A Deep Neural Network based model for the prediction of Hybrid Electric Vehicles carbon dioxide emissions

Energy and AI 5 (2021) 100073

Contents lists available at ScienceDirect

Energy and AI

journal homepage: www.elsevier.com/locate/egyai

A deep neural network based model for the prediction of hybrid electric vehicles carbon dioxide emissions

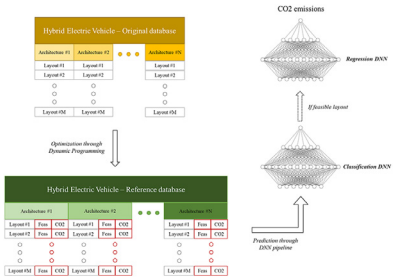
Claudio Maino*, Daniela Misul, Alessandro Di Mauro, Ezio Spessa

IC Engines Advanced Laboratory, Dipartimento Energia "Galileo Ferraris", Politecnico di Torino, c.so Duca degli Abruzzi 24, 10129 Torino, Italy

HIGHLIGHTS

- Generation of a HEV database based on the design parameters of different HEV architectures.
- Definition of a data management approach for introducing the data into the prediction model.
- Definition and automatic calibration of the main Deep Neural Network hyperparameters.
- Investigation on the performance of the classification Deep Neural Network and the entire pipeline when tested upon different HEV datasets.
- Analysis of the pipeline behavior in case of classification of false positive HEV layouts.

GRAPHICAL ABSTRACT



ARTICLE INFO

Article history:
Received 3 February 2021
Received in revised form 28 March 2021
Accepted 28 March 2021
Available online 6 April 2021

Keywords:
Deep neural networks
Carbon dioxide emissions
Hybrid electric vehicles
Dynamic programming

ABSTRACT

Hybrid electric vehicles (HEV) are nowadays proving to be one of the most promising technologies for the improvement of the fuel economy of several transportation segments. As far as the on-road category is concerned, a wise selection of the powertrain design is needed to exploit the best energetic performance achievable by a HEV. Amongst the methodologies developed for comparing different hybrid architectures, global optimizers have demonstrated the capability of leading to optimal design solutions at the expense of a relevant computational burden. In the present paper, an innovative deep neural networks-based model for the prediction of tank-to-wheel carbon dioxide emissions as estimated by a Dynamic Programming (DP) algorithm is presented. The model consists of a pipeline of neural networks aimed at catching the correlations lying between the design parameters of a HEV architecture and the main outcomes of the DP, namely powertrain feasibility and tail pipe CO_2 emissions. Moreover, an automatic search tool (AST) has been developed for tuning the main hyperparameters of the networks. Interesting results have been registered by applying the pipeline to three databases related to three different HEV parallel architectures. The capability of the pipeline has been proved through an extensive testing campaign made up by multiple experiments. Classification performances above 91% as well as average regression errors below 1% have been achieved during an extensive set of simulations. The presented model could hence be considered as an effective tool for supporting HEV design optimization phases.

* Corresponding author.
E-mail addresses: claudio.maino@polito.it (C. Maino), daniela.misul@polito.it (D. Misul), alessandro.dimauro@polito.it (A. Di Mauro), ezio.spessa@polito.it (E. Spessa).

https://doi.org/10.1016/j.egyai.2021.100073

Nomenclature

Acronyms

AI	Artificial Intelligence
AST	Automatic Search Tool
BC	Battery Charging
BEV	Battery Electric Vehicle
CrateChar	Charge C-rate
cDNN	Classification Deep Neural Network
CoD	Coefficient of Determination
CB	Computational Burden
CM	Confusion Matrix
DNN	Deep Neural Network
DNNs-PM	Deep Neural Networks based Predictive Model
DoE	Design of Experiment
DesPars	Design Parameters
CrateDis	Discharge C-rate
DP	Dynamic Programming
EMS	Energy Management System
ED	Engine Displacement
ES	Engine State
ECMS	Equivalent Consumption Minimization Strategy
FDf	Front Final Drive
FCEV	Fuel-Cells Electric Vehicle
FC	Fuel Consumption
HEV	Hybrid Electric Vehicle
ICE	Internal Combustion Engine
ML	Machine Learning
MS	Massive Simulation
MCC	Matthews Correlation Coefficient
MG	Motor/Generator
MGPower	Motor/Generator Peak Power
MGSpRatio	Motor/Generator Speed Ratio
NN	Neural Network
PMP	Pontryagin's Minimum Principle
PS	Power Split
PEratio	Power-to-Energy Ratio
PE	Pure Electric
PT	Pure Thermal
FDr	Rear Final Drive
FDrSpRatio	Rear final Drive Speed Ratio
rDNN	Regression Deep Neural Network
RA	Results Accuracy
RMSE	Root Mean Squared Error
SoC	State of Charge
TC	Torque Coupling Device
t/t	Train-to-Test
t/v	Train-to-Validation
VL	Variation List
WHVC	World Harmonized Vehicle Cycle

Variables

$\#_{gear}$	Set of gear numbers
a	Vehicle acceleration [m/s^2]
C	Model-predicted tank-to-wheel CO_2 emissions [g/km]
C_{bat}	Battery capacity [Ah]
CO_2	Tank-to-wheel CO_2 emissions [g/km]
$C_{rate,dis}$	Charge C-rate [-]
$C_{rate,dis}$	Discharge C-rate [-]
d	Total covered distance [km]
$DesPars$	Set of design parameters
$\hat{E}$	Average prediction error
ES	Engine state
F	Set of feasible layouts
F_{aero}	Aerodynamic drag [N]
FC	Cumulative fuel consumption [g]
FN	Number of false negatives
FP	Number of false positives
F_{road}	Resistive force coming from road slope [N]
F_{roll}	Rolling resistance [N]
I_{bat}	Battery current intensity output [A]
$I_{ch,max}$	Maximum current intensity admitted in charge mode [A]
$I_{dis,max}$	Maximum current intensity admitted in discharge mode [A]
J	Objective function of the Dynamic Programming
k	Conversion factor [-]
L	Number of nodes in the hidden layer
L_1	Number of nodes in the first hidden layer
$Max\ E$	Maximum prediction error
$min\ E$	Minimum prediction error
m_{veh}	Vehicle mass [kg]
n	Number of total predictions
N_{GN}	Total admissible number of gears
P_{batt}	Battery power output [W]
R^2	Coefficient of determination
r_{wheel}	Wheel effective radius [m]
SoC	Battery state of charge [-]
SS_{res}	Sum of the squared residuals
SS_{tot}	Total sum of squares
$std(x)$	Standard deviation
t	Time [s]
$t_{char,I=max}$	Total battery charging time at maximum current [s]
$t_{dis,I=max}$	Total battery discharging time at maximum current [s]
T_{ICE}	Internal combustion engine output torque [N m]
T_{MG}	Motor/generator output torque [N m]
TN	Number of true negatives
T_{out}	Vehicle output torque [N m]
TP	Number of true positives
U	Set of control variables
v_{target}	Cycle-imposed vehicle velocity [m/s]
v_{veh}	Real vehicle velocity [m/s]
X	Set of state variables
x	Input feature of the neural network before normalization
$\tilde{x}$	Input feature of the neural network after normalization
$\bar{x}$	Average value of the input features of the neural network
$\bar{y}$	Average value of the target CO_2 emissions [g/km]
y	Target CO_2 emissions [g/km]

Greek symbols

α	Power-split level
α_{ICE}	Internal combustion engine torque fraction
α_{MG}	Motor/generator torque fraction
$\eta_{\tau ICE}$	Internal combustion engine gear efficiency
$\eta_{\tau MG}$	Motor/generator gear efficiency
θ_c	Set of structural and algorithmic parameters for the classification deep neural network
θ_r	Set of structural and algorithmic parameters for the regression deep neural network
τ_{ICE}	Internal combustion engine gear ratio
τ_{MG}	Motor/generator gear ratio
ω_{MG}	Motor/generator speed [rad/s]

1. Introduction

The increasing need for reducing pollutants and greenhouse gases emissions from road transportation systems has tailored the research of the last few years towards new technologies, such as battery electric vehicles (BEVs), fuel-cells electric vehicles (FCEVs) and hybrid electric vehicles (HEVs) [1]. Even though BEVs and FCEVs show promising results in terms of fuel economy, few but critical limitations arise: BEVs are still not capable of guaranteeing an electric range comparable with the driving range of a fuel-propelled vehicle due to battery limitations [2] and are typically characterized by higher initial costs caused by the current components' markets prices [3]; FCEVs are still in the early development stages, at least in the transportation field [4] and consistent improvements are needed regarding safe on-board hydrogen storage systems and recharging infrastructures [5]. On the other hand, HEVs have demonstrated their capabilities in effectively reducing fuel consumption-related CO_2 and pollutant emissions [6], while providing reliable driving ranges. Moreover, the HEVs penetration in the market is increasing through time thanks to a good compromise between retail prices, total cost of ownership [7], independency from an incomplete recharging infrastructure and on-board safety [8].

Different HEV architectures have been studied and discussed in the literature, such as series, parallel and complex series/parallel architectures [9,10]. Briefly, any architecture is characterized by a given position of the electric machine/s in the powertrain and a customized connection between the electric and thermal energy sources. Furthermore, any HEV architecture is also characterized by different design specifications (e.g., engine sizing, electric motors' power and battery capacity, etc.). For the sake of clarity, the term "HEV layout" (or simply "layout") is hereafter used to address to any HEV configuration for which the main components' sizes have been defined.

The necessity of defining a proper methodology for the comparison of very different HEV layouts hence arises. A widely used first step is that of discriminating between the three above-mentioned HEV architectures in order to reduce the design domain. Therefore, assumptions have to be made about the HEV Energy Management System (*EMS*), i.e. the algorithm (or set of algorithms) that coordinates and controls the propellers operation during the vehicle use [11,12]. Several methodologies have been developed in the last years and can be classified in three main categories: heuristic algorithms, static optimizers and global optimizers [13]. The performance of the different EMSs can be compared by means of two parameters: computational burden (CB) and results accuracy (RA). Heuristic controllers (i.e. rule-based strategies and fuzzy logics) are characterized by the lowest CB and RA [14]. Static optimizers, such as the Equivalent Consumption Minimization Strategy (*ECMS*) [15], show intermediate CB and RA. Finally, global optimizers, such as Dynamic Programming (*DP*) [16] and Pontryagin's Minimum Principle (*PMP*) [6], achieve the highest possible RA by identifying the optimal control strategy at the expense of the most expensive CB.

From an optimal design perspective, once the HEV architecture is defined, a global optimizer is typically selected as the most promising EMS thanks to its capability of exploiting the optimal energetic performance achievable by a HEV layout. Such an algorithm allows for properly comparing different HEV layouts at their best and would have to be run through a wide range of design parameters. Given the high CB of such massive set of simulations (MS), a learning algorithm could therefore be trained to grasp the correlations between the layout specifications (i.e. design parameters) and the global optimizer's outputs. Thanks to this approach, fewer simulations could be performed, drastically reducing the CB still preserving the RA.

A promising family of algorithms with the capability of learning from a data set is classified as Machine Learning (ML). ML algorithms are a branch of the Artificial Intelligence (AI) framework as they usually encompass automatic computing procedures that learn a task from a set of samples without being explicitly programmed [16].

A typical macro-discrimination is made between classification and regression-aimed ML algorithms [17]. Briefly, classification algorithms are employed to predict a class the processed inputs belong to, whereas regression algorithms are used to predict numerical values. Beyond their capability of predicting a class or a figure, ML algorithms can be further divided into two sub-branches: supervised learning and unsupervised learning [18] algorithms. As far as supervised learning algorithms are concerned, "labelled data" have to be available within the input dataset: labelled data consist of data for which the class/value to be predicted is a priori known. In fact, the algorithm is trained to understand the connections between the inputs and the outputs so as to replicate them for new and unknown data. On the other hand, unsupervised learning algorithms use "unlabelled data" as they are trained to catch the patterns underneath the data and use them to intercept the trend inputs.

