A novel procedure for real-time SOH estimation of EV battery packs based on Time Series Extrinsic Regression

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2024.0429000

A novel procedure for real-time SOH estimation of EV battery packs based on Time Series Extrinsic Regression

RAIMONDO GALLO[2], (Member, IEEE), TOMMASO MONOPOLI[4], MARCO ZAMPOLLI[4], RÉMI JACQUES PHILIBERT JABOEUF[4], PAOLO TOSCO[4], EDOARDO PATTI[2,3], (Member, IEEE), and ALESSANDRO ALIBERTI[1], (Member, IEEE)

[1]Inter-university Dept. of Regional and Urban Studies and Planning, Politecnico di Torino, 10129 Torino, Italy (e-mail: name.surname@polito.it)
[2]Department of Control and Computer Engineering, Politecnico di Torino, 10129 Torino, Italy (e-mail: name.surname@polito.it)
[3]Energy Center Lab, Politecnico di Torino, 10129 Torino, Italy (e-mail: name.surname@polito.it)
[4]Edison S.p.A., Milano, Italy (e-mail: name.surname@edison.it)

Corresponding author: Alessandro Aliberti (e-mail: alessandro.aliberti@polito.it).

ABSTRACT One of the critical challenges posed by the spread of Lithium-ion Batteries (LIBs) within Electric Vehicles (EVs) is the real-time estimation of their State-of-Health (SOH), commonly regarded as the leading indicator of EV aging. However, SOH estimation is still challenging due to the electrochemical complexity of LIBs. This work proposes a novel, computationally-inexpensive, and chemically agnostic Machine Learning (ML) procedure for onboard real-time SOH estimation. The proposed methodology requires a narrow time window of voltage, current, and State-of-Charge battery data, collected while driving the vehicle. We defined a Simulink-based EV simulator, modeling a specific real-world EV, and we utilized it to generate a synthetic dataset by simulating multiple driving sessions of the EV to compensate for the lack of large-scale publicly available EV monitoring data. Then, we examined three feature extraction methods and three ML regression models, estimating the battery pack's SOH. We conducted a thorough comparison of the proposed feature extraction methods and ML models, training the ML model with processed synthetic data and inferring over real driving session monitoring data from the corresponding real-world EV model. The best synthetic-trained ML model achieves an MAE of 0.27% and 5.08%, and an RMSE of 0.37% and 5.92% over synthetic and real test data, respectively. Finally, we implemented transfer learning over the ML models, employing a portion of the available real data, reaching the lowest MAE of 1.97%, and an RMSE of 2.56% over the remaining real test set.

INDEX TERMS Electric Vehicle, Battery Pack, Regression, Machine Learning, State-of-Health.

NOMENCLATURE

DD	Data-Driven
EChM	ElectroChemical Models
ECM	Equivalent Circuit Models
EM	Empirical Models
EOL	End-of-Life
EV	Electric Vehicle
FFNN	Feed-Forward Neural Network
LIB	Lithium-ion Battery
MAE	Mean Absolute Error
MB	Model-based
ML	Machine Learning
MR	MiniRocket
OLS	Ordinary Least Squares
PCA	Principal Component Analysis
RF	Random Forest
RMSE	Root Mean Squared Error
RR	Ridge Regression
SOC	State-of-Charge
SOH	State-of-Health
TL	Transfer Learning
TS	Theil-Sen

I. INTRODUCTION

Mitigating climate change is considered one of the critical challenges of our century. It is estimated that the transport sector accounts for 27 of the global emissions of greenhouse gases [1] and, more specifically, road travel accounts for three-quarters of transport CO_2 emissions [2]. Research and development toward a greener automotive industry have gained worldwide momentum in recent years. Electric Vehicles (EVs) have been widely accepted as a clean and reliable alternative to fossil fuel vehicles, both in the private and public transportation sectors. They are expected to take over the market quickly in the upcoming years [3]. It is, therefore, essential to investigate the leading technologies that enhance the performance of an EV.

The core component of an EV is its rechargeable battery pack, and its design is critical in determining the energy efficiency and driving range of an EV. A battery pack typically comprises many battery cells connected in parallel and series. Lithium-ion Battery (LIB) cells are currently the dominating technology in battery pack design thanks to favorable properties such as high energy density and efficiency, low memory effects, long cycle life, low self-discharge rate, high charging and discharging rate capability, and - last but not least - plummeting costs of their composing materials. EV battery packs are hierarchically structured into three levels: cell, module, and pack [4]. Multiple battery cells are connected in series and parallel to form a battery module, and then a certain number of modules are assembled to form a battery pack. This design, shown in Figure 1, aims to achieve a sufficiently high terminal voltage while adding up the capacity of each individual cell.

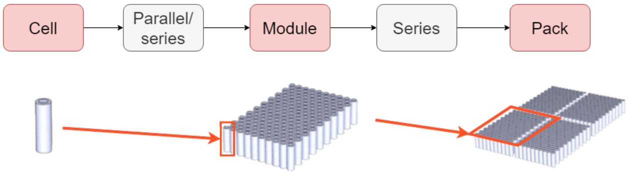

FIGURE 1. The hierarchical structure of an EV battery pack [5].

A parameter to describe the electrochemical state and dynamics of the battery cells is the capacity, defined as the total amount of charge it can deliver and commonly measured in ampere-hours [Ah]. Manufacturers usually declare the nominal (or rated) capacity, $C_{nominal}$, as the battery's capacity when fresh out of the factory. Multiple external factors determine the effective battery capacity, C_{actual} at any given moment. LIB cells, like all batteries, are subject to degradation phenomena with time and usage due to various chemical and mechanical changes to the cell's electrodes. These phenomena gradually lead to a decrease in capacity (capacity fading) and an increase in internal resistance [6]. The cells inside a battery pack are subject to uneven degradation over time due to their asymmetrical placement, uneven temperature distribution, different inter-cell contact resistances, and possible subtle differences in manufacturing quality [7]. As a result, each cell is characterized by a different capacity and internal resistance, which causes current to flow unevenly among the cells, worsening the imbalance of cell degradation rates. Hence, estimating the conditions at the battery pack level in real-time is not easy.

It is possible to measure battery aging in a variety of ways. The most common method of quantifying the capacity loss of a battery is the State-of-Health (SOH), defined as the ratio of the actual capacity of the battery to its nominal capacity:

$$SOH = \frac{C_{actual}}{C_{nominal}} \tag{1}$$

In the context of EVs, a battery pack is said to be at its end-of-life (EOL) when its SOH reaches 80% [8]. This threshold is commonly adopted since, below 80% of its SOH, a LIB incurs a faster, typically more-than-linear degradation [9]. Monitoring SOH is a critical task, since the EOL of a battery pack typically marks the time when it has to be retired. Nowadays, it is possible to perform SOH estimation in a laboratory setting, which is time-consuming. Therefore, developing fast and reliable techniques to quantify the battery pack's SOH would greatly benefit EV drivers.

In recent years, researchers devoted their efforts to designing real-time and on-board procedures, allowing a more manageable and cheaper alternative for estimating SOH. Among the proposed solutions, we notice the growing interest in Machine Learning (ML) solutions, emerging as a powerful tool exploiting historical data for estimation and forecasting issues in many disciplines. However, in the context of monitoring the battery pack's SOH of an EV, implementing ML solutions is still challenging due to the unavailability of open battery data. Nevertheless, much effort has been put into defining EV simulators that generate accurate internal battery pack signals.

The presented work is the outcome of an extended study jointly aiming at tackling the mentioned issue of data unavailability and the implementation of a novel ML approach to estimate the SOH of an EV's battery pack in real-time without time-consuming laboratory experiments, allowing the deployment of the proposed methodology in an actual vehicular environment. In particular, we propose a low computational and chemistry-agnostic data-driven (DD) methodology regarding the definition of ML models to estimate the SOH of the battery pack exclusively trained utilizing synthetic EV battery pack data, generated with an EV simulator proposed in our previous work [10], tackling the lack of such data that are extremely challenging to find in literature. We formulate our problem as a regression task to assess the battery pack's SOH of an EV, throughout its lifespan, using multivariate time series of driving session monitoring data.

We utilize distinct feature extraction methods, with a low computation burden, to speed up the training phase and to reduce the numerosity of the input time series of internal battery signals, which include current, voltage, and State-of-Charge (SOC). The performance of the synthetic-trained ML model is then determined by inferring over test synthetic and real data, proving the benefits of the utilization of training simulated data whenever actual data are not available. Then, we implement Transfer Learning (TL) over the synthetic-trained ML to refine their hyperparameters, thinning the misalignment between real and synthetic data, which struggle to capture all key features of a battery operating in urban traffic.

The high-level overview of the adopted methodology is shown in Figure 2, which is composed by three main phases, the *data acquisition*, *data processing*, *feature extraction*, *regression models & training*, and the final *SOH estimation* phase. However, each phase of the methodology will be thoroughly explained in Section II.

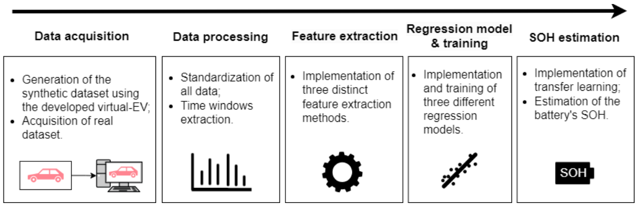
Data acquisition
Data processing
Feature extraction
Regression model & training
SOH estimation
Generation of the synthetic dataset using the developed virtual-EV;
Acquisition of real dataset.
Standardization of all data;
Time windows extraction.
Implementation of three distinct feature extraction methods.
Implementation and training of three different regression models.
Implementation of transfer learning;
Estimation of the battery's SOH.
SOH

FIGURE 2. The high-level pipeline of the proposed methodology.

