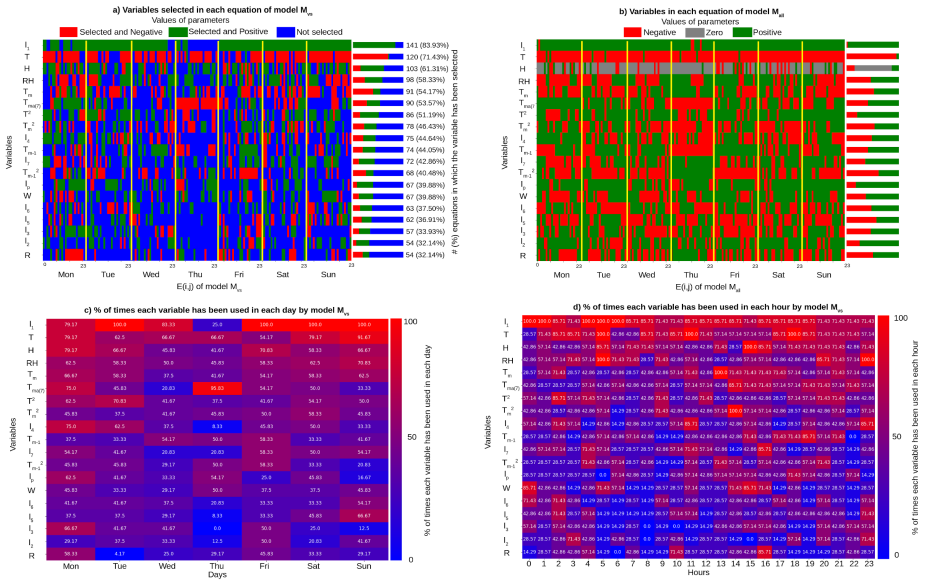

Fig. 7 Variables selected by model M_{vs} in the experiment on dataset D_1: a) Variables selected in each equation $E_{(i,j)}$; b) Percentage of times each variable has been used in each day; c) Percentage of times each variables has been used in each hour

bar chart on the right of Figure 7.a shows the selection rate and the percentage of negative and positive parameters, while the same chart in Figure 7.b shows only the percentage of negative and positive parameters since all variables are always selected in M_{all}.

The first information that we get from Figure 7.a is that the most selected variable is the *load of the previous day* l_1, selected in 141 equations out of 168 (i.e., in 83.93% of the equations). This variable has a positive parameter in all equations, since the past load is used as a reference for the prediction, namely, if that value is high then probably also the predicted value is high, while if the load at the previous day is low then it is probable that also the predicted load is low, under the same weather conditions. The second most selected variable is *temperature* T, selected by 120 equations out of 168 (i.e., in 71% of the equations) and always with negative parameters. This is also expected, since the heat load is known to inversely depend on the temperature. Namely, when the temperature decreases, the heat load required by the the building is higher. The third most selected variable is *holiday* H, with 61.31% of equations, and the fourth is relative humidity, with 58.33% of equations. These two variables have positive parameters in the majority of cases but in some equations (i.e., weekday/hour) have also negative parameters. Finally, the two least selected variables are *rainfall* R and the *load of two days before* l_2, both present in 32.14% of equations.

Other interesting information can be extracted from Figures 7.a. For instance, the wind speed W is selected in 39.88% of equations with positive parameters in the majority of cases. Moreover, the second most selected variable related to the temperature is the maximum temperature of the day T_m (54.17% of equations), which is known by experts to be a good proxy for *thermal radiation*, that nega-

tively influences heat load because buildings radiated by the sun are warmed up more than not irradiated buildings, at the same temperature. The second most selected variable related to past heat load is the load of four days before l_4 (44.64% of equations) and the next one is the load of the previous week l_7 (42.86% of equations). The second was expected to be more frequently selected because the weekly behaviour is strongly affected by social factors (e.g., people tend to have similar behaviours in the same days of the week). Finally, the comparison between Figure 7.a and 7.b shows that model M_{vs} has a more simplified structure since almost the half of parameters are not used (see blue areas in the heatmap). Furthermore, the removal of unnecessary parameters also highlights the ratios between positive and negative parameters.