Regarding the CO_2 prediction task for HEVs, the choice of a global optimizer would guarantee the estimation of optimal outputs for each of the tested layouts. Labelled data would be available, resulting in a supervised approach problem.

Since the numerical correlations between the layouts' specifications (i.e. the inputs) and the global optimizer's results (i.e. the outputs) are expected to be highly non-linear [19], Neural Networks (NNs) have been selected as the most promising solution for the considered prediction task. NNs are a sub-family of ML algorithms and are considered as particularly advantageous when complex prediction problems are targeted [20]. Common features are the layered structure, a variable number of nodes for each computational layer and a recursive approach that is of paramount importance in the learning process [21]. Concerning the NNs algorithm, the authors chose to resort to Deep-NNs algorithm (*DNNs*) as the latter proved to possess a greater learning potential due to their deep layered structure [22,23,24]. In fact, the term "Deep" refers to the wide number of layers in the net itself.

An innovative exploitation of DNN is proposed in the present paper. The latter mainly differs from the literature solutions where neural networks are basically used to monitor real-time vehicle performances and to dictate a proper control strategy. As a matter of fact, the potentials of a learning DNNs-based Predictive Model (*DNNs-PM*) when applied to the optimization task of HEVs design parameters through the prediction of tank-to-wheel CO_2 emissions are presented in the paper. More specifically, the developed method could be easily implemented within the design operations of HEV fleets and could represent a fast and reliable way to identify possible optimal design parameters combinations with respect to a foreseen driving mission.

A single VL (i.e. only one MS is performed) is generated by means of a Design of Experiment (DoE) approach [25] for three different HEV parallel architectures and processed by a DP algorithm. Once the simulations are completed, a dataset comprising the layout specifications is generated for each architecture and fed into the DNNs-PM. At this stage, two steps are performed by the model: first, the discrimination between feasible/unfeasible layouts (i.e. a classification task) is carried out; then, the prediction of the tank-to-wheel CO_2 emissions of the admissible layouts (i.e. regression task) is performed. In order to obtain a dual nature prediction, a two-DNN pipeline has been developed.

2. Vehicle model and control logic

In the current section, the low-throughput models adopted for reproducing the three HEVs parallel architectures are presented together with the formulation of the EMS. Finally, a thorough description of the datasets is provided.

2.1. HEV model

In the present paper, three HEV parallel architectures featuring only one electric machine have been analysed, namely P2, P3 and P4. The just mentioned architectures, and their related design parameters, are derived from a reference heavy-duty vehicle for which an experimental

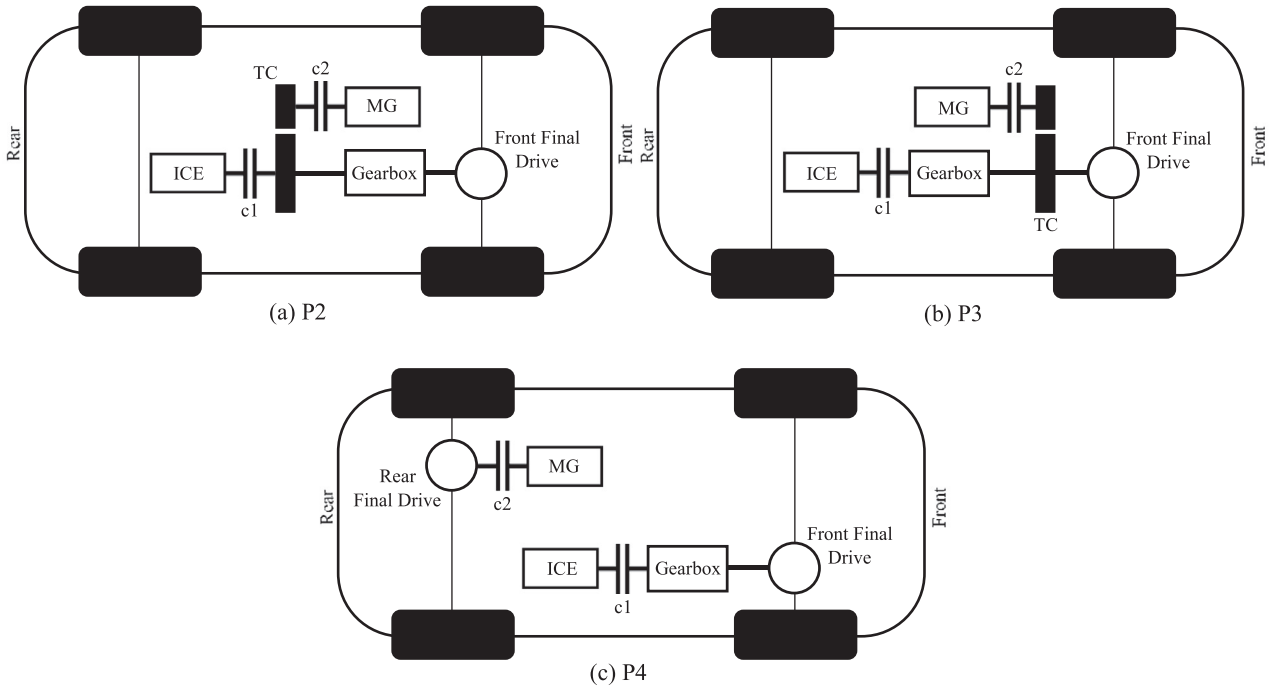


Fig. 1. P2, P3 and P4 parallel hybrid architectures.

campaign was conducted in the context of an industrial partnership. It is worth noticing that this choice has been driven by the amount of experimental data that were otherwise not possible to retrieve. Still, the versatility of the method allows for a simple transfer to other HEV design optimizations, such as those related to passenger-cars. A schematic view of the different hybrid configurations is proposed in Fig. 1. For the sake of clarity, the electric machines involved in the present study are considered capable of providing traction to the vehicle as well as of allowing for the battery recharging during regenerative braking phases. Hence, the electric machines will be referred as motors/generators (*MGs*).

In the P2 configuration, the MG is placed in between the Internal Combustion Engine (*ICE*) and the gearbox; the connection between the MG and the ICE shaft is achieved considering a Torque Coupling device (*TC*). The disengagement of both the ICE and the MG is realized thanks to the use of two clutches (c1 and c2): during pure electric operating modes, c1 is engaged and c2 is disengaged whereas the inverse engagement is used during pure thermal operating modes. In the P3 configuration, the MG is connected to the main shaft through a TC positioned on the gearbox output shaft; the same system of clutches of the P2 configuration is used to disengage the ICE or the MG in the P3 architecture. Differently from the P2 and P3 configurations, in the P4 one, the MG drives a separate axle with respect to the one powered by the ICE. Finally, a Front Final Drive (*FDf*) is defined for the three architectures whereas a Rear Final Drive (*FDr*) is considered in the P4 configuration.

The operating modes of each above-mentioned parallel architecture can be classified as follows [6]:

- Pure Electric (*PE*) mode: the driver's power demand is met by the MG alone (the ICE is off); regenerative breaking is included in the category.
- Pure Thermal (*PT*) mode: the driver's power demand is met by the ICE alone (the MG is off).
- Power Split (*PS*) mode: the driver's power demand is met synergically by the ICE and the MG.
- Battery Charging (*BC*) mode: the ICE output power is higher than the power required to move the vehicle so that the exceeding power could be used to recharge the battery as the MG is employed as generator during a traction a phase.

The mathematical formulation of the HEV model is derived from a quasi-static backward-facing modelling approach: the power to be delivered by the on-board energy sources is derived from the total power demand imposed by the driving mission considering the selected power-split levels [7]. The choice of such a simplified HEV model has been forced by the necessity of running multiple simulations embedding a computationally-heavy HEV control strategy (i.e. DP, see Section 2.3). Anyhow, no influence on the consistency of the proposed method for the prediction of CO_2 emissions through DNNs arises by the level of the HEV model complexity.

Once the power required to the vehicle is estimated, the torque to be exerted by the transmission T_{out} is computed, resulting in the following formulation:

$$T_{out} = \left(F_{roll} + F_{road} + F_{aero} + m_{veh} \cdot a\right) \cdot r_{wheel} \tag{1}$$

where F_{roll}, F_{road} and F_{aero} contribute to the resistive load and are the rolling resistance, the resistive force coming from the road slope, and the aerodynamic drag, respectively, m_{veh} represents the vehicle mass, a represents the acceleration imposed to the vehicle and finally r_{wheel} is the wheel effective radius.

For a given GN, the angular speeds of both ICE and MG can be computed as they are directly linked to the vehicle velocity [6]. Therefore, the output torque T_{out} of the two power components can be computed ensuring the torque balance:

$$T_{out} = T_{ICE} \cdot \alpha_{ICE} \cdot \tau_{ICE} \cdot \eta_{\tau ICE} + T_{MG} \cdot \alpha_{MG} \cdot \tau_{MG} \cdot \eta_{\tau MG} \tag{2}$$

where τ and η are the gear ratios and the relative gear efficiencies, respectively; α_{MG} represents the fraction of the torque outputted by the MG (T_{MG}) with respect to the total torque required by the vehicle T_{out}, and finally α_{ICE} represents the fraction of the torque outputted by the ICE (T_{ICE}) with respect to T_{out}.

So as to simulate the response of the HEV propulsion systems (i.e. ICE, MG and battery) to the torque and speeds requirements, three main sub-models are built. The ICE is modelled by means of an empirically derived 2D look-up table which relates the fuel consumption to the ICE speed and torque. The maximum load curves, which are function of the ICE speed, are intrinsically embedded in the just mentioned 2D look-up table. The details about the reference ICE are protected by non-

disclosure agreements and could therefore not be presented. Anyhow, the validity of the proposed method is not compromised since the reference data are only employed to properly scale the look-up tables when variations in ICE design occur.

At the end of the simulation, the tank-to-wheel CO_2 emissions in g/km can be obtained through:

$$CO_2 = \frac{k \cdot FC}{d} \tag{3}$$

where d represents the total distance covered in km, FC is the cumulative fuel consumption in g and k is a conversion factor [26].

Regarding the MG, the energy losses occurring during the power conversion from the electric to the mechanical form (and vice versa) are modelled by means of an empirically derived 2D efficiency map function of the MG speed and torque in traction. As for the ICE, the maximum torque curve admitted is intrinsically embedded in the 2D efficiency map. For regeneration, the MG efficiency map and full load curve are considered to be symmetrical. About the battery sub-model, an equivalent open circuit model is employed considering experimentally derived open circuit voltage and internal resistance as functions of the battery State of Charge (*SoC*) [27]. The battery power output P_{batt} can be computed adopting the following formulation:

$$P_{batt} = \omega_{MG} \cdot T_{MG} \cdot \frac{1}{{\eta_{MG}}^{sign(\alpha_{MG} \cdot T_{MG})}} \tag{4}$$

where ω_{MG} is the speed of the MG, T_{MG} is the MG output torque and η_{MG} is the efficiency of the MG. If the MG is used as motor α_{MG} is positive so η_{MG} divides the right-hand side, vice versa the latter is multiplied by η_{MG} if the MG is employed as motor (i.e. negative α_{MG} values). The variation of the battery SoC is also evaluated using the aforementioned equivalent open circuit model.

$$SoC_t = SoC_{t-1} - \int \frac{I_{bat}}{C_{bat}} \cdot dt \tag{5}$$

Where SoC_t is the battery SoC at time t, SoC_{t-1} is the battery SoC for the previous time instant, I_{bat} is the current intensity delivered by the battery pack, and, finally, C_{bat} is the battery capacity.