The best synthetic-trained ML model achieves, over real data, an RMSE and MAE of 5.92% and 5.08%, respectively, demonstrating the benefits of synthetic training data in a scenario where real data are extremely difficult to find. The accuracy of the ML models further improves after TL, achieving the lowest RMSE and MAE, over real data, of 2.56% and 1.97%, respectively. Hence, the proposed methodology demonstrates that synthetic-trained ML models might be a starting point to deploy the presented methodology into a real EV, and such models can be fine-tuned using few real-time battery data, achieving state-of-the-art results for the battery's SOH estimation. Moreover, the low computational burden of the adopted processing steps, as demonstrated in the manuscript, allows real-time estimation of the battery pack's SOH in an actual vehicular scenario.

The rest of this paper is organized as follows. Section II presents an overview of the available solutions in literature to estimate the battery's SOH; Section III describes in detail the adopted methodology, starting from the datasets, feature extraction methods, and ML models. Section IV discusses the experimental estimation results. Finally, Section V provides our concluding remarks and future works.

II. RELATED WORKS

In recent years, research on battery SOH estimation has been extensive and very diverse. SOH estimation methods mainly fall into two categories: Experimental methods and Model-based (MB) methods. We could further expand this taxonomy, discriminating between online and offline methods. The former employs sensor data recorded while driving or charging an EV in real-time; the latter utilizes historical data gathered from an EV, using them for later use [11].

The experimental methods compute relevant quantities directly from experimental battery data and use them to estimate the SOH. Such methods are usually conducted in laboratories given the necessity of special equipment, and they measure internal resistance, playing a central role in the battery's SOH, through either direct or indirect measurements [12].

On the other hand, MB methods estimate the battery's SOH determining the relationship between current, voltage, and capacity over the battery's aging. Moreover, with respect to the experimental methods, the MB methods do not cause the battery destruction during its SOH estimation [12], which is damaging or causing future failures of the battery, compromising the original part [13]. Some of the available MB methods are equivalent circuit models, electrochemical model-based methods, machine learning and deep learning methods.

Equivalent Circuit Models (ECM) [12] are mathematical models of a battery based on electrical components such as resistors, capacitors, and direct current voltage sources, which describe the external characteristics of a battery [14]. ECM consist of mathematical abstractions of a battery: the electrical components that compose it do not have any physical meaning, and the complex physical and chemical dynamics inside a battery are ignored.

Similarly to ECM, Empirical models (EM) [12] are mathematical abstractions of a battery, which try to capture the non-linear relationship between SOH and degradation factors through a single function. It is possible to choose many degradation factors and functional forms. However, like ECM, EM need extensive data in different experimental conditions to be fitted accurately. The ElectroChemical Models (EChM) [15] try to explicitly capture the complex physical and chemical phenomena inside a battery, causing the capacity to fade over time.

DD methods are a family of SOH approaches that leverage large-scale datasets of experimental battery aging data, and whose performances often depend on the size and quality of such datasets [16]. DD methods are gaining increasing interest due to their extreme flexibility and variety. Many DD methods have the advantage of being domain-agnostic, i.e., they do not require domain expertise in electronics and chemistry [17], as they automatically learn the relationship between experimental data and SOH. DD methods based on machine learning and deep learning are becoming the most prominent approaches to SOH estimation [11], [17], [18]. Different feature extraction techniques and regression models are widely explored in the literature [15]. Usually, these methods require an offline training phase. EV monitoring data measured by the battery management system are preprocessed and used for training a machine learning model, providing fast and accurate predictions. Due to these advantages, machine learning methods are particularly suited for real-time, on-board SOH estimation.

Roman et al. [19] proposed a manual feature extraction pipeline that identifies 30 features. They employed recursive feature elimination, based on random forests, to filter out the minor relevant features for the SOH regression task. The resulting dataset trains several regression models, including random forest, deep neural network ensemble, Bayesian ridge, and Gaussian process regression. Many other traditional regression models have been applied to SOH prediction pipelines in the literature. Among these, there are support vector regression [20], linear regression models (ridge, lasso, elastic net) [21], and several variations of Feed-Forward Neural Networks (FFNN) [22]–[24].

Deep learning also offers techniques to develop new performing methodologies or enhance existing ones. For instance, Li et al. [25] proposed a transformer-based learning framework to accurately estimate occupancy of charging stations even when large portions of data are missing, enhancing EV charging monitoring systems. Chemali et al. [26] developed a SOH estimation method based on a Convolutional Neural Network (CNN), which is given raw charging data (voltage, current, and temperature over time) as input. This methodology's advantage is eliminating the manual feature engineering, as a CNN automatically learns how to extract relevant features from raw data.

More complex DL methods have been adapted to the task of predicting the SOH of a battery cell, namely Recurrent Neural Networks (RNN) [27], Long Short-Term Memory (LSTM) [28], Gated Recurrent Units (GRU) [29], independent RNN [30] and many more. However, given the limited computing power of an EV, the complexity of recursive models might be an obstacle for the deployment of real-time solutions [31]. Therefore, simpler models, e.g., FFNN, can be a valuable compromise between complexity and accuracy [31].

Merkle et al. [32] introduced a digital battery twin for a 2014 Volkswagen e-Golf, setting up a data pipeline to predict the battery pack's SOC and SOH in real-time. They acquired a training dataset directly from the onboard diagnostics interface during an actual drive cycle; however, this dataset has not been publicly available. Song et al. [33] trained their SOH estimation methodology on a seemingly large battery pack monitoring dataset from the Shanghai Electric Vehicle Public Data Collecting, Monitoring, and Research Center (SHEVDC). However, researchers need to apply for a membership to access the dataset.

We highlight that almost all of these methods leverage experimental data regarding single battery cells, primarily due to the unavailability of public datasets of EV battery packs' monitoring data. Hence, a solution to generate knowledge that is not directly observable and measurable in the real-world system, and compensate for the lack of publicly available EV monitoring data, lies in the implementation a digital twin of the system under investigation [32].

Therefore, in this work, we propose a low computational and chemistry-agnostic DD methodology to estimate the SOH of the battery pack leveraging internal battery signals. The full and detailed description of the adopted methodology is shown in Figure 3, examining in depth what is reported in Figure 2.

The data acquisition phase represents a substantial portion of the extended study, during which we utilized the developed virtual-EV [10], mimicking a specific real-world EV model, to generate a synthetic dataset which will be the sole source of data employed to train the selected ML models. Subsequently, we acquired actual EV monitoring data from a real-world EV model matching the one of the virtual-EV, and almost covering the whole lifespan of the EV's battery pack, with an SOH ranging from 85 to 99%. We utilized the real data to perform inference of the ML methodology, assessing the generalization capabilities of the synthetic-trained ML models over real data, and finally to implement TL trying to further improve the regression models' performances.

During the data processing phase, we standardized all datasets, a necessary step for the training of the regression models, and then we extracted fixed-length 5-minute-long time series from both synthetic and real datasets. While, in the feature extraction phase, we process the obtained time windows belonging to both synthetic and real datasets using several feature extraction methods. Particularly, we utilized a domain-agnostic unsupervised feature extraction method called MiniRocket, and implemented a novel computationally-inexpensive feature extraction method for which we utilized ordinary least squares and Theil-Sen.

In the regression models & training phase we defined and trained the ML models, which are Ridge Regression, Random Forest, and Feed-Forward Neural Networks. Specifically, we trained each regression model feeding the synthetic dataset generated by each feature extraction method independently. In this way, we could gain insights into the performances for each combination of the feature extraction method and regression model, furnishing a thorough comparison analysis.

Finally, in the last SOH estimation phase, we assessed the estimation performances of the considered synthetic-trained regression models over real data, and then we implemented TL for each analyzed combination of feature extraction method and regression model. Indeed, the synthetic data, although accurate, cannot grasp all key and tiny variations within a battery pack operating in actual traffic. Therefore, we fine-tuned the ML models, solely trained with synthetic data, using 50% of real data, making the synthetic-trained ML models closer to a real scenario.

III. METHODOLOGY

In this section, we provide a thorough description of the proposed methodology, shown in Figure 3. We first briefly describe the developed EV simulator, employed to generate the synthetic dataset. Subsequently, we give details on the simulated and real datasets. Next, we discuss the datasets preprocessing procedures, the chosen feature extraction methods, and their characteristics and limitations. Finally, we report the selected ML methods and their properties, and how we implemented TL.

A. DATA ACQUISITION

Referring to Figure 3, in the data acquisition phase, we acquired the real dataset from the EV model under analysis, and we generated the synthetic dataset utilizing the available EV simulator. Indeed, in our previous work [10], we presented a virtual-EV, developed using MATLAB/Simulink, utilized to generate the synthetic and realistic dataset internal battery signals.