Variables selected in each day of the week. We compute the variable selection rate for each day of the week to get information about the percentage of equations each variable has been selected in different days of the week. The result is shown in Figure 7.c. We initially observe that variables selected in equations related to Thursday are somehow different from those related to other days. Mainly, only 25% equations use l_1 and this variable seems to be substituted by $T_{ma(7)}$, the moving average of the temperature in the past week, which is selected several times (in 95.83% equations related to Thursday. Strangely enough, all variables related to past loads (e.g., l_4, l_7) are selected a smaller number of times than in other days, with the only exception of the load at the previous peak l_p, which has been selected in the 54.17% of the equations (the second higher percentage for that variable). Then, the day with the larger number of variables is Monday, with 62.06% of equations selected per variable, on average. The day with the smaller number of variables is instead Thursday, with 37.06% of equations selected per variable, on average. This shows that the structure of the predictions made on Monday is the most complex, possibly because of the discontinuity with the weekend. Monday is also the day in which variables l_3 and l_4 are most selected (in 66.67% and 75.0% of equations, respectively). This is because the load of the previous two days refers to the weekend, in which people use the heating differently than in working days. Also variable l_7, which refers to the load of 7 days before, is used quite a lot on Monday (54.17% of equations) and also on Friday, Saturdays and Sunday, namely, in days in which the load has a discontinuity with respect to previous days, hence information is gathered from the previous week.

Variables selected in each hour of the day. Figure 7.d shows an equivalent heatmap where columns are the hours of the day (instead of the days of the week). This chart shows, for instance, that the most selected variables for making predictions in the peak hour (corresponding to models 5.00) are l_1 (selected in 100% equations), T (100% equations), H (85.71% equations) and RH (100% equations). The temperature is also always selected by models of 11.00 and 18.00, while the maximum temperature T_m is often selected between 13.00 and 17.00, which are the warmest hours of the day. RH has the highest selection rate in the first part of the night (from 20.00 to 23.00) in addition to the peak hour. This is because the relative humidity is higher in that part of the day, hence its value can influence the prediction. Also for rainfall R we observe that the highest selection rates correspond to 10:00 and 16:00 that coincide with hours of the day in which maximum amount of precipitations have been observed in the training set.

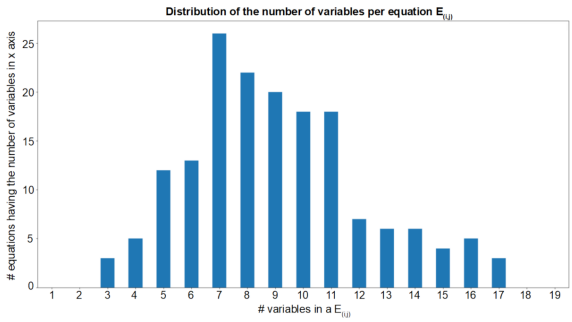

Fig. 8 Distribution of the number of variables in each equation $E_{(i,j)}$ of model M_{vs}

Number of variables in each equation. Another analysis we performed, concerns the distribution of the number of variables per equation. This number can be from 0 to 19 since the total number of available variables is 19. Figure 8 shows a bar chart in which the x-axis represents the number of variables in an equation $E_{(i,j)}$ and the y-axis the number of equations having that number of variables. The mode is 7, with 29 equations having this number of variables. The maximum is 17 with 3 equations (i.e., $E_{(2,20)}$, $E_{(5,23)}$ and $E_{(6,11)}$) having 17 variables, and the minimum is 3, with 3 equations (i.e., $E_{(3,9)}$, $E_{(4,13)}$ and $E_{(5,7)}$) having 3 variables. For instance, equation $E_{(3,9)}$ has the following form $E_{(3,9)} = -0.970 + 0.937 \cdot l_1 + 0.021 \cdot RH - 0.150 \cdot R$. The average number of variables is 9, which corresponds to the 47.37% of 19, hence the reduction introduced by the variable selection method is considerable.

Sensitivity of model performance to historical, meteorological and social factors. We evaluated the performance of model M_{vs} trained on dataset D_2 when some variables are removed from the set of available variables. In particular, we removed one by one the following factors (and all related variables): past loads $l_i, i = 1, \ldots, 7$, temperature T, wind W, relative humidity RH, rainfall R and holidays H. Table 5 shows the change of performance achieved for each removed factor. Notice that when temperature is removed, also all variables related to temperature (e.g., T^2, T_m) are removed. The factor that most influences the $\overline{RMSE}_{48h}$ is the past load (+39%), then comes the temperature (+32%). The next factors, having a much smaller impact are the wind (+0.5%), the relative humidity (+0.3%) and holidays (+0.1%). The removal of the rainfall yields an error reduction (-0.7%). This can happen because the proposed variable selection method computes the set of variables using a greedy strategy (for complexity reasons), hence the solution that it computes is not guarantee to be the best solution. Computing the best solution would require to test all possible combinations of variables which is impractical for large number of variables.