2.2. Database definition

The learning algorithm that is presented in the next sections relies upon an input set of features that are thought to be meaningful and informative about the phenomenon to be predicted (i.e. the CO_2 emissions).

Considering the HEV model presented in Section 2.1, eight design parameters have been chosen as the most informative features about the CO_2 estimates resulting from the simulation of a HEV layout for a given driving mission:

- Engine displacement (*EngDispl*): displacement of the ICE in l.
- Motor/generator peak power (*MGPower*): power of the MG in kW.
- Power-to-energy ratio (*PEratio*): ratio between the MG peak power and the maximum energy storable in the battery pack measured in W/Wh.
- Front final drive speed ratio (*FDfSpRatio*): transmission ratio at the front final drive.
- Rear final drive speed ratio (*FDrSpRatio*): transmission ratio at the rear final drive. It assumes positive values only for the P4 architecture since the MG is connected on the rear axle. Therefore, it is null for P2 and P3 architectures.
- Motor/generator speed ratio (*MGSpRatio*): transmission ratio at the TC level. It assumes positive values for both P2 and P3 architectures since the MG is coupled to the ICE shaft through the TC. It is null for P4 architectures.
- Discharge C-rate (*CrateDis*): it is defined as:

$$C_{rate,dis} = \frac{I_{dis,max}}{C_{batt}} \tag{6}$$

Where $I_{dis,max}$ is the maximum current admitted in discharge (measured in A) and C_{batt} is the battery capacity (measured in Ah).

- Charge C-rate (*CrateChar*): it is defined as:

$$C_{rate,char} = \frac{I_{char,max}}{C_{batt}} \tag{7}$$

Where $I_{dis,max}$ is the maximum current admitted during charge mode (measured in A).

Even if the values assumed by each design parameter have to be consistent with real HEV applications, a spread range of values can be exploited within a design optimization operation. In the present research activity, the values of the design parameters related to each layout in the VL have been generated by means of a DoE technique, namely the Sobol sequence [28]. Thanks to the latter, once the lower and upper threshold of the existing range are defined along with the desired amount of points (i.e. the number of layouts to be produced), the VL is automatically built through a space filling approach based on the generation of a quasi-random sequence using primitive polynomials [29]. In the present research, 1500 samples (i.e. different HEV layouts) have been generated for each HEV architecture presented in Section 2.1.

2.3. Dynamic programming

A hybrid powertrain requires an EMS to manage the operations of the on-board multiple power sources (i.e. battery, MG and ICE). The DP has been selected as the most promising global optimizer for the present research as it proved to be the most effective algorithm to be adopted when a fair comparison of the best energetic performances is required for different HEVs.

Once the driving mission is chosen and discretized by means of a given time step (in this paper, 1 second has been selected), the DP is used to assess for the optimal control trajectory based on a user-defined objective function (also called cost function) [19,30]. According to the algorithmic procedure, the cost function is computed for any possible combination of discretized control and state variables at each discrete time instant, starting from the last one and going backwardly to the first one. The control variables coordinate the actions that could be realized by the controller, whereas the state variables completely describe the state of the vehicle.

Three inputs have to be defined so that the DP could perform the desired computations: the state variables space X (i.e. the set of state variables), the control variables space U (i.e. the set of control variables), and the objective function J. In the present research, they have been defined as follows:

$$\begin{gathered} X = \begin{Bmatrix} SoC \\ ES \end{Bmatrix}, U = \begin{Bmatrix} \#_{gear} \\ \alpha \end{Bmatrix} \\ J = \min\left(CO_2\right) \propto \min(FC) \end{gathered} \tag{8}$$

where the battery SoC and the Engine State (*ES*) are representative of the state variables whereas the gear numbers $\#_{gear}$ and the power-split levels α are representative of the control variables; the objective function has been set as the CO_2, which is a direct function of the fuel consumed during the driving mission. According to the necessity of creating a grid of values to be managed by the DP, the battery SoC is defined by means of a set of discretized SoC levels whereas the ES is considered as a binary variable (i.e. ICE on/off). Similarly, $\#_{gear}$ represents the set of possible gear numbers and is composed by integer values in [1, N_{GN}] range, where N_{GN} is the total number of gears admissible. Contrarily, despite the power split levels α should correctly be represented by any real number (i.e. theoretically infinite ICE-MG power combination), only a finite amount of levels have been evaluated in order to restrain the CB.

Moreover, two additional constraints are imposed to the optimization algorithm:

$$\begin{gathered} v_{veh} = v_{target} \\ \begin{cases} SoC_{end} = SoC_{start} \pm tolerance \\ 0.4 \leq SoC \leq 0.8 \end{cases} \end{gathered} \tag{9}$$

Table 1
Extract of the P4 dataset.

Example[#]	EngDispl[l]	MGPower[kW]	PERatio[W/Wh]	FDfSpRatio[-]	FDrSpRatio[-]	CrateDis[1/h]	CrateCh[1/h]	CO_2[g/km]
1	2.62	106	5.1	4.14	14.80	8.8	9.3	377.1
2	3.92	113	5.1	4.58	10.48	8.0	8.0	389.4
…	…	…	…	…	…	…	…	…
1500	3.94	52	29.3	4.69	10.44	6.3	11.5	10000.0

where SoC_{end} and SoC_{start} are the SoC at the battery SoC at the end and at the start of the driving mission, respectively, v_{veh} represents the real vehicle velocity whereas v_{target} represents the cycle-imposed velocity. It is worth noticing that a tolerance (1e-5) around the final SoC value has been taken into account so that numerical approximation operations could not affect the feasibility of a given control trajectory. Anyhow, the final SoC constraint has been imposed so that a satisfying battery charge sustaining strategy could be guaranteed throughout the driving missions [13]. Finally, the velocity constraint ensures a complete match between the considered cycle-imposed velocity and the vehicle one.

If any of the constraints described by Eq. (9) is not fulfilled, the simulation is stopped and the corresponding HEV configuration is defined as non-acceptable (i.e. unfeasible). Such an event can occur at two moments: in a pre-simulation phase [6] or during the DP backward run. In the pre-simulation phase, the entire set of calculations about the power and speed required at any stage of the driveline for any combination of GN and α is performed. At this step, the impossibility of satisfying the road requirements in one or more time steps could be discovered. Such a condition is defined "pre-simulation unfeasibility". Within a pre-simulation feasibility check, the maximum power levels demanded by the driving mission at each discretized time step are checked upon. Such a process would differ from the 'unfeasibility' deriving from tailored and user-defined maximum power requests to the vehicle. In order to comply with such different constraints, an index for the vehicle performance request should be better defined and the whole procedure presented in the paper could be hence repeated assigning the analysed HEV layouts with a score corresponding to the performance index. If no pre-simulation unfeasibility is detected (i.e. the velocity required by the road can be fully matched), the first constraint of Eq. (9) is fulfilled and the DP simulation can begin. At this stage, the backward set of calculation begins and the optimal control trajectory is progressively built. The second constraint of Eq. (9) (i.e. the battery SoC constraints) is verified during the DP computations at each time step. Before reaching the initial time step, the battery SoC could have dropped below or risen above the imposed SoC window (i.e. 40% at minimum and 80% at maximum); moreover, once the initial time step has been reached, the specified tolerance could be exceeded. If one of the two SoC checks is failed, the simulation is stopped and a "SoC unfeasibility" is found. Any HEV layout leading to an unfeasibility is stored as an unfeasible layout. Such layouts will be considered as the opposite to the "feasible" one, which represent those HEV configurations leading to a successful simulation.

In Table 1, as an example, an extract of the database obtained at the end of the MS and used for the prediction of the P4-related emissions is reported. It is constituted by 1500 entries (number of rows) and 8 features (number of columns), the latter comprising 7 features for the design parameters (columns 2nd to 8th of Table 1) and 1 feature for the CO_2 emissions related to the DP-based optimal control strategy (9th column of Table 1). It is worth observing that the same structure layout applies for the datasets of the P2 and P3 architectures. As from the last row of Table 1, a CO_2 emission value equal to 10000.0 g/km might arises: such a value is assigned by the DP to the unfeasible layouts as a warning. It represents a purely numeric value, and it has been chosen by the authors to avoid misleading results that could affect the training process of the DNNs-PM. In fact, the expected results are far below this value, thus the unfeasible layouts can be easily spotted.

3. Deep neural networks for CO_2 prediction

The approach used for the prediction of the DP-based CO_2 emissions is made up of a couple of steps: first, the feasibility/unfeasibility of a specific HEV layout, and then the CO_2 estimate for the only layouts considered as feasible. For such a reason, both macro-implementations of a ML algorithm have been involved in the study, namely classification and regression.

After the description of the model overall logic, a set of preprocessing operations aimed at preparing a suitable dataset for the algorithms training is presented. Particularly, two steps are examined: targets definition and dataset normalization; finally, the dataset should be split into smaller sets, distinguishing from training, validation and test samples.

3.1. From the database definition to the CO_2 regression

Once the general dataset has been successfully extracted, the preprocessing set of operations begins followed by the model training phase; then, the model validation occurs in order to verify the effectiveness of the neural networks; finally, the model is considered as ready to be test on unknown data. The dual nature prediction task is performed under a two-steps procedure:

1 Assessment of the HEV layout feasibility: a Classification Deep Neural Network (cDNN) is used to predict whether a specific layout is capable of completing the cycle or not (feasible/unfeasible layout).
2 Prediction of the CO_2 emissions: a Regression Deep Neural Network (rDNN) is used to predict the CO_2 emitted by the feasible layouts.

The formulation of the DNNs-PM can be written as:

$$\begin{cases} F = f\left(DesPars,\ \theta_c\right) \\ C = f\left(DesPars(F),\ \theta_r\right) \end{cases} \tag{10}$$

Where F represent the set of feasible HEV layouts to be considered for the regression task, $DesPars$ represents the layouts' set of design parameters, C are the CO_2 emissions predicted by the model while θ_c and θ_r represent the set of network structural and algorithmic parameters for the cDNN and rDNN, respectively.

The workflow of the DNNs-PM is summarized in Fig. 2. After a preprocessing operation aimed at manipulating the data so as to make them interpretable by the DNNs, the first network to be trained is the cDNN. During the learning process, an Automatic Search Tool (AST) is employed to select the most effective combination of network's parameters. Once the feasible layouts are identified and the unfeasible ones are discarded (i.e. filtering procedure), the database is ready to be employed in rDNN's development. The rDNN learning process can finally take place with the support of the AST and the CO_2 prediction can occur.