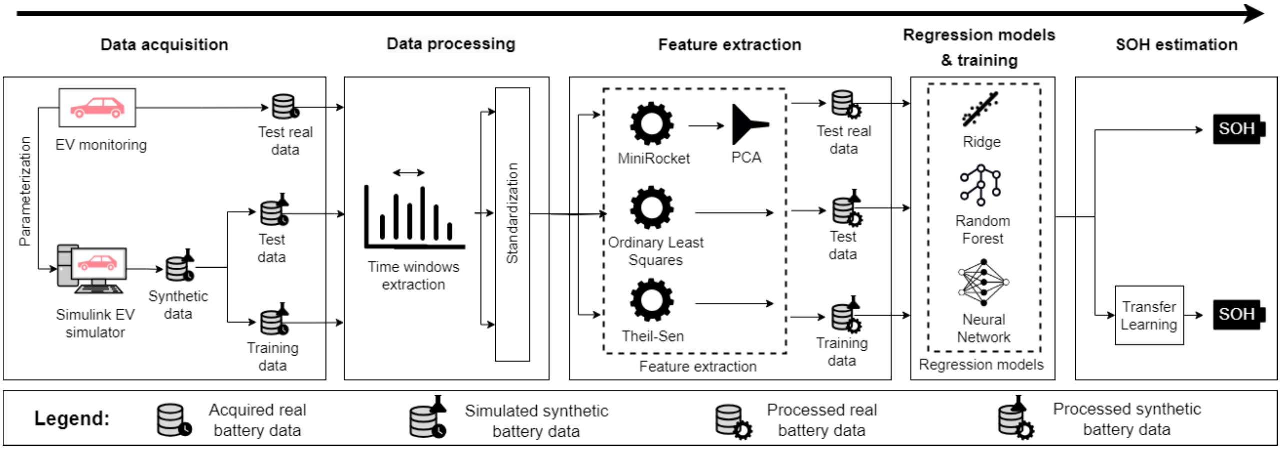

FIGURE 3. The detailed pipeline of the proposed methodology.

The simulator is composed of several mutually dependent subsystems, e.g., electric motor, wheels, braking system, and battery pack, to emulate the main operational mechanisms of a real EV. We tuned a subset of the inner blocks parameters, sourcing from available technical EV data sheets, trying to mimic a target real-world EV model [10] as much as possible. In Table 1, we report a list of the main parameters specific to the mimicked reference EV model and that we utilized to parameterize the main blocks of the virtual-EV.

The choice of designing such a specific EV model is solely due to the availability of real data, presented later in this same section, gathered from the same EV model.

Notable effort has been dedicated to the definition of the battery pack module, being the source of the synthetic outputs of interest. Indeed, to mimic the selected real-world EV, we defined the battery pack module as a multi-cell system composed by 264 cells, organized in 22 modules, each including four in series with three in parallel. The resulting overall nominal energy capacity of the battery is 35.8 kWh. In our previous work [34], we demonstrated that, in a virtual EV, a multi-cell battery pack provides greater accuracy than a simplified model, though with longer simulation times.

To simulate a single driving session, the user must specify a driving cycle, defined as a time series of speeds mimicking the user's driving routine, the outside temperature [°C] (fixed or as a time series of measurements), initial average temperature [°C], initial SOC and SOH of the battery pack. The generated signals are EV's speed [km/h], the battery pack's terminal voltage V [V], current I [A], SOC [%], and average internal temperature T [°C]. The virtual-EV generates such signals with a temporal granularity of 0.1 seconds, although customizable by the user.

The EV simulator embeds the battery's thermal and aging models to properly characterize the battery pack's state, allowing the generation of synthetic and realistic battery signals for the specified temperatures and SOH. In this way, the user can properly characterize the simulated EV model's environmental and internal battery pack's conditions for the driving session of interest. Particularly, the battery pack's aging model implements equation-based age fading, considering the number of discharge cycles over which the specified SOH percentage occurs. Moreover, calendar aging is also included, which leads to an increase of internal resistance and a decrease of capacity.

TABLE 1. Detailed information for battery, motor, and vehicle belonging to the mimicked EV model.

	Description	Value
Battery pack	Type	Lithium-ion
	Capacity [kWh]	35.8
	Voltage [V]	323
	Number of cells	264
	Number of modules	24
	Cell weight [kg]	0.692
Motor	Type	Synchronous AC Permanent Magnet
	Maximum torque [lb-ft]	214
Vehicle	Drag Coefficient	0.27
	Curb Weight [kg]	1567
	Gross Vehicle Weight [kg]	2010
	Frontal area [m^2]	2.19
	Wheels radius [m]	0.26
	Tire rolling resistance	0.01

To provide the reader with a glance of the virtual-EV's performances, in Figure 4, we show the comparison between the signals of current, voltage, SOC, and average internal temperature generated by the EV simulator along with the measured target real battery signals, for a sample driving session.

Referring to Figure 4, the high variability in both the simulated and target signals is primarily due to the data's sampling frequency of 0.1 seconds, which captures small fluctuations in the monitored signals throughout the simulation. While we could have reduced the apparent variability by resampling the signals, we chose to present the data in its original form to show the reader the simulator's actual outputs.

We report in Table 2 the deviation between simulated and real battery data for the selected driving session, and the virtual-EV achieves an RMSE of 38.43 A and 2.28 V over current and voltage, in this order. Moreover, the R^2 remains positive for all the simulated signals, reaching 1.00 and 0.93 for SOC and voltage, respectively.

We utilized the developed EV simulator to generate a

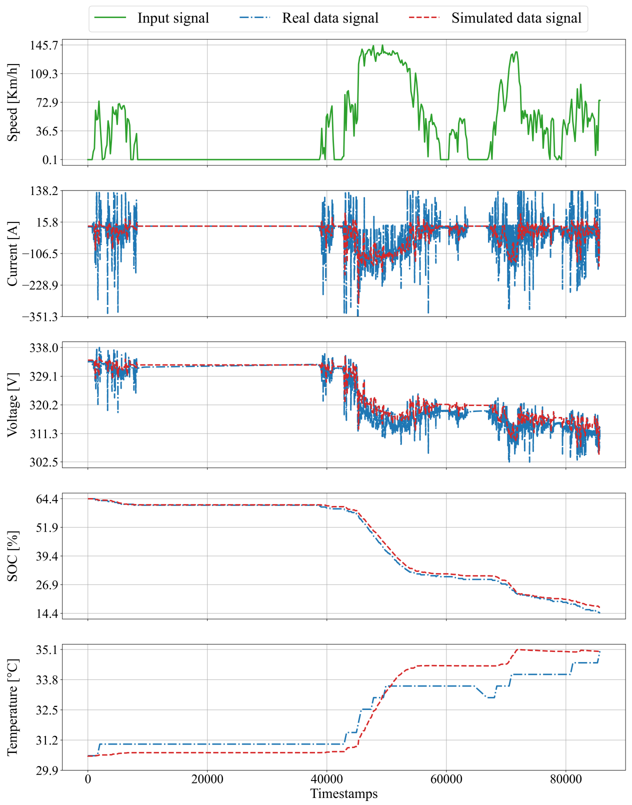

FIGURE 4. The comparison between the output signals generated by the EV simulator, and the real data signal from the corresponding real-world EV.

TABLE 2. The deviation between the output signal generated by the EV simulator and the target real signals.

Battery signal	RMSE	R^2
Current	38.43 A	0.28
Voltage	2.28 V	0.93
SOC	1.11 %	1.00
Avg. internal temp.	0.65 °C	0.77

synthetic EV battery pack data dataset, aiming at acquiring a sufficiently representative monitoring dataset in a wide range of driving conditions. We populated the synthetic dataset by executing individual simulations, given a unique combination of inputs, which include: the driving cycle, the initial battery pack's SOH, SOC, internal temperature, and the environmental temperature.

We selected 9 input standard drive cycles [35] to span different driving conditions, including city, urban and highway driving. The drive cycles are categorized based on the maximum speed observed, that is, 50 Km/h, 90 Km/h, and above 90 Km/h for city, urban, and highway, respectively. The initial battery pack's SOH remains the same throughout the simulation, and it is set, through the battery pack block's aging model, to an integer value ranging between 100% and 80% (i.e. battery's EOL [8]). In this way, we try to retrieve synthetic data, covering the whole lifespan of the EV battery pack.

The initial SOC always starts from 100%, while the starting environmental and internal temperatures are randomly drawn from two uniform distributions with bounds [-5,30]°C and [T_env, T_env+5]°C, respectively, where T_env is the environmental temperature. In Table 3 we summarize the set of simulation inputs and their values.

Once all the simulation inputs have been defined, the simulation starts, and the driving cycle is looped until the battery pack's SOC gets below 10%. The simulation stops when such a condition is met, and the output signals are collected.

We executed a number of simulations matching the total number of combinations between the 9 driving cycles and the 21 integer SOH values. In this way, the synthetic dataset will include 189 simulated driving sessions covering a vast range of driving conditions.

For each simulated driving sessions, and for the sake of our study, we solely selected signals of current, voltage, and SOC, characterized by a temporal granularity of 0.1 seconds.

Apart from the synthetic dataset generation, we acquired a real EV dataset from a private EV fleet management company relative to the same real-world EV model, mimicked by the virtual-EV. The real-world EV is equipped with a battery pack characterized by 264 lithium-ion cells, organized in 22 modules, each including four in series with three in parallel, with overall nominal energy capacity is 35.8 kWh. The same battery pack's architecture defined in the virtual-EV.

The general overview over the real dataset is provided in Table 4, in which we report for each value of SOH, the number of available driving sessions' data, and their total temporal converge in seconds. The real dataset span several months of data acquisition, and it includes monitoring data signals collected by the battery management system of distinct EVs, all belonging to the same considered real-world EV model. The collected real data signals include information related to EV's speed, battery pack's current, voltage, SOC, average internal temperature, and environmental temperature. Each of such signals is sampled using different sampling rates [10]. For our methodology, We only extrapolated data signals of voltage, current, and SOC.