Sensitivity of model performance to non-accurate sensor signals. As a final analysis we perform an experiment to evaluate the importance of using "trusted" sensors to collect data. As described in Section 3 model M_{vs} is trained using a temperature signal collected by a sensor certified by AGSM and located in the area of the buildings served by the network. Here, we compare model M_{vs} trained with dataset

Table 5 Sensibility of $\overline{RMSE}_{48h}$ to the removal of historical factors (i.e., past loads l), meteorological factors (i.e., temperature T, wind W, relative humidity RH, rainfall R) and social factors (i.e., holidays H). First row: $\overline{RMSE}_{48h}$. Second row: percentual increment of $\overline{RMSE}_{48h}$. Symbol $\mathbf{M_{vs}^{-x}}$ represents model obtained using variable selection on dataset D_2 and removing factor x.

$\mathbf{M_{vs}}$	$\mathbf{M_{vs}^{-l}}$	$\mathbf{M_{vs}^{-T}}$	$\mathbf{M_{vs}^{-W}}$	$\mathbf{M_{vs}^{-RH}}$	$\mathbf{M_{vs}^{-H}}$	$\mathbf{M_{vs}^{-R}}$
1.295	1.802	1.711	1.301	1.299	1.296	1.286
	+39%	+32%	+0.5%	+0.3%	+0.1%	-0.7%

D_2 with a model M'_{vs} trained with the same dataset having the temperature signal gathered in a nearby area. In particular, the new temperature signal has been collected by a weather station placed in the Villafranca airport, about 12.5km away from the city center of Verona. Data has been collected from the same website from which also RH, W and other variables were collected because we did not have any certified sensor for those variables. Results show that the change of information source generates a performance decrease in both 1-hour predictions and 48-hours prediction. Namely, M'_{vs} has $\overline{RMSE}_{1h} = 1.048$ MWh (+5.6%) and $\overline{RMSE}_{48h} = 1.385$ MWh (+6.9%). This result highlights the importance of quality signals for weather and social factors. Moreover, it suggests that the introduction of more accurate signals for relative humidity, wind and rainfall could increase the performance of the proposed models.

6 Conclusions

The methodology here presented generates multi-equation predictive models having good performance in heat load prediction from weather and social factors in DHNs. An important property of the proposed model is also interpretability, because it splits the prediction problem in sub-problems related to different times of the week that are representable by simple multivariate autoregressive linear models. We performed an in-depth analysis of this model providing valuable insight about the process investigated. Future work will focus on two main lines, the automatization of feature engineering through representation learning techniques and the integration of the predictive model into a planner for optimizing the operational decisions taken by the utility company.

Acknowledgements The research has been partially supported by the projects "Dipartimenti di Eccellenza 2018-2022, funded by the Italian Ministry of Education, Universities and Research (MIUR), and "GHOTEM/CORE-WOOD, POR-FESR 2014-2020", funded by Regione del Veneto.

References

1. Ahmed, N.K., Atiya, A.F., Gayar, N.E., El-Shishiny, H.: An empirical comparison of machine learning models for time series forecasting. Econometric Reviews **29**(5-6), 594–621 (2010)
2. Bengio, Y., Courville, A., Vincent, P.: Representation learning: A review and new perspectives. IEEE Transactions on Pattern Analysis and Machine Intelligence **35**(8), 1798–1828 (2013)