3.2. Data management and normalization techniques

A first distinction is to be made when referring to the target's nature of the cDNN and rDNN. In the regression task performed by the rDNN, no action has been performed on the values of the CO_2 column of Table 1 as the rDNN is asked to predict the exact CO_2 estimates. Contrarily, given that the cDNN is supposed to perform a classification, categorical targets are needed instead of numerical targets, both for training and prediction

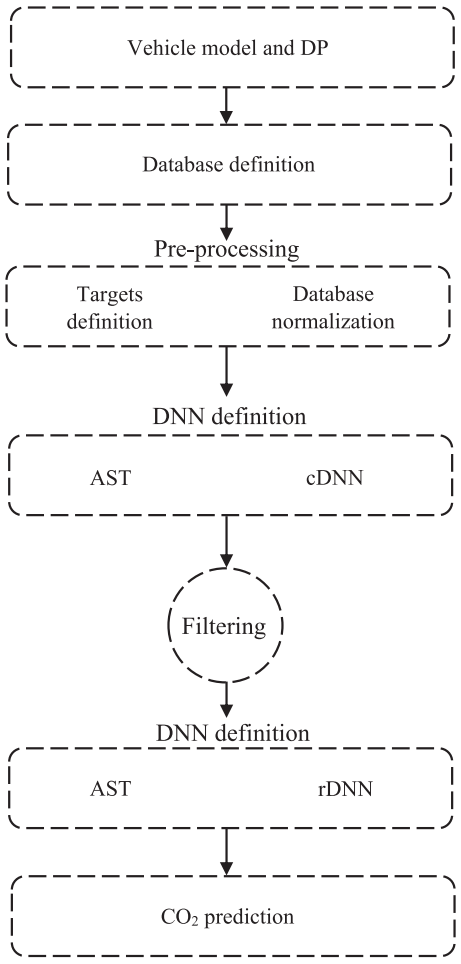


Fig. 2. Workflow of the DNNs-PM.

purposes. To this end, the CO_2 column is replaced by another one, featured by the same amount of points and defined as:

$$Labels = \begin{cases} 1 \; if \; CO_2 \neq 10000 \\ 0 \; if \; CO_2 = 10000 \end{cases} \tag{11}$$

Once again, the value "10000" is an intentionally out-of-scale fictitious value assigned to the unfeasible layouts to clearly distinguish them from the feasible ones.

Another important consideration has been made about the values assumed by the design parameters. The seven descriptive features, as it can be noticed in Table 1, show different order of magnitudes since they represent different physical quantities. From a numerical perspective, such a condition can lead to inefficient training processes as the network predictions could be learnt only through the features characterized by larger values [31,32]. An effective and fast solution is the normalization the of whole dataset. This procedure is usually accomplished by substituting each entry of the dataset with an equivalent one using a relation which is able to enforce the same range of variation for all the features, preserving simultaneously their informative potential [33,34]. To avoid unbalanced distributions of data in the normalized dataset, "standardization" is chosen as normalization technique. All the dataset features are then modified as follows:

$$\tilde{x}_i = \frac{x_i - \bar{x}}{std(x)}, \; i = 1 \dots N \tag{12}$$

where:

- $\tilde{x}_i$ is the i-th layout's feature after normalization.
- x_i is the i-th layout's feature before normalization.
- $\bar{x}$ is the average of the considered feature.
- $std(x)$ is the standard deviation of the considered feature.

Thanks to the standardization of the input dataset, the entire set of descriptive features of the dataset vary in the same order of magnitude.

The input dataset can actually be split into smaller datasets, namely training-set, validation-set, and test-set. The larger share of data is selected as training-set and processed during the training phase by the DNNs to catch the inputs/outputs correlations. Then, two smaller sets are generated: the former is employed to validate the DNNs (i.e. validation-set) during validation procedures aimed at defining the best possible networks' parameters combination; the latter (i.e. test-set) is employed to test the network on new and unknown data.

Within the present research, the training, validation, and test sets have been selected in a stochastic manner so that any DNN behaviour would not be biased by specific data splits. Particularly, the numerosity of the whole training and validation set has been defined as a consequence of a tuneable network parameter, namely Train-to-Test (t/t) ratio. The latter is typically considered too strongly affect the model performance as it modifies the number of data available during the training, validation and test phases [23].

3.3. Multi-stage deep neural network model

A distinction is to be made regarding the parameters of the neural networks considered in the present study: weights, fixed-parameters and hyper-parameters. The weights have been considered as the parameters involved in the computation of the activation functions and are autonomously tuned during the training process (specifically during the back-propagation phase [18]). The fixed-parameters have been considered as the features or the numeric parameters to be a priori defined by the user before the training process of the networks begins: in the present work, the values of the fixed parameters have been chosen after several attempts in finding the optimum time/accuracy trade-off [35] and kept unchanged for the entire set of analyses (the description and investigation of these parameters is beyond the scope of this study and the related investigations are not presented in this paper). The list of fixed-parameters is presented in Table 2. Finally, the hyper-parameters have been considered as the networks' features to be tuned during the training process by means of a dedicated tool (AST); given the different nature of the DNNs involved in the pipeline, different sets of hyper-parameters have been considered for the cDNN and rDNN and will be detailed in the following sections. For the sake of completeness, the simulations presented in the paper have been performed on a 2.4GHz CPU-12GB RAM personal computer.

3.4. Performance index of the cDNN

As far as the classification task of the cDNN is concerned, a binary classification (i.e. each example belongs to one of the two available classes) has to be performed. In the present work, the classification network index used to evaluate the cDNN performance is the Matthews Correlation Coefficient (MCC). The latter is considered as one of the best "single-number" performance indexes, deriving directly from the associated Confusion Matrix (CM) [36]. The MCC can be computed as:

$$MCC = \frac{TP * TN - FP * FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{13}$$

Where:

- TP are the true positives.
- TN are the true negatives.
- FP are the false positives.
- FN are the false negatives.

The MCC ranges from -1 (the model is in complete opposition with respect to the observations) to 1 (perfect classification); in the case of

Table 2
Fixed-parameters for the cDNN and the rDNN.

Fixed-parameter	cDNN	rDNN
Number of epochs	250	100
Activation function (internal layers)	ReLU	ReLU
Activation function (output layer)	Sigmoid	Linear
Learning algorithm	Adam	Adam
Standardization method	Batch-Normalization	Batch-Normalization & Drop-Out

null MCC, the model is considered to produce random predictions. Along with the MCC, the associated Confusion-Matrix is also considered for an even more complete evaluation.

3.5. Performance index of the rDNN

Since the rDNN is intended to replicate the results of the DP algorithm, the widely used Coefficient of Determination (CoD) R^2 is selected to monitor the fitting goodness of the DNNs-PM predictions with respect to DP-based targets [37]. The CoD is calculated through:

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}} \tag{14}$$

With:

- $SS_{tot} = \sum_i (y_i - \bar{y})^2$
- $SS_{res} = \sum_i (y_i - f_i)^2$
- $\bar{y}$: target values average
- y_i: true value
- f_i: predicted value

The CoD ranges from infinitely negative number to 1: a baseline model predicting the average value of the distribution exhibits a null R^2; a perfect fit of the model scores an R^2 equal to 1, while negative values typically account for models worse than the baseline one [38].

Moreover, the Root Mean Squared Error (RMSE) is also monitored throughout the simulations as it is selected to be the loss-function of the rDNN. The RMSE is calculated through:

$$RMSE = \sqrt[2]{\frac{\sum_i \left(f_i - y_i\right)^2}{n}} \tag{15}$$

where n is the number of total predictions. Notice that the RMSE is expressed in g/km since these latter are the units of both target and predicted values. The RMSE values range from 0 (perfect predictions) to infinitely large values (predictions infinitely distant from target values).

3.6. Automatic search tool

It is now possible to introduce the AST procedure employed to select the best hyper-parameters' combination. Each tested hyper-parameters' combination is referred hereafter as "trial". Particularly, several combinations of hyper-parameters are randomly selected (i.e. Random-Search) and their performance are assessed for using an 8-fold Cross-Validation. Briefly, the dataset resulting from the merging between the training and validation sets is split into 8 sub-sets, alternatively treated as the validation-set. The task-specific performance index resulting from the average of the 8 validation splits is considered as the desired performance for the entire trial. In this manner, the same importance is given to each validation split, leading to a more robust estimation of the real performances of a given trial. The performances are then compared using a task-specific algorithm and the most promising set of hyper-parameters is selected.

3.6.1. Selection of the cDNN hyper-parameters

The cDNN hyper-parameters considered for the tuning operation are:

- Learning rate: logarithmic variation with base 10.
- Number of hidden layers.
- Neurons at first layer.
- L2 regularizer parameter: logarithmic distribution with base 10.
- Batch size: logarithmic distribution with base 2.

The number of neurons featuring the hidden layers should also be considered as an additional hyper-parameter to be tuned. Nevertheless, such a value has been computed through the following equation in order to bound the hyperparameters' space:

$$L_i = \frac{L_1}{2^{(i-1)}} \tag{16}$$

where L_i is the i-th hidden layer's number of neurons, and L_1 is the number of neurons of the first hidden layer. The resulting hyper-parameters' space is therefore a 5-dimensional one. Notice that the range in which each hyper-parameter varies is user-defined. Moreover, even though some of them (e.g. learning rate and L2 regularizing parameter) could theoretically exhibit a continuous distribution (i.e. infinite number of points in the variation range) a fixed number of points is considered in this study. Each axis of the space, namely each hyper-parameter range of variation, is divided into 3 equal sectors. Such a procedure generates 3^5 (243) sub-spaces. The logic is characterized by 4 steps:

1 A Random-Search in the whole hyper-parameters' space with a user-defined number of trials.
2 An 8-fold Cross-Validation (Section 3.6) of each trials. Whenever a trial is found to exhibit an MCC higher than a threshold value of 0.95, the sub-space in which the related hyper-parameters' combination is located is selected for the second step. If less than three sub-spaces are selected, the best three are chosen.
3 A Random-Search (same number of trials as the first step) in each selected sub-space.
4 An 8-fold Cross-Validation of each trials. The trial showing the higher MCC is considered as the best hyper-parameters' combination and chosen as the final one.

3.6.2. Selection of the rDNN hyper-parameters

A slightly different set of hyper-parameters has been considered for the tuning operation of the rDNN:

- Learning rate: logarithmic distribution with base 10.
- Number of hidden layers.
- Neurons at first layer.
- Batch size: logarithmic distribution with base 2.
- Type of weights initialization: choosing between Glorot uniform, Glorot normal, Random uniform, Random normal and truncated normal.

Once again, the number of neurons involved in the hidden layers is computed through Eq. (16). The tuning procedure of the AST is performed beginning with a Random-Search during which the R^2 value is monitored. The R^2 value is the result of the 8-fold Cross-Validation (Section 3.6). Whenever a trial shows an R^2 higher than a threshold value of 0.75, a second verification step is activated:

1 If any hyper-parameters combination has still been selected, the value of the loss function (RMSE, Section 3.5) obtained at the end of the Cross-Validation is stored and the combination is temporarily selected as the best one.

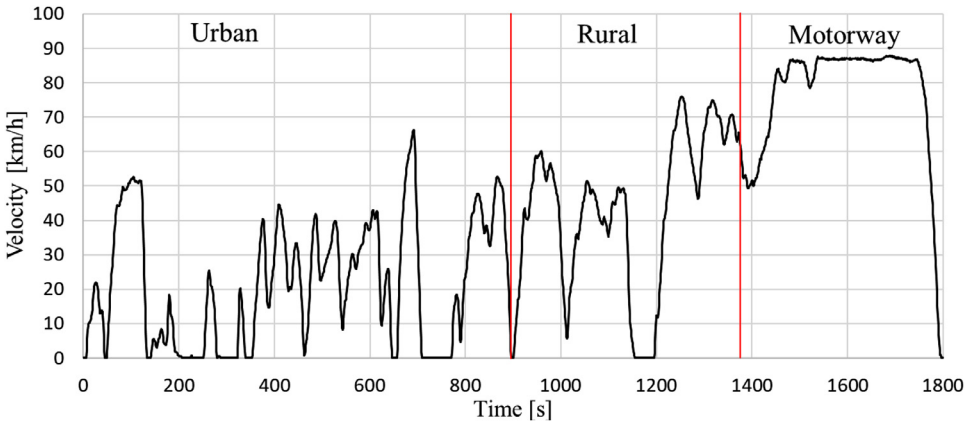


Fig. 3. World Harmonized Vehicle Cycle (WHVC) velocity profile.

Table 3
Main specifications of the baseline vehicle and its main components.