The real driving session is accompanied by the battery's SOH value, obtained discovering the nominal and actual capacity of the battery pack through a proprietary and commercial test, whose methods and conditions remain undisclosed to the public. The estimated SOH values have a maximum error of ±3%, as claimed by the private EV fleet management company. The estimated SOH values during the monitored periods exhibit sufficient variability, including SOH values ranging from 99% to 85%, almost covering the whole lifespan of the EV's battery pack.

B. DATA PREPROCESSING

Before using the feature extraction methods, as shown in Figure 3, we performed a few preprocessing steps on synthetic and real datasets. As previously mentioned in Section III-A, each measured real signal is sampled with a different frequency. Thus, we independently resampled all signals in the

TABLE 3. The list of inputs utilized to characterize all executed simulations.

Input	Value/characteristics
Driving cycle	5 urban, 2 city, 2 highway
SOH	[100, 99, 98, ..., 82, 81, 80]%
SOC	100%
Environmental temp. [T_env]	U[-5, 30]°C
Avg. internal temp.	U[T_env, T_env+5]°C

TABLE 4. Real dataset composition, grouped by the SOH values, with the relative number of available drive sessions, and total temporal coverage in seconds.

SOH [%]	# available driving sessions	Total time coverage [s]
85	29	17303
86	25	17563
87	27	17351
88	31	17322
92	2	8499
93	5	13930
94	2	15003
95	8	2853
96	1	6106
99	1	16961

Algorithm 1 Time windows extraction

```
time_windows_dataset $\leftarrow$ empty list;
for driving session $\in$ dataset do
  duration $\leftarrow$ duration of the current driving session;
  if random_starting_point then
    start $\leftarrow \mathcal{U}(0,$length$)$;
  else
    start $\leftarrow$ 0;
  end if
  end $\leftarrow$start+length;
  while end $\leq$ duration do
    time_window $\leftarrow$[start, end];
    if SOC values in time_window $\in$
        [max_SOC, min_SOC] then
      append data of time_window to
        time_window_dataset;
    end if
    start $\leftarrow$start+slide;
    end $\leftarrow$end+slide;
  end while
end for
return time_window_dataset;
```

real datasets through linear interpolation with the same sampling rates specific of the synthetic dataset, that is 0.1 seconds. Such a resampling is necessary to have the synthetic dataset and the real one sharing a common time base for all included signals, allowing a correct implementation of the proposed methodology over the synthetic and real data.

Then, as shown in Figure 3 , we split all simulated and real driving sessions signals into several non-overlapping 5-minute-long time windows, which inherit the corresponding SOH value, the target label, associated with that session. Each extracted time window includes a time series of current, voltage, and SOC.

The complete time window extraction algorithm is reported in the Algorithm 1,
where dataset is the dataset from which to extract time windows, either synthetic or real; length is the duration of the time windows in [s], by default we adopted a length of 300 s; slide is the time between the starting points of two consecutive time windows in [s], and we choose a value for slide that would generate non-overlapping time windows; freq is the sampling rate of the measurements [Hz], in our case 5 Hz; random_starting_point is a Boolean to choose the starting point of the first time window at random in the interval [0, length], by default it is set to True; min_SOC and max_SOC are set to 0 and 100, respectively, to make sure time windows do not contain SOC values out of the range [0, 100]%.

The time window extraction, described in the Algorithm 1, has been accomplished independently for all driving sessions belonging to the synthetic dataset and real one. Each extracted time window can be viewed as a multivariate time series with C = 3 channels (V, I, SOC) and length T = 1500 (i.e., 300 s × 5 data/s).

To fulfill the requirements of the proposed DD approach, we randomly split the synthetic time windows dataset into a synthetic-training-set and a synthetic-test-set, including 80% and 20% of the total time windows, respectively. The time windows are assigned to the synthetic-training-set and synthetic-test-set in a stratified fashion so that the distribution of different SOH values, in each set, is the same as in the original dataset. Table 5 reports the total number of time windows extracted from the synthetic and real datasets. We also note in Table 5 the cardinality of the synthetic-training-set, synthetic-test-set and real-test-set that will be further processed during the feature extraction phase and, subsequently, employed to train the regression models.

TABLE 5. The total time windows extracted using the Algorithm 1, from synthetic and real datasets, and their relative partitioning into sets for the regression models.

	Synthetic dataset	Real dataset
Training set	7210	-
Test set	2509	440
Total	9013	440

When training an ML model, it is common to standardize the data, as the scale of the training data has a relevant effect on the scale of the learned parameters and, thus, on the quality of the final solution. Having input variables scaled to zero mean and unit standard deviation implies that each feature will be deemed equally relevant in the learning process. It can be shown that this improves the convergence rate of training a neural network [36]. The values $s_1, ..., s_T$ of each signal are transformed using Equation 2:

$$z_i = \frac{s_i - \bar{\mathbf{s}}^{(train)}}{\hat{\sigma}^{(train)}} \tag{2}$$

where s_i is the input value of the signal, $\bar{\mathbf{s}}^{(train)}$ and $\hat{\sigma}^{(train)}$ are the sample mean and standard deviation of the considered signal in the training set, respectively, and z_i is the standardized output. We individually compute the mean and standard deviation for current, voltage, and SOC across all time windows belonging to the synthetic-training-set. We then, as depicted in Figure 3, standardized all available time windows within synthetic-training-set, synthetic-test-set, and real-test-set, using the discovered mean and standard deviation. Computing the mean and standard deviation concerning only the synthetic-training-set avoids data leakage from both synthetic-test-set and real-test-set, which must be kept unseen until the end of the training phase.

C. FEATURE EXTRACTION

In many experimental settings, the length of the recorded time series might be too long to train a regression model directly on raw time series data. Indeed, each sample is regarded as a feature, and many machine learning algorithms do not scale well with the number of features of a dataset [37]. Moreover, when dealing with continuous signals, data taken at contiguous timestamps are typically correlated in some way, meaning that some feature selection is desirable to reduce such redundancy. Therefore, it is often preferable to find a static feature-based representation of the time series [37].

Thus, we may want to find a way to transform a generic $N \times C \times T$ time series dataset into a static $N \times M$ features dataset where, in our case, N, C, and T are the number of extracted time windows, available channels, and duration of the time series, respectively. While, M is the number of new input attributes extracted using feature extraction methods. Then, the new dataset with dimensions $N \times M$ is employed to train the regression models. These approaches to time series extrinsic regression problems are called *feature-based* regression algorithms [38].

In the literature, we could identify two distinct time series feature extraction techniques: manual and automatic. With the former, features are hand-crafted by a domain expert to capture relevant information about a specific problem, but they are often poorly generalizable to other regression problems. The latter uses algorithms to extract features in a supervised or unsupervised fashion automatically and without the requirement of data expertise.

As previously highlighted, in this work we utilized three different automatic feature extraction approaches to distill meaningful features from the extracted 5-minute-long time windows. Particularly, we employed MiniRocket and a novel extraction method leveraging linear regression in the I-V-SOC space through ordinary least squares and Theil-Sen, independently.

The proposed feature extraction methods are individually applied to the synthetic-training-set, synthetic-test-set, and real-test-set, as reported in Table 5. We then use the resulting new datasets to train and test the regression models to estimate the target battery's SOH.

Dempster et al. [39] proposed MiniRocket (MR), a new time series regression procedure that achieves state-of-the-art accuracy in time series regression with a low computational expense.

Although MR is considered state-of-the-art for time series extrinsic regression problems, the utilized transform operation is computationally expensive. Almost 10,000 convolutions per time window needs to be computed. Moreover, it produces many features (9996) from each input time series. The extremely high number of features may cause machine learning models to not scale well with high-dimensional datasets due to sparseness in data, a phenomenon known as the curse of dimensionality. Therefore, we used Principal Component Analysis (PCA) [40] to perform a dimensionality reduction of the features extracted by MR to overcome the discussed issue.

PCA [40] is a method to perform a certain change of basis on the data, in which the axes of the new coordinate system are the set of orthonormal eigenvectors commonly known as *principal components* of the dataset. PCA is typically exploited as an unsupervised dimensionality reduction technique, since it maximizes the data's variance when mapped down to lower dimensional spaces.

PCA is applied to MR-transformed synthetic and real datasets. By visualizing the scree plot for the synthetic-training-set in Figure 5, we observe that with just 32 features out of the original 9996, we can explain over 94% of the total variance in the data. The first 32 principal components of the synthetic-training-set are then used for all PCA transforms. Moreover, since PCA requires standardized data, time series data transformed through MR is standardized before the PCA transform with sample mean and standard deviation defined for the synthetic-training-set. Furthermore, the PCA transform does not necessarily output standardized data, so standardization is also applied after every PCA transform.

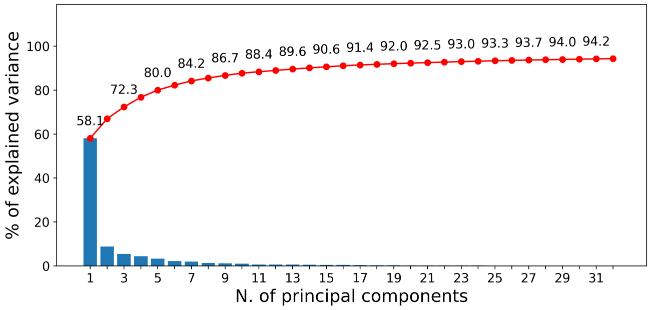

FIGURE 5. **The results of PCA over the MR-transformed dataset for the synthetic dataset highlight the 32 features that explain over 94% of the total variance in the data produced by MR.**

Thus, we can reduce the 9996 features to 32 for each time window after applying PCA over the features extracted by MR. The processed synthetic-training-set will be used as input to train the regression models, while the synthetic-test-set and real-test-set will be utilized to perform inference over the synthetic-trained models estimating the battery's SOH.