3. Bianchi, F., Castellini, A., Tarocco, P., Farinelli, A.: Load forecasting in district heating networks: Model comparison on a real-world case study. In: Machine Learning, Optimization, and Data Science LOD 2019, vol. 11943, pp. 553–565. Springer International (2019)
4. Bianchi, F., Castellini, A., Tarocco, P., Farinelli, A.: Convolutional neural network and stochastic variational gaussian process for heating load forecasting. In: Machine Learning, Optimization, and Data Science LOD 2020, vol. 12565, p. 720. Springer International (2020)
5. Bianchi, F., Masillo, F., Castellini, A., Farinelli, A.: XM HeatForecast: Heating load forecasting in smart district heating networks. In: Machine Learning, Optimization, and Data Science LOD 2020, vol. 12565, p. 720. Springer International (2020)
6. Box, G., Jenkins, G.M.: Time Series Analysis: Forecasting and Control. Holden-Day (1976)
7. Camero, A., Alba, E.: Smart city and information technology: A review. Cities **93**, 84 – 94 (2019)
8. Castellini, A., Bianchi, F., Farinelli, A.: Predictive model generation for load forecasting in district heating networks. IEEE Intelligent Systems pp. 1–8 (2020)
9. Chen, T., Guestrin, C.: XGBoost: A scalable tree boosting system. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD'16, pp. 785–794. ACM, New York,NY,USA (2016)
10. Dahl, M., Brun, A., Kirsebom, O.S., Andresen, G.B.: Improving short-term heat load forecasts with calendar and holiday data. Energies **11**(7), 1678 (2018)
11. Elattar, E., Sabiha, N., Alsharef, M., Metwaly, M., Abd-Elhady, A., Taha, I.: Short term electric load forecasting using hybrid algorithm for smart cities. Applied Intelligence (2020)
12. Fang, T.: Modelling district heating and combined heat and power. Ph.D. thesis, Aalto University publication series Doctoral Dissertations; 107/2016; Aalto-yliopisto (2016)
13. Fang, T., Lahdelma, R.: Evaluation of a multiple linear regression model and SARIMA model in forecasting heat demand for district heating system. Applied Energy **179**, 544–552 (2016)
14. Gelazanskas, L., Gamage, K.A.: Demand side management in smart grid: A review and proposals for future direction. Sustainable Cities and Society **11**, 22 – 30 (2014)
15. Gong, M., Zhou, H., Wang, Q., Wang, S., Yang, P.: District heating systems load forecasting: a deep neural networks model based on similar day approach. Advances in Building Energy Research pp. 1–17 (2019)
16. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press (2016)
17. Gross, G., Galiana, F.: Short-term load forecasting. Proceedings of the IEEE **75**(12), 1558–1573 (1987)
18. Hastie, T., Tibshirani, R., Friedman, J.: The elements of statistical learning. Springer, New York, NY, USA (2009)
19. Jacob, M., Neves, C., Vukadinović Greetham, D.: Short Term Load Forecasting, pp. 15–37. Springer International Publishing, Cham (2020)
20. Li, C., Li, S., Liu, Y.: A least squares support vector machine model optimized by moth-flame optimization algorithm for annual power load forecasting. Applied Intelligence **45**(4), 1166–1178 (2016)
21. Lim, B., Zohren, S.: Time series forecasting with deep learning: A survey. Philosophical transactions of the Royal Society A mathematical, physical and engineering sciences **379** (2021)
22. Mirowski, P., Chen, S., Ho, T., Yu, C.: Demand forecasting in smart grids. Bell Labs Technical Journal pp. 135–158 (2014)
23. Omitaomu, O.A., Niu, H.: Artificial intelligence techniques in smart grid: A survey. Smart Cities **4**(2), 548–568 (2021)
24. Panagiotelis, A., Athanasopoulos, G., Hyndman, R., Jiang, B., Vahid, F.: Macroeconomic forecasting for Australia using a large number of predictors. International Journal of Forecasting **35**(2), 616–633 (2019)
25. Rabiner, L.: A tutorial on hidden markov models and selected applications in speech recognition. Proceedings of the IEEE **77**(2), 257 –286 (1989)
26. Ramanathan, R., Engle, R., Granger, C., Vahid-Araghi, F., Brace, C.: Short-run forecast of electricity loads and peaks. Int. J. Forecasting **13**, 161–174 (1997)
27. Roberts, S., Osborne, M., Ebden, M., Reece, S., Gibson, N., Aigrain, S.: Gaussian processes for time-series modelling. Phil. Trans. Royal Soc. (Part A) **371** (2013)
28. S.Buffa, Cozzini, M., D'Antoni, M., Baratieri, M., Fedrizzi, R.: 5th generation district heating and cooling systems: A review of existing cases in Europe. Renewable and Sustainable Energy Reviews pp. 504–522 (2019)

29. Sheng, Z., Wang, H., Chen, G., Zhou, B., Sun, J.: Convolutional residual network to short-term load forecasting. Applied Intelligence **51**(4), 24852499 (2021)
30. Soares, L., Medeiros, M.: Modeling and forecasting short-term electricity load: A comparison of methods with an application to brazilian data. International Journal of Forecasting **24**, 630–644 (2008)
31. Suryanarayana, G., Lago, J., Geysen, D., Aleksiejuk, P., Johansson, C.: Thermal load forecasting in district heating networks using deep learning and advanced feature selection methods. Energy **157**, 141 – 149 (2018)
32. Zanella, A., Bui, N., Castellani, A., Vangelista, L., Zorzi, M.: Internet of things for smart cities. IEEE Internet of Things Journal **1**(1), 22–32 (2014)