Vehicle	
Vehicle class	Heavy-duty
Vehicle weight [kg]	7500
Number of wheels [#]	6
ICE	
Displacement [L]	4.5
Rated Power [kW]	150
Maximum Torque [Nm]	800 @ 1400 RPM
MG	
Rated Power [kW]	125
Maximum Torque in traction [Nm]	300
Maximum Torque in braking [Nm]	-300

2 If a hyper-parameters combination has already been selected, a new combination can force the already selected combination out only in case of a lower value of the loss function.

Through the AST procedure for the rDNN, a combination is not selected unless the threshold R^2 value is reached; it is therefore possible that no combination is selected at the end of the tuning process. For such conditions, the AST procedure is repeated searching for a suitable combination of hyper-parameters: a new random-search is started and hence followed by steps 1 and 2.

4. Results

The DNNs-PM pipeline has been applied to the prediction of both HEV layout feasibility (classification task) and tank-to-wheel CO2 emissions (regression task) identified by the DP for a heavy-duty vehicle. The main specifications of the baseline vehicle used for modelling the HEV parallel architectures are reported in Table 3 whereas the driving cycle velocity profile of the World Harmonized Vehicle Cycle considered for the DP optimization is illustrated in Fig. 3. For the sake of clarity, additional information about the ICE and the MG were not presented for both for confidentiality reasons and because not relevant for the target of the present research.

The performance of the DNNs-PM has been evaluated following two steps: the evaluation of the effective model learning capability over the dataset of the P4 architecture, namely P4-dataset, and the comparison of the model generalization capability when tested to different HEV architectures (i.e. P2 and P3). Such an evaluation approach has been first applied to the cDNN stand-alone, then it has been extended to the entire pipeline. The study of the results produced on a single dataset is thought to be useful for understanding the DNNs operation, whereas the comparison on different datasets is mandatory to highlight the real generalization capability of the model when tested under various conditions.

The outcomes of 46 independent simulations (28 simulations for the cDNN and 18 for the pipeline) have been post-processed in order to evaluate the realistic potentials of the DNNs-PM and are presented in the next sections. First, the assessment of the cDNN performance is achieved by a preliminary simulation with a fixed t/t split so as to verify the network consistency of the learning process. Then, the results obtained by 24 simulations are presented in order to thoroughly analyse the performance variation of the model when changes into the training-set's numerosity are introduced: multiple tests (6) are performed so as to evaluate the generalization performance of the model under several random training, validation and test sets (Section 3.2) for 4 t/t ratios (60-40, 70-30, 80-20 and 90-10). Consistently, no bias in the model learning process is introduced. Once the evaluation of the model robustness is completed for a single HEV architecture, a comparison of the results produced by the model when applied to the three architectures' datasets (i.e. P2, P3 and P4 alternatively) is carried out under slightly different conditions. In fact, for such comparison the hyper-parameters' space has been enlarged and the number of trials (i.e. the number of tested hyper-parameters' combinations) has been increased. In this manner, the cDNN is allowed to deeply search in a wider numerical space increasing the probability of detecting an optimal hyper-parameters setup. The latter operation is made possible by the automatic search tool presented in Section 3.6. A single trial has been performed with a fixed t/t ratio comparing the MCC over the three architectures. As far as the performance of the entire pipeline are concerned, the assessment of its prediction capability is presented through the results of 18 simulations carried out over the P2, P3 and P4 datasets alternatively (i.e. 6 tests for each architecture).

As a final step, a detailed analysis of cDNN misclassified false positive layouts has been performed and presented to the reader.

4.1. Investigation on the cDNN performance

As a first step, the consistency of the cDNN's learning process is assessed. Particularly, the learning curves (i.e. the trend of the performance index during the training phase) are analysed so as to ensure the absence of overfitting or biased behaviours.

Recalling Section 3.2, a random selection of the training, validation and test sets is asked to the DNN before entering the training process, in order to avoid a biased network operation caused by a fixed split. Consistently, a diversification of the simulation setup is ensured to the DNNs-PM and an unbiased behaviour is promoted.

Within this section, the results produced by one simulation performed with the cDNN on the P4-dataset is presented as a qualitative example of the cDNN achievable outcomes. Twenty hyper-parameters' combinations are included for each random search and cross validation

Table 4
Setup of the investigation on the cDNN performance.

Hyper-parameter	Range of variation [-]
Learning rate	0.0002 – 0.02
Hidden Layers	1 – 4
Neurons first hidden layer	130 – 230
L2 regularizer	0.003 – 0.3
Batch size	16 – 128
Dataset split	**Number of samples [#]**
Training	1050
Validation	150
Testing	300

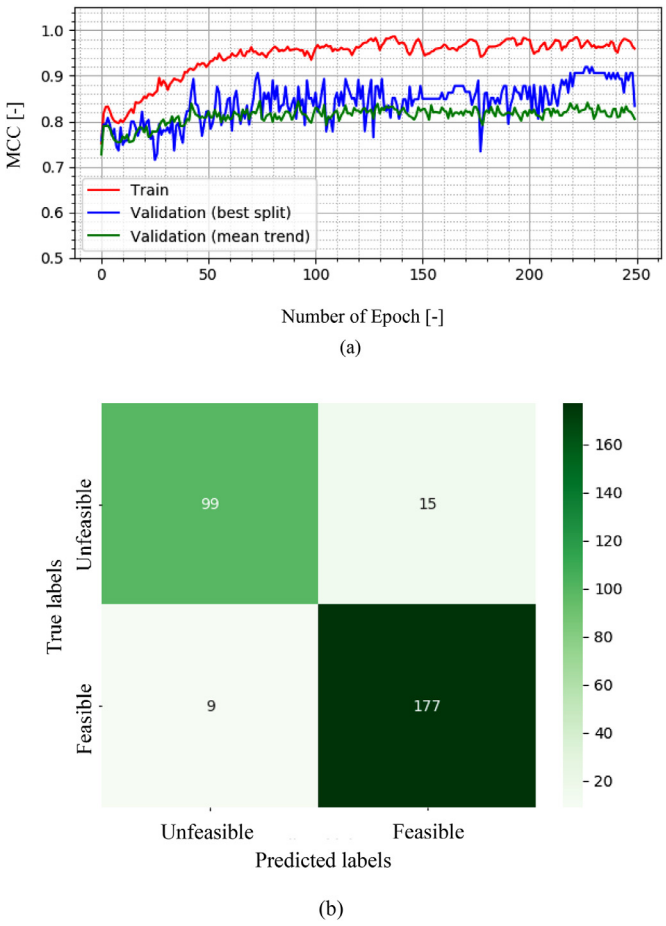


Fig. 4. MCC trend (a) and test-set's CM (b) for the P4-dataset simulation.

step of the abovementioned automatic search tool; 250 epochs are employed for the training phase and the t/t ratio is set to 80-20 (i.e. the fraction of training and validation samples with respect to the entire dataset numerosity is 80%). Notice that, due to the presence of the 8-fold cross-validation, the Train-to-Validation (t/v) ratio is 7-1. The complete set-up for the simulation is resumed in Table 4.

Two indices have been accounted for to monitor the cDNN performances through different trials: the MCC (see Section 3.4) and the test-set CM, which are presented in Fig. 4. The MCC (Fig. 4(a)) is monitored throughout the training phase (i.e. red curve) and for each split considered in the cross-validation procedure. Then, the average MCC trend is calculated for the cross-validation splits (i.e. green curve) and compared with the MCC trend of the best performing split (i.e. blue curve). For the sake of clarity, notice that the latter is tracing the cDNN's performance throughout the training process over one of the 8 validation

Table 5
Specifications of the experiments with the cDNN.

t/t ratio [%]	# of simulations	Number of samples [#]		
		Training-set	Validation-set	Test-set
60/40	6	788	112	600
70/30	6	919	131	450
80/20	6	1050	150	300
90/10	6	1181	169	150

split, namely the one that is found to achieve the highest MCC at the last training epoch. Promising results are highlighted by cDNN as the MCC increases along with the training epochs. The increasing trend of the learning curves is considered as a symptom of an healthy learning process, not affected by overfitting [39]. On the other hand, no warnings are shown by the curves noise, which can be considered as a direct consequence of the stochastic learning algorithm and of the batch-normalization technique (Section 3.3). From the noise perspective, the blue curve is clearly the most affected one: such a condition can be directly linked to the restricted number of layouts used for the model validation (i.e. validation-set of the best performing split). Consistently, the noise is smoothed-out by averaging the MCC on any split (green curve).

In addition, the complete CM for the P4 test-set is shown in Fig. 4(b), which reports the exact number of true positives, true negatives, false positives and false negatives. Given 1500 layouts in the P4-dataset and a 80-20 t/t ratio, 300 units are involved in the test-set. As a matter of fact, the sum of the CM outputs is equal to the test set size. An additional check of the robust performance of the cDNN is given by the test-set MCC computation through Eq. 13 (Section 3.4), resulting in 82.92%. Such a value is aligned with the final value of the average MCC trend (Fig. 4(a), green curve) proving the validity of the cross-validation performance prediction with respect to new and unknown data.

At this stage, the model generalization capability is evaluated based on the results obtained by 24 simulations in which changes into the training-set's numerosity are introduced by considering 4 t/t ratios (60-40, 70-30, 80-20 and 90-10). Multiple tests (6) are performed for each t/t ratio. In Table 5, a description of the datasets numerosity for the different t/t ratios is reported. The hyper-parameters' space and the number of examined hyper-parameters' combinations are kept the same as for the previous simulation, as well as the number of training epochs.

The main results obtained at the end of the 24 simulations are presented in Table 6. The actual MCC index resulting for each t/t ratio is reported for each trial along with the MCC average value (avg) and the standard deviation (std). A direct proportionality between classification performances and training-set's size arises: the MCC average value increases as the training-set is enlarged. Such a result can be considered as the most coherent outcome of a robust learning process. Consistently with the stochastic nature of the training set selection performed by the cDNN, not a fixed MCC is found for each trial, but a restrained standard value is highlighted. It is worth noticing that the 80/20 and 90/10 t/t ratios produce larger fluctuations with respect to the 60/40 and 70/30 ratios. This can be mainly linked to the reduced numerosity of the test-set: as the latter lowers, the relative influence of the single test prediction with respect to the overall prediction process increases; this finally results in more evident performance deficiencies/peaks throughout simulations in case strong discrepancies between training-set and test-set are found

4.2. Application of cDNN to different HEV architectures

Within this section, the results obtained by a different analysis aimed at highlighting the cDNN capability of producing accurate predictions when tested over different architectures' (i.e. P2, P3 and P4 alternatively) are presented.

Table 6
Main results of the cDNN on the P4-dataset.

t/t ratio [%]	I [%]	II [%]	III [%]	IV [%]	V [%]	VI [%]	avg [%]	std [%]
60/40	81.88	82.13	76.46	82.21	79.69	76.45	79.80	2.33
70/30	82.08	86.34	81.59	76.82	81.63	81.62	81.68	2.55
80/20	82.92	82.22	87.22	85.09	84.39	75.01	82.81	3.55
90/10	85.79	78.79	80.36	84.50	88.79	85.80	84.01	3.16

Table 7
Setup for the comparison of the cDNN performance over different HEV architectures.

Hyper-parameter	Range of variation [-]
Learning rate	0.00001 – 0.1
Hidden Layers	1 – 15
Neurons first hidden layer	20 – 300
L2 regularizer	0.0001 – 0.09
Batch size	16 – 516
Dataset portion	**Number of samples [#]**
Training	1181
Validation	169
Testing	150

Table 8
Comparison of the MCC obtained by the cDNN for the three compared datasets.

Architecture	MCC [%]
P2	96.98
P3	92.40
P4	91.51

Recalling Section 4, the authors chose to increase the number of trials and enlarge the hyper-parameters' space. Given that no a priori assumptions can be made about the effectiveness of the optimal hyper-parameters' combination found for the P4 architecture on different architectures, extending the number of realizable hyper-parameters' combinations to a larger extent can be considered so as to increase the probability of spotting an optimal (or near optimal) hyper-parameters' combination for each architecture. For the just mentioned reasons, an increase in the overall performance is expected.