As already highlighted, MR computes almost 10,000 convolutions between the time window of monitoring data given as input and the MR kernels. However, embedding a SOH estimation method on a real EV requires the procedure to have a low memory footprint and – most critically – a little computational cost due to the need to estimate the SOH in real-time. Therefore, we introduce in this section a new feature extraction method that is domain-specific, computationally cheap, and low dimensional.

The EV field data typically consists of time series of V, I, and SOC data (among others), taken during a driving session of the vehicle. We will refer to the tuple of data $(V(t_i), I(t_i), SOC(t_i))$ for a given timestamp t_i as an *operating point* of the EV. The operating points of a specific driving session appear to lie on a hyperplane in the V-I-SOC space.

We show in Figure 6 the cloud of operating points for two synthetic and two real time windows, drawn from the respective datasets, characterized by a SOH of 99% and 94%, along with their interpolating planes computed using OLS. Observing Figure 6, it is clear that such an observation is accurate concerning both synthetic and real data, although with certain degree of approximation.

Interestingly, the hyperplane defined by the operating points seems to be discriminative of the age of the battery during that driving session. Different battery SOH values define different hyperplanes in the V-I-SOC space.

Therefore, we could consider the problem of predicting the SOH of a battery pack as equivalent to finding the hyperplane, which approximates all the possible operating points of an EV at a specific point of its operating lifetime. To make this task compatible with the requirements of a real-time SOH estimation method, we need to find a hyperplane from the knowledge of only a relatively short time window of operating points (i.e., a small sample of consecutive operating points) instead of the whole statistical population of operating points. The equation of an affine hyperplane in a three-dimensional space is defined in Equation 3 as follows:

$$z = ax + by + c \tag{3}$$

where the three parameters $a, b, c \in \mathbb{R}$ uniquely identify a 3D hyperplane. Hence, we can represent each time window, i.e., a distribution of operating points, with just three features, namely the a, b, and c defining the hyperplane that better fits those operating points. Estimating the hyperplane parameters that pass near all the points in a given set can be stated as a linear regression problem [41]. We explored two such methods for designing our novel feature extractor: *Ordinary Least Squares estimation* (OLS) and *Theil-Sen estimation* (TS).

OLS estimation is the most common solver for a linear regression problem. It consists in finding the parameters $\boldsymbol{\beta}$ minimizing the residual sum of squares defined in Equation 4:

$$RSS(\beta) = \sum_{n=1}^{N} (y_i - f(\mathbf{x_i}; \boldsymbol{\beta}))^2 \tag{4}$$

where $\mathbf{x_i}$ is the vector of input samples, y_i the output variable, N is the number of input samples, and $\boldsymbol{\beta}$ is the set of parameters that minimizes the residual sum of squares. The inputs can be represented as a matrix X of dimensions $N \times M + 1$, where N is the number of samples and M are the features of each input, with the dataset samples $\mathbf{x_1}, ..., \mathbf{x_N}$ as rows, with a constant $x_{i,0} = 1$. It is possible to define the Gram matrix as $X^T X$. If the Gram matrix is positive definite, then the OLS regression problem has a unique solution, reported in Equation 5:

$$\hat{\boldsymbol{\beta}}^{OLS} = (X^T X)^{-1} X^T \mathbf{y} \tag{5}$$

where, $\mathbf{y}$ is the vector of outputs and $\hat{\boldsymbol{\beta}}^{OLS}$ is the set of parameters of the OLS solution.

The OLS generally yields a reasonable estimate for the parameters $\boldsymbol{\beta}$ under the assumption that $\mathbb{E}(y|\mathbf{x})$ is a linear function with good approximation. However, OLS estimates are susceptible to outliers since it seeks to minimize all residuals equally and penalize large residuals more [42].

In the literature, more robust regression methods [42] are designed to limit the effect of outliers on regression estimates. Indeed, besides OLS, we also analyzed the TS estimation method, a robust alternative to the OLS method that reduces outliers' contribution to the error loss, thereby limiting their impact on regression estimates. TS estimation was first introduced by Henri Theil [43] and Pranab K. Sen [44] as a method for fitting simple linear models with just one explanatory variable, $f(x; m, b) = mx + b$; recently, it has been generalized to multiple linear regression by Dang et al. [45].

As originally defined, the TS estimate for the slope parameter m, of the simple linear model on a set of two-dimensional points $(x_1, y_1), ..., (x_N, y_N)$, is the median of the slopes $m_{i,j} = (y_j - y_i)/(x_j - x_i)$, determined by all pairs of sample points. The intercept $\hat{\boldsymbol{b}}^{TS}$ can be estimated in several ways; one of them is setting b as the median of the values $b_i = y_i - \hat{\boldsymbol{m}}^{TS} x_i$. It is clear that computing $\hat{\boldsymbol{m}}^{TS}$ and $\hat{\boldsymbol{b}}^{TS}$ with this procedure becomes very expensive as the dataset size N increases. For instance, having N points, $\binom{N}{2}$ slopes must be computed one for each pair of points.

One possible solution is sampling at random a subpopulation $S \ll \binom{N}{2}$ of pairs of points and performing TS estimation on this reduced training set, effectively computing only S slopes. In this way, we can reduce the computational cost, but the estimation may become very sensitive to the particular random choice of the subpopulation, especially if S is very small. In this work, S has been set to 10,000 for all experiments.

We believe that MiniRocket, thanks to its accuracy and speed at processing the input samples, combined with PCA, allows the reduction of the input features to the most infor-

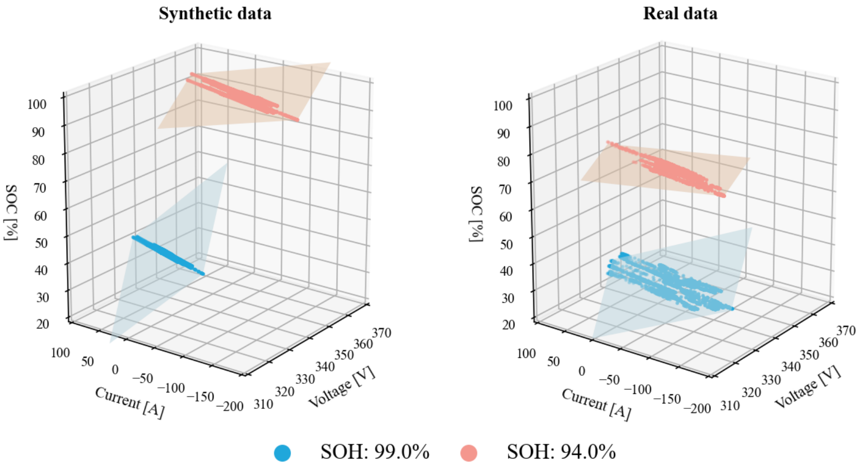

FIGURE 6. The distribution of operating points for two synthetic and two real time windows in the SOC-I-V space, generated by a battery pack characterized by an SOH of 99% and 94%, and their interpolating plane obtained using OLS.

mative ones, improving the estimation precision of the ML models.

Conversely, the spatial distribution of the time windows' operating points in the V-I-SOC space, shown in Figure 6, leads us to believe that the application of linear regression in the same space can condense extended multi-dimensional time windows to just few features, strictly related to the initial signals in the time windows.

D. REGRESSION MODELS & TRAINING

Using the feature extraction methods, reported in the regression models & training block in Figure 3, we transformed the time series belonging to the synthetic-training-set, synthetic-test-set, and real-test-set, as reported in Table 5, to their static feature-based representations. The dimensions of the new datasets provided by the feature extraction methods are reported in Table 6. Subsequently, we chose and trained regression models to predict the battery pack's SOH given the transformed time series of its internal signals of I, V, and SOC, as reported in Table 6. Many regression models exist, both linear and non-linear. In this work, we consider three well-known regression models, namely, *Ridge Regression* (RR), *Random Forest* (RF), *Feed-Forward Neural Network* (FFNN).

TABLE 6. Dimensions of the synthetic-training-set, synthetic-test-set, and real-test-set after the application of the feature extraction methods.

	Synthetic dataset		Real dataset
Feature extraction method	Training set	Test set	Test set
MR+PCA	(7210, 32)	(1803, 32)	(440 32)
OLS	(7210, 3)	(1803, 3)	(440 3)
TS	(7210, 3)	(1803, 3)	(440 3)

Considering a linear regression model, as defined in Equation 4, the Gram matrix might sometimes present a singularity, which means that it includes highly correlated independent variables in the data. In such cases, the OLS estimation problem becomes ill-posed, leading to severe numerical issues. Hence, to mitigate this problem, we may impose a size constraint on the coefficients by introducing an L2 regularization term in the residual sum of squares defined in Equation 4, which becomes,

$$RSS_{L2}(\boldsymbol{\beta};\lambda)=\sum_{n=1}^{N}(y_i-f(\mathbf{x_i};\boldsymbol{\beta}))^2+\lambda\sum_{n=1}^{M}{\beta_i}^2 \qquad (6)$$

where $\lambda \in \mathbb{R}$ is a user-defined penalty factor. Equation 6 defines the linear regression problem, called ridge regression problem, which can be solved by minimizing the L2-regularized RSS. *Ridge regression* has the following closed-form solution,

$$\hat{\boldsymbol{\beta}}^{ridge}=\left(X^TX+\lambda I_0\right)^{-1}X^T\mathbf{y} \qquad (7)$$

where I_0 is the $(M+1)x(M+1)$ identity matrix with 0 as the first diagonal entry. Therefore, adding a positive constant to the diagonal of X^TX before inversion makes the resulting matrix non-singular, even when X^TX is non-singular, thus making the linear regression problem well-conditioned. Notice also that due to the addition of λI_0, ridge estimates are not equivariant under scaling of the inputs (as opposed to OLS), and so one normally standardizes the dataset before solving the ridge regression problem.

Whenever the number of samples N and features M is high, the resulting matrix X becomes very large. Hence, the matrices X^TX and $X^TX+\lambda I_0$ are expensive to compute. Therefore, we solve the ridge regression problem using the Stochastic Gradient Descent (SGD) [46].

Random forest [41], [47] is an ensemble learning method for classification and regression tasks, which performs prediction through majority voting (classification) or averaging (regression) the predictions of an ensemble of decision trees.

A *Feed-Forward Neural Network* is a non-linear model which can be used for regression tasks. The central idea is to extract linear combinations of the input variables as derived features, and then model the target as a non-linear function of these features. The building block of a neural network is the neuron, which computes the linear combination of an input vector $\boldsymbol{x}$ with weights $w_1, ..., w_M$ and bias b. The neuron's output is then used as the argument of a non-linear function, called the activation function. The non-linear inner structure of the neurons makes the FFNN a non-linear model.

The training phase of a FFNN consists in estimating a set of learnable parameters $\boldsymbol{\theta}$, namely its weights and biases, in such a way that the predicted values for the target variable associated with the training samples $\boldsymbol{x}_1, ..., \boldsymbol{x}_N$ are close to the ground-truth targets $y_1, ..., y_N$. The function that quantifies the deviation between training samples and targets is called cost function $\mathcal{L}$. During the training phase, the FFNN's parameters are usually updated iteratively via SGD. The SGD requires the gradient of the cost function with respect to each learnable parameter to perform an update. An efficient algorithm to compute analytically, and recursively, the partial derivatives $\partial\mathcal{L}/\partial\theta$ is called backpropagation [48].

We discuss now the adopted procedures to train the selected RR, RF, and FFNN regression models to estimate the battery pack's SOH. As a result of the feature extraction phase, we obtain the new synthetic-training-set, synthetic-test-set, and the real-test-set, as reported in Table 6.

We trained the regression models utilizing the new ynthetic-training-sets, independently. Subsequently, we assessed the performances of the regression models over the corresponding synthetic and real test sets. In this way, we can determine the best-performing combination of the feature extraction method and regression model separately. Since we considered three regression models and three feature extraction methods, we trained three instances of each model, one for each feature extraction method.

We utilized K-fold cross-validation to tune the hyperparameters of the RR and RF models. The K-fold cross-validation randomly splits the training set into K subsets called folds. One fold is held out as a validation set, while the remaining $K - 1$ folds are used for training the model. Each prediction pipeline is fitted and trained on the $K - 1$ training folds and tested on the remaining validation fold, on which one or more performance metrics are evaluated. Such a procedure is repeated K times to use each fold as a validation set; therefore, a total of K regression models are trained. The resulting performance metrics are averaged over the K rounds to estimate the model's predictive performance on unseen data.

Typically, a grid search is performed over different hyperparameters through cross-validation. The hyperparameter combination achieving the highest cross-validation score is considered the optimal one. The regression procedure with the optimal hyperparameters is refit on the whole training set, thus obtaining the final regression model. Consequently, we performed a grid search with 5-fold cross-validation to tune some hyperparameters of RR and RF models. While the FFNN hyperparameters were tuned using a "trial and error" approach due to the higher computational cost of training such models. Hence, for the FFNN, we selected 10% of the synthetic-training-set to validate its performances at the end of each epoch. Moreover, the FFNN is trained via SGD with a learning rate divided by 10 whenever the training loss does not decrease enough for several consecutive epochs.

We report in Table 7 the discovered optimal hyperparameters for each of the nine combinations of feature extraction method and regression model. In particular, for RR α is the constant that multiplies the regularization term, η_0 is the value of the initial learning rate, and p is the exponent for the inverse scaling learning rate. While, for the FFNN, the neurons per layer refer to the number of neurons per hidden layer, α is the learning rate, η_0 is the rate at which, in the SGD, the learning rate is reduced whenever the training loss does not decrease enough for P consecutive epochs, and μ is the SGD's momentum.

We run the experiments over a machine with the following specifications: Intel Xeon W-2155 CPU @ 3.30 GHz (10 cores, 20 threads), 64 GB RAM, NVIDIA Quadro P4000 GPU (8 GB GDDR5).

E. SOH ESTIMATION

At this point, following the procedures discussed in Section III-D, we independently trained the selected ML models using the synthetic-training-set, each generated utilizing the methods presented in Section III-B. Then, during the SOH estimation phase, as reported in Figure 3, we assess the performances of the synthetic-trained ML models over the synthetic-test-set and real-test-set.

Comparing the estimation results of the ML models trained using the distinct processed synthetic datasets, as reported in Table 6, we intend to discover the best combination of feature extraction method and ML model to estimate the battery pack's SOH.

As depicted in Figure 3, after assessing the performances of the synthetic-trained ML models for the distinct feature extraction methods, we applied Transfer Learning (TL) over the same models, trying to improve their estimation results. We utilized a fraction of the corresponding processed real dataset, i.e., 50%, to fine-tune the hyperparameters of the ML models. Specifically, we randomly selected, roughly, 50% of real samples for each SOH label to populate the dataset utilized for TL, preserving the same distribution of SOH labels in the original real dataset. In this way, it is possible to fine-tune the ML models preventing them to, possibly, overfit over a specific SOH value.

More in detail, the parameters of the synthetic-trained RR model are updated through a new fitting procedure utilizing the selected fraction of the specific real dataset. While, for the RF, starting from the synthetic-trained model, we added 200 new decision trees fitted using the 50% of the specific processed real dataset. Finally, the FFNN undergoes a new training phase, during which we solely update the parameters of the neurons belonging to the output layer.

TABLE 7. Hyperparameters chosen for each regression model and training set.

	Ridge regression			**Random forest**		**Feed-Foreward Neural Network**					
	α	η_0	p	max. tree depth	n. trees	neurons per layer	max epochs	α	η_0	μ	P
MR+PCA	0.0278	0.001	0.25	30	1000	(120, 60, 40)	300	0.0001	0.1	0.9	5
OLS	0.01	0.0464	0.25	50	500	(40, 20)	300	0.0001	0.1	0.9	5
TS	0.01	0.0464	0.25	unlimited	500	(40, 20)	300	0.0001	0.1	0.9	5

We believe that, given the issue of data unavailability, we can train the backbone ML models with the realistic synthetic data, constituting a solid foundation to the regression models. Then, we fine-tune the hyperparameters of the same synthetic-trained ML models with a fraction of real data, making them closer to a real scenario.

The resulting fine-tuned ML models will embed hyperparameters discovered using the synthetic data, and then improved with the real data. We assessed the precision of the fine-tuned ML models performing inference on the remaining 50% portion of real data not utilized during the TL procedure.

In the next Section IV we discuss and compare the estimation results of the ML models, trained with the synthetic-training-set, over the synthetic-test-set and real-test-set. Afterward, we present the improvements introduced by TL, performing inference for the fine-tuned ML models over the remaining 50% of the real-test-set.

IV. RESULTS

This section presents the estimation performances achieved by the different regression models and feature extraction methods. Firstly, we evaluated the results of the proposed methodology in terms of Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE). The former gives information on the prediction error that is committed on average, whereas the latter measures the average difference between simulated and the actual values. The mathematical formulation of the MAE and RMSE is the following,

$$MAE = \frac{1}{N}\sum_{n=1}^{N}(\hat{y}_i - y_i)^2 \tag{8}$$

$$RMSE = \sqrt{\frac{\sum_{n=1}^{N}(\hat{y}_i - y_i)^2}{N}} \tag{9}$$

where, y_i target ground-truth sample, $\hat{y}_i$ is the estimation generated by the considered regression model, and N is the cardinality of the test set.

Using the MAE and RMSE, we have firstly evaluated the synthetic-trained regression models on the held-out synthetic-test-set; then, we analyze the performances of the same synthetic-trained regression models over the real-test-set. In Table 8, we report the obtained estimation results for each combination of feature extraction method and ML model, over the synthetic-test-set and real-test-set. Observing Table 8, we can determine that the FFNN, trained with features extracted with MR+PCA, substantially outperforms, over the synthetic-test-set, all the rival regression models and feature extraction methods. Indeed, the combination of FFNN and MR+PCA achieves over the synthetic-test-set an RMSE and MAE of 0.37% and 0.27%, in that order.

The RR models are the least accurate regression models, which achieve, paired with MR+PCA, the lowest RMSE and MAE of 3.14% and 2.51%, respectively, over the synthetic-test-set. The RR, fitted with OLS and TS data, struggle to reach an RMSE lower than 6.00%. The discrepancy between results obtained by the RR and FFNN can be justified addressing the relatively lower design complexity of the RR compared to the FFNN model. On the other hand, the RF models perform well over the synthetic-test-set, although not as much as the FFNN.

Now, we analyze the performances of the synthetic-trained regression model over the real-test-set. In contraposition with the results examined for the synthetic data, the best-performing regression model over real data is the RF. In fact, the RF combined with MR+PCA reaches, over the real-test-set, an RMSE of 5.08%. While, the FFNN, that obtained lower errors over the synthetic-test-set, performs quite poorly over real data, reaching the minimum RMSE of 4.59%.