For the present analysis, the t/t ratio has been set to 90/10 since it was found to be beneficial from the cDNN outcomes discussed in Section 4.1. The complete setup for the analysis is described in Table 7. The results of the comparison between the three architectures are resumed in Table 8. Notice these latter are the result of one simulation for each dataset (i.e. P2, P3 and P4 alternatively). The expected increase in the classification performance is clearly confirmed. Particularly to the P4-dataset, the obtained MCC is 91.51%, showing an increase of more than 2.7% with respect to the maximum MCC registered in Table 6 for the same t/t ratio. Anyhow, the cDNN performances on the classification of feasible P2 and P3 layouts are found to outperform the best performances found for the P4 dataset. In fact, the MCC obtained for the P2 and P3 datasets are roughly 97% and 92%, respectively. This phenomenon however has not been deepened in this study and could be grounds for future research. It is anyway believed to be related to the stochastic nature of the NNs.

4.3. Investigation of the pipeline performance

Once the robustness of the cDNN has been assessed for, the performance of the entire pipeline can be verified. Particularly, a first in-depth analysis of the results produced by the DNNs-PM when applied to the P4-datasets is performed. The generalization capability is hence evaluated comparing the outcomes on the P2, P3 and P4 datasets.

Table 9
Setup of the investigation on the pipeline performance.

Hyper-parameter	Range of variation [-]
Learning rate	0.002 – 0.1
Hidden Layers	1 – 6
Neurons first hidden layer	30 – 80
Batch size	8 – 64
Drop-out	0 – 0.5
Weights initialization	Xavier, random, truncated normal

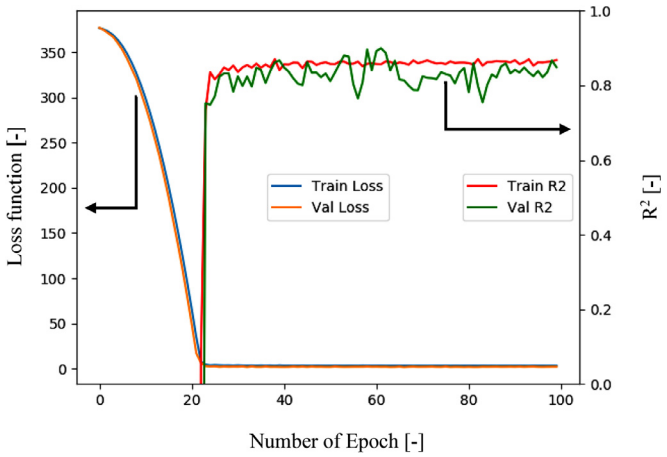


Fig. 5. Regression task performance indeces trend: R-square for the training-set (red line); R-square for the validation-set (green line); loss function for the training-set (blue line); loss function for the validation-set (orange line). The black arrows indicate the reference axis. (For interpretation of the colors in this figure legend, the reader is referred to the web version of this article.)

The rDNN is thought to be responsible for predicting the CO_2 of the only HEV layouts positively passing through the cDNN classification filter. Therefore, it is worth observing that the numerosity of the datasets inputted into the rDNN and the cDNN can differ. In fact, the amount of data read by the rDNN is equal to the amount of feasible samples outputted by the cDNN. However, the DNNs-PM is required to control the number of samples used to train and test the rDNN in order to avoid heavy discrepancies on the regression task between multiple tests. Therefore, preserving the feasible-to-unfeasible layouts proportion is required to the rDNN. For the present investigation, a two-to-one proportion has been guaranteed for each HEV dataset.

As for the cDNN, 6 tests are performed keeping the t/t ratio fixed at 90/10 over the P4-dataset; the hyper-parameters' setup used for the rDNN is resumed in Table 9.

Once again, two indices are monitored throughout the analysis, namely the RMSE (i.e. the loss function of the regression task) and the R2 value (Section 3.6.2). The trend of the two performance indicators through increasing number of epochs are referred to as rDNN's learning curves; these latter are reported in Fig. 5 for one of the 6 simulations as an example case.

Table 10
Main results of the rDNN on the P4-dataset.

	Ê		Max E		min E		R^2[-]	RMSE[g/km]
	Abs[g/km]	Rel[%]	Abs[g/km]	Rel[%]	Abs[g/km]	Rel[%]		
I	2.11	1	9.39	2	0.03	<1	0.923	2.50
II	0.92	<1	4.67	1	0.01	<1	0.988	1.25
III	1.49	<1	7.43	2	0.03	<1	0.972	1.77
IV	4.12	1	11.48	3	0.15	<1	0.672	4.61
V	1.28	<1	7.88	2	0.00	<1	0.963	1.93
VI	0.78	<1	3.13	1	0.01	<1	0.989	1.06
avg	1.78	<1	7.33	2	0.04	<1	0.918	2.19
std	1.13	<1	2.78	1	0.05	<1	0.112	1.18

The same considerations made for the classification task-related learning curves (Section 4.1) can be fortunately applied to the regression ones: the increasing trend of the R^2 index, both for training (red curve) and for validation sets (green curve), as well as the decreasing trend of loss function (blue and orange curves), are a promising sign of an effective, overfitting-free, learning process [39].

The CO_2 emissions predictions and target values (i.e. the DP-labelled outputs of the test-set) are reported in Fig. 6 for all simulations. The regression Average Errors (Ê), Maximum Errors (Max E) and Minimum Errors (min E) obtained are summarized in Table 10; it is worth noticing that relative errors of "<1" are referring to relative errors below 0.5%. As for the cDNN, the performance indexes are reported for each simulation along with the average value and the standard deviation.

The CO_2 emissions predicted by the pipeline are considered as comparable with respect to the target ones as an overall average relative error under 0.5% (Table 10) arises. Interestingly, the worst prediction (4^{th} simulation) leads to a restrained average relative error, equal to 1%.

Finally, the relative errors of the CO_2 predictions are shown in Fig. 7 for each test case. Few relevant outliers are found through the six simulations, a symptom of the rDNN robustness when several tests are considered. Particularly, the overall maximum absolute error is 3% (4^{th} simulation). Notice finally that no particular trend is found concerning the relative error dimensions. This can be considered proof of the absence of biased behaviour towards specific category of layouts.

As final step, a further test is conducted by applying the entire pipeline to the P2 and P3 datasets using the same t/t ratio (i.e. 90/10): 12 simulations (6 for each dataset) have been performed and their relative R^2 value and RMSE are reported in Table 11 together with the average, maximum and minimum errors. It is worth highlighting that the results concerning the P4-dataset are the same as those presented in Table 10. The relative errors falling very close to zero are reported as "<1". Robust performances can be observed for each of the tested HEV architecture. As far as the Δ is concerned, very small errors arise regardless of the HEV architecture as the maximum relative discrepancy is detected at 1 %. On the other hand, despite the fluctuations of the Max E are more pronounced, the maximum relative discrepancy never exceeds 4% (P2 and P3 datasets). At the same time, the min E are also very close to zero as quasi-perfect predictions are attained.

The highest R^2 is achieved when the DNNs-PM is applied to the P4-dataset; nevertheless, the indexes Δ, Max E and min E show larger values than the corresponding ones featuring the P2 and the P3 datasets. Given that the RMSE is considered as a more reliable index to assess for the real discrepancies between the predictions and target values, the performance of the entire pipeline can be evaluated as consistent with those produced by the cDNN on the P2 and P3 datasets.

4.4. False positives

Even if a perfect prediction is considered as the ideal scenario, a restrained error has to be assumed even for very performing neural networks. Within a classification task, the prediction error can turn into the generation of false positives (i.e. unfeasible layouts that are incorrectly classified as feasible) or false negatives (i.e. feasible layouts that are incorrectly classified as unfeasible). Regarding the latter, the only consequence for the present application would be the avoidance of the CO_2 prediction for few layouts. Such a scenario would not be dramatic from a design perspective since no misleading results could be produced and post-processed. Contrarily, a false positive layout could affect the data post-processing: if an unfeasible layout is classified as feasible, a completely unrealistic CO_2 prediction would occur.

This said, a final analysis has been carried out in order to evaluate the rDNN behavior in case of a cDNN misclassification which leads to a given amount of false positive layouts. The cDNN false positive classifications resulting from the 18 simulations of Section 4.3 have been extracted and fed to the rDNN; the CO_2 emissions values predicted by the rDNN for those layouts are reported in Table 12 in increasing order. The minimum predicted CO_2 values related to the true positive layouts are reported in the table for convenience: given that the minimum emissions are considered as a guidance for a design optimization procedure, monitoring that the predicted CO_2 values of the false positive layouts do not fall below the minimum of the true positive ones is mandatory.

The results show that 86 unfeasible layouts out of 2700 total layouts (of which roughly 900 are unfeasible) are incorrectly marked as feasible, that is 3.1% (roughly 9.6% of the unfeasible layouts); moreover, only 2 layouts out of the 86 misclassified unfeasible layouts (2.3%) show a predicted CO_2 value under the minimum predicted one. In Table 12, they are denoted with an asterisk. Overall, the false positive layouts for which a CO_2 value is lower than the minimum true positive-related CO_2 value amount to the 0.07% of the total analysed layouts (roughly the 0.22% of the total unfeasible layouts). Since the identification of the minimum emissions region is typically preferred to the actual minimum emission value, the behaviour of the DNNs-PM in case of false positive layouts is considered as a promising outcome for the present application.

4.5. DNNs-PM for HEV design optimizations

As a last step, the discussion of the DNNs-PM integration within a possible design operation for HEV fleets is presented. Specifically, the tool capability of spotting realistic regions within the design space featured by low CO_2 emissions is assessed for with respect to the results achieved by the DP algorithm. Assessing for a good response of the DNNs-PM would turn into the confirmation of the possibility to involve the model within a real-world energy-oriented design optimization for HEVs.

For the sake of conciseness, the main outcomes produced by the DNNs-PM when applied to the P3 architecture alone are accounted for. In Fig. 8, the CO_2 emissions levels obtained for the layouts from the testing set (see Table 7) are depicted both for estimation through DP and DNNs-PM. More specifically, the CO_2 emissions of Fig. 8-a(1) and Fig. 8-a(2) are referring to the design domain of *EngDispl* and *MGPower* whereas *EngDispl* and *PEratio* are employed to represent the CO_2 surfaces in Fig. 8-b(1) and Fig. 8-b(2). It is worth highlighting that the authors' choice of representing the CO_2 emissions with respect to *EngDispl, MGPower, PEratio* is arbitrary and is meant to exemplify the logic. Hence,

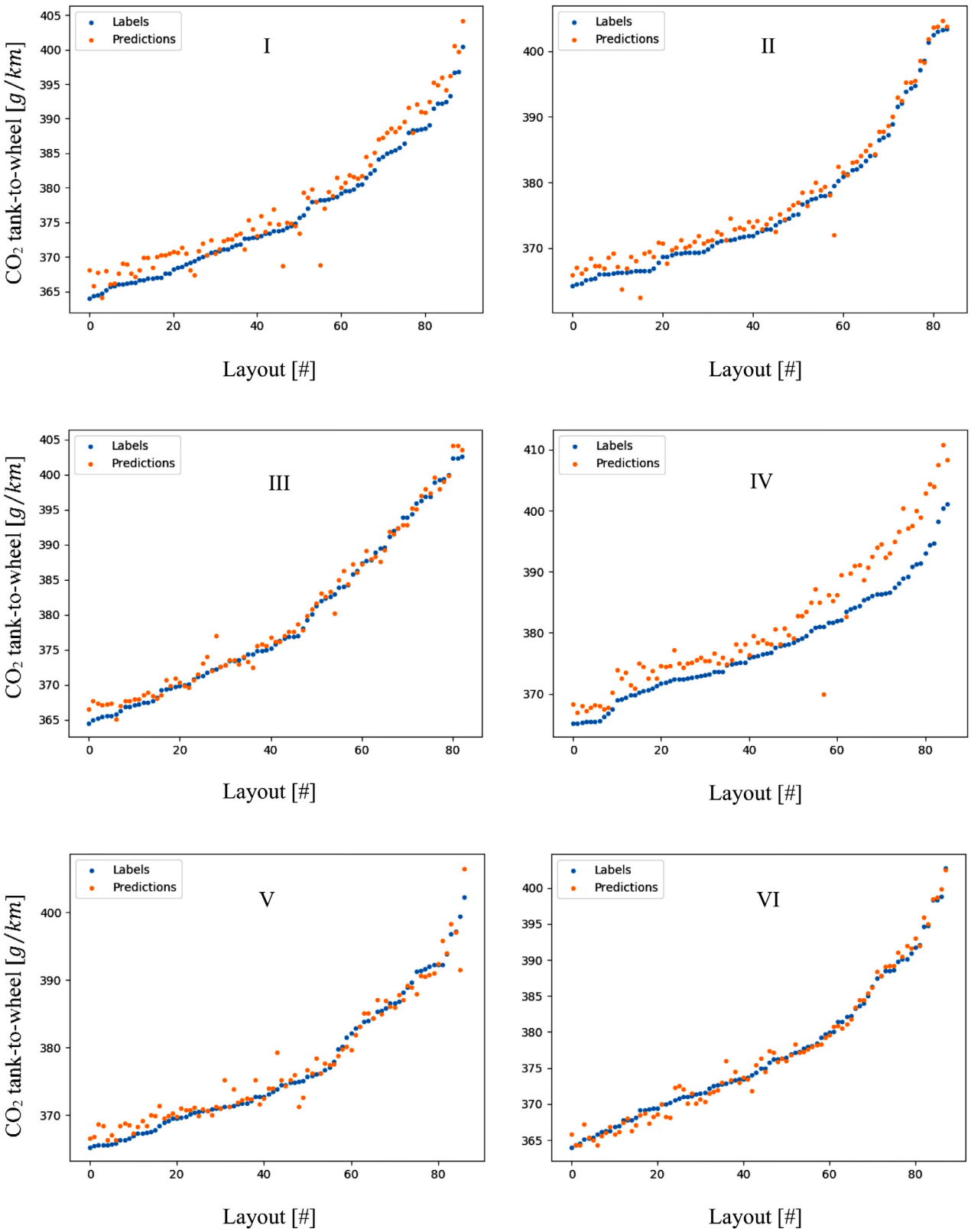


Fig. 6. Predicted emissions (in orange) and DP-computed emissions (in blue) for the six simulations. (For interpretation of the colors in this figure legend, the reader is referred to the web version of this article.)

Table 11
Comparison of the main results obtained by the entire pipeline for the three datasets.

Dataset	R^2 value [-]							
	I	II	III	IV	V	VI	avg	std
P2	0.911	0.536	0.788	0.806	0.879	0.716	0.773	0.114
P3	0.937	0.947	0.719	0.955	0.879	0.953	0.899	0.078
P4	0.923	0.988	0.972	0.672	0.963	0.989	0.918	0.112
Dataset	RMSE [g/km]							
	I	II	III	IV	V	VI	avg	std
P2	0.81	1.36	1.17	1.00	1.14	1.45	1.16	0.21
P3	0.84	0.93	1.33	0.75	1.11	0.77	0.95	0.21
P4	2.50	1.25	1.77	4.61	1.93	1.06	2.19	1.18
Dataset	Ê [g/km - %]							
	I	II	III	IV	V	VI	avg	std
P2	0.55 - <1	1.05 - <1	0.77 / <1	0.68 / <1	0.72 / <1	1.13 / <1	0.82 / <1	0.19 / <1
P3	0.59 - <1	0.64 - <1	0.99 / <1	0.56 / <1	0.92 / <1	0.54 / <1	0.71 / <1	0.17 / <1
P4	2.11 - 1	0.92 - <1	1.49 / <1	4.12 / 1	1.28 / <1	0.78 / <1	1.78 / <1	1.05 / <1
Dataset	Max E [g/km - %]							
	I	II	III	IV	V	VI	avg	std
P2	4.36 - 1	15.14 - 4	12.60 - 4	5.47 - 2	5.55 - 2	7.31 - 2	8.41 - 3	3.73 - 1
P3	2.83 - 1	4.72 - 1	13.79 - 4	2.83 - 1	2.38 - 1	3.20 - 1	4.96 - 2	3.72 - 1
P4	9.39 - 2	4.67 - 1	7.43 - 2	11.48 - 3	7.88 - 2	3.13 - 1	7.33 - 2	2.58 - 1
Dataset	min E [g/km - %]							
	I	II	III	IV	V	VI	avg	std
P2	0.00 - <1	0.01 - <1	0.00 - <1	0.01 - <1	0.01 - <1	0.00 - <1	0.01 - <1	0.01 - <1
P3	0.01 - <1	0.03 - <1	0.01 - <1	0.00 - <1	0.06 - <1	0.00 - <1	0.02 - <1	0.02 - <1
P4	0.03 - <1	0.01 - <1	0.03 - <1	0.15 - <1	0.00 - <1	0.01 - <1	0.04 - <1	0.05 - <1

Table 12
CO_2 prediction related to the false positive layouts for the three datasets.

P2-dataset						
	I	II	III	IV	V	VI
$CO_{2,min}$ [g/km]	331.62	331.53	331.97	332.19	331.52	330.13
CO_2	338.93	-	338.28	346.66	345.11	342.37
[g/km]	365.86	-	-	362.53	349.64	343.44
	-	-	-	441.07	377.57	349.68
	-	-	-	-	400.52	351.10
	-	-	-	-	-	357.13
	-	-	-	-	-	369.83
	-	-	-	-	-	396.27
P3-dataset						
	I	II	III	IV	V	VI
$CO_{2,min}$ [g/km]	333.81	334.09	332.89	333.38	334.91	333.57
CO_2	340.30	338.83	341.00	347.10	336.12	347.86
[g/km]	343.99	355.30	344.03	347.47	338.42	-
	358.06	362.79	-	352.92	339.19	-
	387.31	-	-	-	342.44	-
P4-dataset						
	I	II	III	IV	V	VI
$CO_{2,min}$ [g/km]	364.07	365.14	362.41	366.93	366.32	364.30
CO_2	363.15*	369.63	367.24	367.26	363.03*	368.32
[g/km]	367.19	382.43	373.92	367.37	367.97	369.95
	368.06	389.17	379.15	372.64	368.08	372.43
	372.25	-	382.74	377.32	368.20	374.76
	375.38	-	383.56	379.55	368.36	376.72
	377.73	-	385.94	379.98	369.30	379.11
	379.25	-	388.03	380.59	370.53	380.38
	380.48	-	-	382.85	372.37	380.98
	381.08	-	-	391.06	383.40	382.41
	381.30	-	-	393.05	-	388.21
	381.77	-	-	395.80	-	390.32

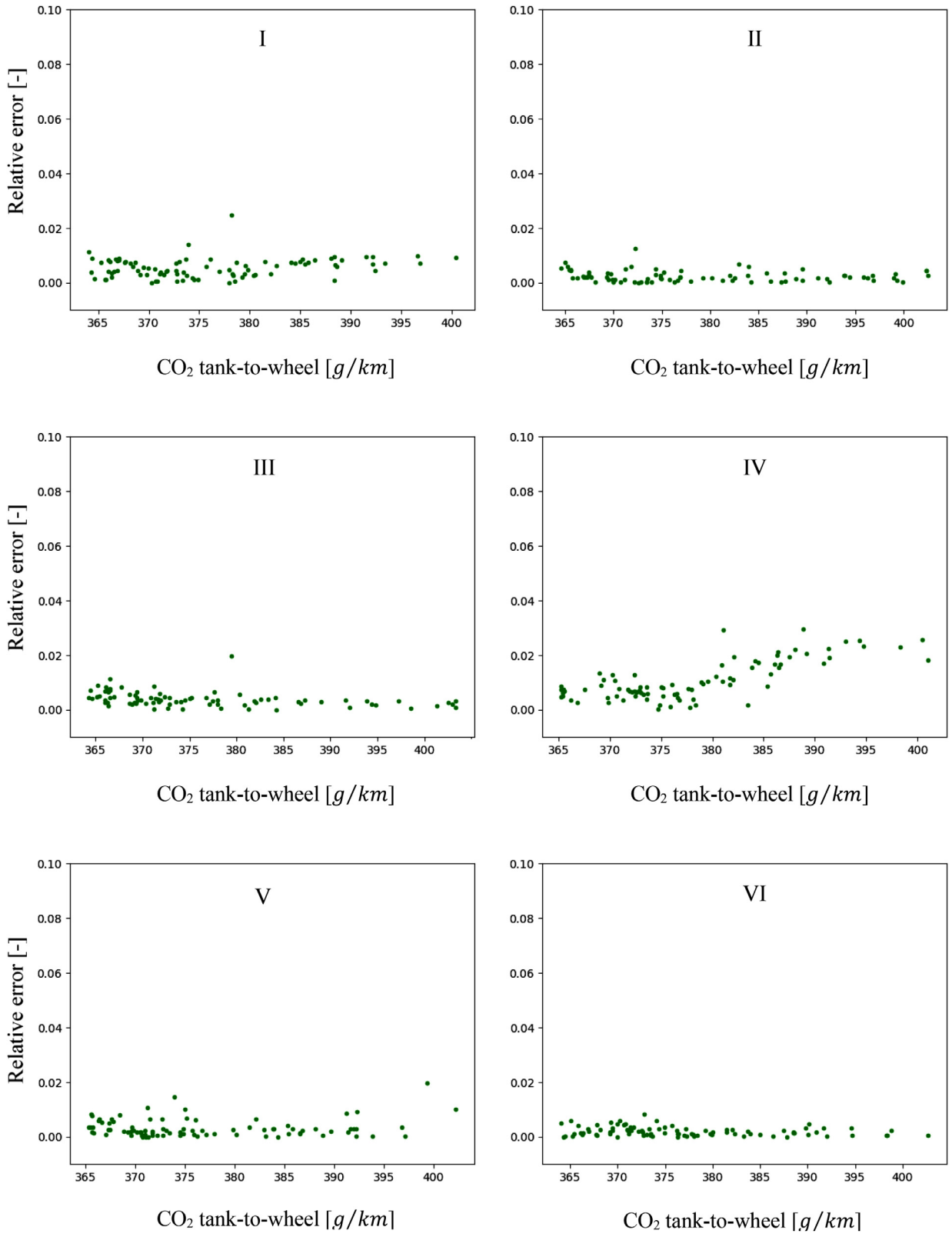


Fig. 7. Relative errors obtained for the six simulations.

the same analysis could be easily carried out for any other design parameter considered in the paper.