Moreover, the RF, compared with the other ML models, achieves more consistent result, with RMSE values below 10.00% and without exploding errors like FFNN and RR. Indeed, due to low generalization capabilities, the synthetic-trained FFNN achieves RMSE and MAE errors out of scale over real data, making it too unreliable compared to the RF model.

Overall, OLS is constantly outperformed by the other feature extraction methods. Indeed, the OLS, as discussed in Section III-C, is strongly affected by the presence of outliers and generally, as shown in Figure 6, time windows exhibit irregular points distribution. Such a susceptibility to outliers, inevitably affects the performances of the OLS method.

Moreover, in Table 8, we report the average feature extraction time for each feature extraction since the required computational time is critical for a real-time SOH estimation procedure. We omit the time for predicting the SOH value since it happens to be negligible, e.g., $< \ 10^{-3}$ seconds per time window, for all regression models. Therefore, we conclude that every possible combination of a single feature extraction method with a single regression model defines a SOH estimation procedure that fulfills the real-time requirement.

Finally, in Table 8, we also compare the performances achieved by the synthetic-trained model and the same models after TL. The RF remains the best-performing model over real data, both before and after TL. Moreover, the MR+PCA proves to be, still, the best feature extraction method compared to the others. Thus, using the combination of RF and MR+PCA and after the TL fine-tuning of the model, we can reach the lowest RMSE and MAE of 2.56% and 1.97% over the reduced real-test-set. Nevertheless, all regression models benefit from fine-tuning, reducing the deviation between estimations and ground truth by a substantial margin. All models, after TL and combined with the proper feature extraction methods, obtain an RMSE and MAE generally lower than 5%.

We utilized 50% of the real dataset to accomplish the models' fine-tuning through TL, which, intuitively, leads to believe that larger datasets would further improve the accuracy of the regression models. However, we believe that synthetic data can become the baseline foundation to accomplish preliminary training of the regression models for estimating the battery pack's SOH, given the issue of data unavailability. Then, with a few real data, we can fine-tune the regression models, reaching good results in the monitoring of battery aging.

In Table 9 we report the computational times required by each feature extraction method and to infer the SOH estimation from the trained ML models, per sample, i.e., 5-minute-long time window. Analyzing the best performing combination of feature extraction method and ML model, that is MR+PCA and RF, the aggregate computational time does not exceed 0.20 seconds, which allows us to fulfill the real-time requirements of the proposed methodology. Nonetheless, given that a substantial variation of SOH becomes considerable over time, the user might instantiate the methodology once every fixed amount of time, possibly absorbing the latency, although very small.

V. CONCLUSIONS AND FUTURE WORKS

In this work, we proposed a novel DD and chemistry-agnostic approach to estimate the battery pack's SOH. Due to a lack of publicly available datasets of EV field data, we defined a MATLAB/Simulink simulator, mimicking a specific real-world EV model [10]. We utilized the EV simulator to simulate many driving sessions in different conditions, collecting synthetically-generated EV field data.

We extracted many 5-minute-long time windows from each artificial driving session, and we processed them using three feature extraction methods: MR+PCA, OLS, and TS. We applied each feature extraction method over the synthetic and the real datasets independently, obtaining a static representation of time series data. We finally used the processed synthetic data to train three different regression models: RR, RF, and FFNN.

In general, the synthetic-trained RF combined with MR+PCA and TS achieve acceptable results over real data. The application of TL using a portion of real data proved to be effective. Indeed, the fine-tuned RF model, combined with MR+PCA, reaches over real data the lowest RMSE and MAE of 2.56% and 1.97%, respectively.

As highlighted in Table 9, the processing and inference procedures proved to be very fast, with their combined completion times way below one second. Therefore, given the offline training of the ML models, the methodology becomes suitable for real-time monitoring of SOH.

For future works, we intend to develop a co-simulation platform to execute new simulation sourcing from actual routes, including information about road's slope. This would allow us to define a more vast and realistic dataset from a whole EV fleet. Moreover, we will investigate the application of scientific ML reducing regression estimates violating physical relationships.

REFERENCES

[1] IEA, "Greenhouse gas emissions from energy data explorer," 2021. [Online]. Available: https://www.iea.org/data-and-statistics/data-tools/greenhouse-gas-emissions-from-energy-data-explorer

[2] IEA, "Transport sector co2 emissions by mode in the sustainable development scenario, 2000-2030," 2021, licence: CC BY 4.0. [Online]. Available: https://www.iea.org/data-and-statistics/charts/transport-sector-co2-emissions-by-mode-in-the-sustainable-development-scenario-2000-2030

[3] EEA, Decarbonising road transport: the role of vehicles, fuels and transport demand. Publications Office of the European Union, 2022.

[4] L. H. Saw, Y. Ye, and A. A. Tay, "Integration issues of lithium-ion battery into electric vehicles battery pack," Journal of Cleaner Production, vol. 113, pp. 1032–1045, 2016.

[5] M. Zwicker, M. Moghadam, W. Zhang, and C. Nielsen, "Automotive battery pack manufacturing – a review of battery to tab joining," Journal of Advanced Joining Processes, vol. 1, p. 100017, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2666330920300157

[6] J. S. Edge, S. O'Kane, R. Prosser, N. D. Kirkaldy, A. N. Patel, A. Hales, A. Ghosh, W. Ai, J. Chen, J. Yang, S. Li, M.-C. Pang, L. Bravo Diaz, A. Tomaszewska, M. W. Marzook, K. N. Radhakrishnan, H. Wang, Y. Patel, B. Wu, and G. J. Offer, "Lithium-ion battery degradation: what you need to know," Phys. Chem. Chem. Phys., vol. 23, pp. 8200–8221, 2021.

[7] Z. Song, X.-G. Yang, N. Yang, F. P. Delgado, H. Hofmann, and J. Sun, "A study of cell-to-cell variation of capacity in parallel-connected lithium-ion battery cells," eTransportation, vol. 7, p. 100091, 2021.

[8] S. Saxena, C. Le Floch, J. MacDonald, and S. Moura, "Quantifying ev battery end-of-life through analysis of travel needs with vehicle powertrain models," Journal of Power Sources, vol. 282, pp. 265–276, 2015.

[9] S. Greenbank and D. Howey, "Automated feature extraction and selection for data-driven models of rapid battery capacity fade and end of life," IEEE Transactions on Industrial Informatics, vol. 18, no. 5, pp. 2965–2973, 2022.

[10] R. Gallo, G. Bussolo, T. Paolo, M. Zampolli, R. Jaboeuf, E. Patti, and A. Aliberti, "An electric vehicle simulator for realistic battery signals generation from data-sheet and real-world data," in COMPSAC 2023, June 2023.

[11] Z. Wang, G. Feng, D. Zhen, F. Gu, and A. Ball, "A review on online state of charge and state of health estimation for lithium-ion batteries in electric vehicles," Energy Reports, vol. 7, pp. 5141–5161, 2021.

[12] G. Nuroldayeva, Y. Serik, D. Adair, B. Uzakbaiuly, and Z. Bakenov, "State of health estimation methods for lithium-ion batteries," International Journal of Energy Research, vol. 2023, 2023.

[13] X. C. A. Chacón, S. Laureti, M. Ricci, and G. Cappuccino, "A review of non-destructive techniques for lithium-ion battery performance analysis," World Electric Vehicle Journal, vol. 14, no. 11, 2023. [Online]. Available: https://www.mdpi.com/2032-6653/14/11/305

[14] J. Zhang, P. Wang, Y. Liu, and Z. Cheng, "Variable-order equivalent circuit modeling and state of charge estimation of lithium-ion battery based on electrochemical impedance spectroscopy," Energies, vol. 14, no. 3, 2021.

[15] S. Jiang and Z. Song, "A review on the state of health estimation methods of lead-acid batteries," Journal of Power Sources, vol. 517, p. 230710, 2022.

[16] Y. Li, K. Liu, A. M. Foley, A. Zülke, M. Berecibar, E. Nanini-Maury, J. Van Mierlo, and H. E. Hoster, "Data-driven health estimation and lifetime prediction of lithium-ion batteries: A review," Renewable and Sustainable Energy Reviews, vol. 113, p. 109254, 2019.

TABLE 8. MAE and RMSE of the SOH values predicted with the proposed regression models over synthetic data and real data before and after transfer learning. Column names refer to regression models; row names refer to feature extraction methods. The last column is the average feature extraction time from a single time window.

	MAE [%]									RMSE [%]									
	Synthetic data			Real data - Before TL			Real data - After TL			Synthetic data			Real data - Before TL			Real data - After TL			
Feature extraction method	**RR**	**RF**	**NN**	**RR**	**RF**	**NN**	**RR**	**RF**	**NN**	**RR**	**RF**	**NN**	**RR**	**RF**	**NN**	**RR**	**RF**	**NN**	**avg. time [s]**
MR+PCA	2.51	0.92	0.27	16.28	5.08	24.34	4.22	**1.97**	3.34	3.14	1.24	0.37	19.45	5.92	26.87	4.67	**2.56**	3.97	0.046
OLS	5.19	1.53	1.48	5.37	8.53	52.24	4.22	2.95	4.19	6.00	2.48	2.39	14.30	9.95	296.32	4.78	3.88	4.70	1.8 10
TS	5.19	1.48	1.48	4.40	8.15	43.70	4.11	3.30	3.69	5.99	2.39	2.39	5.04	9.59	80.24	4.76	3.94	4.36	0.47

TABLE 9. The computational time required by each feature extraction method for the feature extraction and inference phases per sample.

	Feature extraction			Inference		
	MR+PCA	OLS	TS	RR	RF	NN
Required execution time [s]	0.14	0.07	0.50	0.01	0.06	0.07

[17] L. Yao, S. Xu, A. Tang, F. Zhou, J. Hou, Y. Xiao, and Z. Fu, "A review of lithium-ion battery state of health estimation and prediction methods," World Electric Vehicle Journal, vol. 12, no. 3, 2021.

[18] N. Noura, L. Boulon, and S. Jemeï, "A review of battery state of health estimation methods: Hybrid electric vehicle challenges," World Electric Vehicle Journal, vol. 11, no. 4, 2020.

[19] D. Roman, S. Saxena, V. Robu, M. Pecht, and D. Flynn, "Machine learning pipeline for battery state-of-health estimation," Nature Machine Intelligence, vol. 3, no. 5, pp. 447–456, May 2021.

[20] Q. Li, D. Li, K. Zhao, L. Wang, and K. Wang, "State of health estimation of lithium-ion battery based on improved ant lion optimization and support vector regression," Journal of Energy Storage, vol. 50, p. 104215, 2022.

[21] B. Huang, H. Liao, Y. Wang, X. Liu, and X. Yan, "Prediction and evaluation of health state for power battery based on ridge linear regression model," Science Progress, vol. 104, no. 4, p. 00368504211059047, Oct 2021.

[22] J.-H. Leeand and I.-S. Lee, "Estimation of online state of charge and state of health based on neural network model banks using lithium batteries," Sensors, vol. 22, no. 15, 2022.

[23] D. Yang, Y. Wang, R. Pan, R. Chen, and Z. Chen, "A neural network based state-of-health estimation of lithium-ion battery in electric vehicles," Energy Procedia, vol. 105, pp. 2059–2064, 2017, 8th International Conference on Applied Energy, ICAE2016, 8-11 October 2016, Beijing, China.

[24] A. Russo, A. Fiore, and M. Aprea, "Predicting battery health capacity through machine learning techniques: Svr, random forest and fully-connected network," 09 2019.

[25] D. Li, J. Tang, B. Zhou, P. Cao, J. Hu, M. F. Leung, and Y. Wang, "Toward resilient electric vehicle charging monitoring systems: Curriculum guided multi-feature fusion transformer," IEEE Transactions on Intelligent Transportation Systems, vol. PP, pp. 1–11, 01 2024.

[26] E. Chemali, P. J. Kollmeyer, M. Preindl, Y. Fahmy, and A. Emadi, "A convolutional neural network approach for estimation of li-ion battery state of health from charge profiles," Energies, vol. 15, no. 3, 2022.

[27] M. Raman, V. Champa, and V. Prema, "State of health estimation of lithium ion batteries using recurrent neural network and its variants," in 2021 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT), 2021, pp. 1–6.

[28] P. Li, Z. Zhang, Q. Xiong, B. Ding, J. Hou, D. Luo, Y. Rong, and S. Li, "State-of-health estimation and remaining useful life prediction for the lithium-ion battery based on a variant long short term memory neural network," Journal of Power Sources, vol. 459, p. 228069, 2020.

[29] L. Ungurean, M. V. Micea, and G. Cârstoiu, "Online state of health prediction method for lithium-ion batteries, based on gated recurrent unit neural networks," International Journal of Energy Research, vol. 44, no. 8, pp. 6767–6777, 2020.

[30] P. Venugopal and V. T., "State-of-health estimation of li-ion batteries in electric vehicle using indrnn under variable load condition," Energies, vol. 12, no. 22, 2019.

[31] X. Lv, Y. Cheng, S. Ma, and H. Jiang, "State of health estimation method based on real data of electric vehicles using federated learning," International Journal of Electrochemical Science, vol. 19, no. 8, p. 100591, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1452398124001329

[32] L. Merkle, M. Pöthig, and F. Schmid, "Estimate e-golf battery state using diagnostic data and a digital twin," Batteries, vol. 7, no. 1, 2021.

[33] L. Song, K. Zhang, T. Liang, X. Han, and Y. Zhang, "Intelligent state of health estimation for lithium-ion battery pack based on big data analysis," Journal of Energy Storage, vol. 32, p. 101836, 2020.

[34] R. Gallo, T. Monopoli, M. Zampolli, R. Jaboeuf, P. Tosco, A. Aliberti, and E. Patti, "Comparative analysis of electric vehicle simulator for accurate battery pack internal signal generation," IEEE Transactions on Industry Applications, pp. 1–10, 2024.

[35] T. Barlow, S. Latham, I. S. McCrae, and P. Boulter, "A reference book of driving cycles for use in the measurement of road vehicle emissions," TRL Published Project Report, 2009.

[36] T. Stöttner, "Why data should be normalized before training a neural network," 2019. [Online]. Available: https://towardsdatascience.com/why-datashould- be-normalized-before-training-a-neural-network-c626b7f 66c7d

[37] M. Längkvist, L. Karlsson, and A. Loutfi, "A review of unsupervised feature learning and deep learning for time-series modeling," Pattern Recognition Letters, vol. 42, pp. 11–24, 2014.

[38] C. W. Tan, C. Bergmeir, F. Petitjean, and G. I. Webb, "Time series extrinsic regression," Data Mining and Knowledge Discovery, vol. 35, no. 3, pp. 1032–1060, May 2021.

[39] A. Dempster, D. F. Schmidt, and G. I. Webb, "Minirocket: A very fast (almost) deterministic transform for time series classification," in Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, ser. KDD '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 248–257.

[40] J. Shlens, "A tutorial on principal component analysis," 2014.

[41] T. Hastie, R. Tibshirani, and J. Friedman, The Elements of Statistical Learning: Data Mining, Inference, and Prediction, ser. Springer series in statistics. Springer, 2009. [Online]. Available: https://books.google.it/books?id=eBSgoAEACAAJ

[42] J. W. McKean, "Robust analysis of linear models," Statistical Science, vol. 19, no. 4, pp. 562 – 570, 2004.

[43] H. Theil, A Rank-Invariant Method of Linear and Polynomial Regression Analysis. Dordrecht: Springer Netherlands, 1992, pp. 345–381.

[44] P. K. Sen, "Estimates of the regression coefficient based on kendall's tau," Journal of the American Statistical Association, vol. 63, no. 324, pp. 1379–1389, 1968.

[45] X. Wang, X. Dang, H. Peng, and H. Zhang, "Theil-sen estimators in a multiple linear regression model," in Olemiss Edu, 2009.

[46] S. Ruder, "An overview of gradient descent optimization algorithms," 2017.

[47] T. K. Ho, "Random decision forests," in Proceedings of 3rd International Conference on Document Analysis and Recognition, vol. 1, 1995, pp. 278–282 vol.1.

[48] A. G. Baydin, B. A. Pearlmutter, A. A. Radul, and J. M. Siskind, "Automatic differentiation in machine learning: a survey," 2018.

RAIMONDO GALLO Raimondo Gallo received the B.S. degrees in information engineering and the M.S. degree in Information and Communication Technology for Smart Societies from the Polytechnic University of Turin, Italy. He is currently working as research assistant in the polytechnic university of Turin since 2022. His areas of research encompass the formulation of machine learning methodologies aimed at the prediction of time-series data, with a particular emphasis on meteorological analysis and the monitoring of electric vehicle battery performance.

TOMMASO MONOPOLI is an AI Applied researcher at LIKNS Foundation. He received the Master's degree in Data Science and Engineering from Politecnico di Torino in 2022. During his studies, he collaborated with Edison Spa, where he completed an internship and developed his thesis.

MARCO ZAMPOLLI works at Edison SpA as a researcher in the energy sector in the field of electric mobility. He is responsible too for managing systems and equipment of R&D Edison laboratories. His current activities is focused in testing and development of system dedicated to management of charging for electric vehicle and traction battery data acquisition.

RÉMI JACQUES PHILIBERT JABOEUF is a project manager at Edison Spa, specializing in electric vehicle charging technologies and processes. His work focuses on developing, experimenting with, and testing new advanced solutions to optimize the integration of electric vehicles with home energy systems.

PAOLO TOSCO is an R&D Manager at Edison SpA. Throughout his professional career, he has led research activities across a wide range of technical and scientific fields in the energy sector, including low-impact microcogeneration, energy storage via electrochemical systems, photovoltaic conversion, and electrochemical hydrogen production. He is currently involved in the development, experimentation, and testing of new technologies for electric vehicle charging.

EDOARDO PATTI is an Associate Professor at the Dept. of Control and Computer Engineering at Politecnico di Torino. His research interests concern: i) Ubiquitous Computing; ii) Internet of Things; iii) Software architectures with particular emphasis on infrastructure for Ambient Intelligence; iv) Machine learning for both energy data analysis and medical applications; v) Cosimulation techniques for multi-energy systems integration in Smart Cities and Energy Communities.

ALESSANDRO ALIBERTI is Assistant Professor w/ time contract at the Interuniversity Dept. of Regional and Urban Studies and Planning at Politecnico di Torino. Moving towards smart and sustainable energy use, his research activities mainly focus on the design and the optimization of innovative machine learning methodologies, by exploiting primarily neural networks, for the forecasting of time-series in Smart City context. During his academic experience, he proposes innovative and optimized data stream processing and machine learning methodologies, ranging from energy IEEEbiographynophoto environmental data and moving to data from CPS systems.

• • •