Two main considerations arise from the results of Fig. 8: the DNNs-PM is effectively capable of reproducing the shape of the CO_2 surfaces independently from the considered design variables; the minimum CO_2 emissions region can be defined by means of the DNNs-PM and corresponds to the same region obtained by DP. Even though the CO_2 surfaces are fragmented due to the sparse nature of the test-set (i.e. only a portion of the entire dataset is considered), the engine downsizing appears to be the more influencing design parameter to achieve CO_2 reductions with

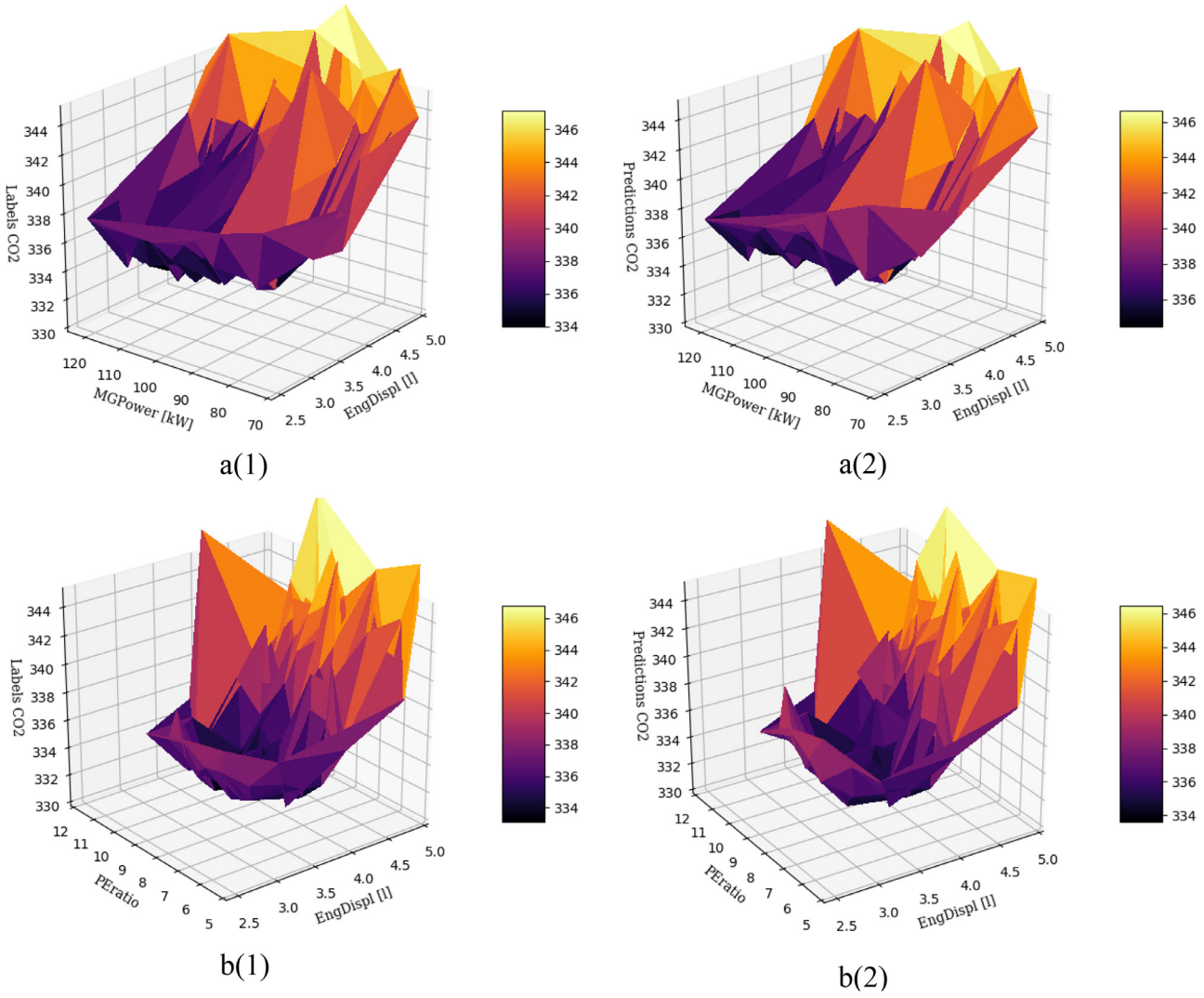


Fig. 8. Low CO_2 emission regions analysis; a: CO_2 vs EngDispl/MGPower; b: CO_2 vs EngDispl/PEratio; (1): DP results; (2): DNNs-PM predictions.

the HEV P3 architecture. In fact, layouts represented by engine displacements between 3.2 and 4.0 liters lead to lower CO_2 emissions regardless of the motor/generator and battery sizing. Moreover, higher MG peak powers relate to low emission zones. Such a result is consistent with the physics behind the optimal control trajectories to be found by the DP. As a matter of fact, once the feasibility of a given set of HEV layouts on a specific driving mission is assessed for during a preliminary analysis, the layouts with the higher degrees of hybridization should lead lower CO_2 emissions. In fact, the controller would reduce the share of exploitation of the thermal engine in favor of an increase into the share of pure electric and power-split modes.

5. Conclusions

In the present paper, the capability of a pipeline made up of two deep neural networks is assessed for. The latter has been considered as an innovative solution for drastically reducing the computational time required during a design optimization procedure of several hybrid electric vehicles architectures. In fact, the pipeline has been conceived as a tool for predicting the feasibility of hybrid layouts as well as the tank-to-wheel CO_2 emissions.

The main outcomes of the study are:

- The deep neural networks proved to be capable of catching the strongly non-linear correlations between components' specifications of different hybrid electric vehicles and predicting both the feasibility of the layouts (classification task) and the tank-to-wheel CO_2 emissions (regression task).
- The performance of the classification deep neural network increased together with the number of layouts employed in the training procedure; the best results have been found for the P2-dataset.
- The regression deep neural network showed CO_2 predictions comparable to the target values with average error lower than 0.5% for any feasible layout outputted by the classification deep neural network.
- The capability of the model to limit the generation of false positive layouts. In fact, only the 0.22% of the total unfeasible layouts are misclassified as false positive layouts, hence not affecting the design optimization procedure.
- The presented model can be embedded into a design optimization operation for hybrid electric vehicles fleets since it showed the capability of identifying low CO_2 emissions regions comparable with those obtained by a global optimization algorithm.

Considering the abovementioned results, further research steps will involve the prediction of indicators about vehicle drivability and total cost of ownership. Furthermore, new hybrid architectures will be tested in order to enlarge the prediction possibility over a wider design domain and different driving missions will be integrated so that cycle-depending indexes could be analysed.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] David Huang K, Quang KV, Tseng KT. Study of the effect of contraction of cross-sectional area on flow energy merger in hybrid pneumatic power system. Appl Energy 2009;86:2171–82. doi:10.1016/j.apenergy.2009.03.002.

[2] Un-Noor F, Padmanaban S, Mihet-Popa L, Mollah MN, Hossain E. A comprehensive study of key electric vehicle (EV) components, technologies, challenges, impacts, and future direction of development. Energies 2017;10. doi:10.3390/en10081217.

[3] Weiss M, Zerfass A, Helmers E. Fully electric and plug-in hybrid cars - an analysis of learning rates, user costs, and costs for mitigating CO 2 and air pollutant emissions. J Clean Prod 2019;212:1478–89. doi:10.1016/j.jclepro.2018.12.019.

[4] Hannan MA, Azidin FA, Mohamed A. Hybrid electric vehicles and their challenges: a review. Renew Sustain Energy Rev 2014;29:135–50. doi:10.1016/j.rser.2013.08.097.

[5] Sharaf OZ, Orhan MF. An overview of fuel cell technology: fundamentals and applications. Renew Sustain Energy Rev 2014;32:810–53. doi:10.1016/j.rser.2014.01.012.

[6] Finesso R, Spessa E, Venditti M. Layout design and energetic analysis of a complex diesel parallel hybrid electric vehicle. Appl Energy 2014;134:573–88. doi:10.1016/j.apenergy.2014.08.007.

[7] Anselma PG, Belingardi G, Falai A, Maino C, Miretti F, Misul D, et al. Comparing parallel hybrid electric vehicle powertrains for real-world driving. In: 2019 AEIT International Conference of Electrical and Electronic Technologies for Automotive, 2019; 2019. p. 1–6. doi:10.23919/EETA.2019.8804609.

[8] Sioshansi R, Denholm P. Emissions impacts and benefits of plug-in hybrid electric vehicles and vehicle-to-grid services. Environ Sci Technol 2009;43:1199–204. doi:10.1021/es802324j.

[9] Çağatay Bayindir K, Gözüküçük MA, Teke A. A comprehensive overview of hybrid electric vehicle: Powertrain configurations, powertrain control techniques and electronic control units. Energy Convers Manag 2011;52:1305–13. doi:10.1016/j.enconman.2010.09.028.

[10] Lanzarotto D, Marchesoni M, Passalacqua M, Prato AP, Repetto M. Overview of different hybrid vehicle architectures. IFAC-PapersOnLine 2018;51:218–22. doi:10.1016/j.ifacol.2018.07.036.

[11] Jiang Q, Ossart F, Marchand C. Comparative study of real-time HEV energy management strategies. IEEE Trans Veh Technol 2017;66:10875–88. doi:10.1109/TVT.2017.2727069.

[12] Zhang F, Hu X, Langari R, Cao D. Energy management strategies of connected HEVs and PHEVs: recent progress and outlook. Prog Energy Combust Sci 2019;73:235–56. doi:10.1016/j.pecs.2019.04.002.

[13] Lin CC, Peng H, Grizzle JW, Kang JM. Power management strategy for a parallel hybrid electric truck. IEEE Trans Control Syst Technol 2003;11:839–49. doi:10.1109/TCST.2003.815606.

[14] Yang C, Jiao X, Li L, Zhang Y, Zhang L, Song J. Robust coordinated control for hybrid electric bus with single-shaft parallel hybrid powertrain. IET Control Theory Appl 2015;9:270–82. doi:10.1049/iet-cta.2014.0321.

[15] Rizzoni G, Pisu P, Calo E. Control strategies for parallel hybrid electric vehicles. IFAC Proc Vol 2004;37:495–500. doi:10.1016/s1474-6670(17)30392-0.

[16] Carbonell JG, Michalski RS, Mitchell TM. An Overview of machine learning. Machine Learning Symbolic Computation. Michalski RS, Carbonell JG, Mitchell TM, editors. Springer; 1983. doi:10.1007/978-3-662-12405-5_1.

[17] Garbade M. J. Regression versus classification machine learning: what's the difference?. https://medium.com/quick-code/regression-versus-classification-machine-learning-whats-the-difference-345c56dd15f7; 2018 [accessed 05 January 2021].

[18] Kotsiantis SB, Zaharakis ID, Pintelas PE. Machine learning: a review of classification and combining techniques. Artif Intell Rev 2006;26:159–90. doi:10.1007/s10462-007-9052-3.

[19] Bertsekas DP. Distributed dynamic programming. Proc IEEE Conf Decis Control 1981;1:774–9. doi:10.1109/cdc.1981.269319.

[20] Joshi S, Shenoy D, Vibhudendra Simha GG, Rrashmi PL, Venugopal KR, Patnaik LM. Classification of Alzheimer's disease and Parkinson's disease by using machine learning and neural network methods. In: ICMLC 2010 - International Conference on Machine Learning and Soft Computing; 2010. p. 218–22. doi:10.1109/ICMLC.2010.45.

[21] Abe S. Overview of neural networks. Neural Netw Fuzzy Syst 1997:1–5. doi:10.1007/978-1-4615-6253-5_1.

[22] Hailesilassie T. Rule extraction algorithm for deep neural networks: a review. International Journal of Computer Science and Information Security 2016;14(7):371–81 arXiv:1610.05267 [cs.CV].

[23] Koutsoukas A, Monaghan KJ, Li X, Huan J. Deep-learning: Investigating deep neural networks hyper-parameters and comparison of performance to shallow methods for modeling bioactivity data. J Cheminform 2017;9:1–13. doi:10.1186/s13321-017-0226-y.

[24] Korotcov A, Tkachenko V, Russo DP, Ekins S. Comparison of deep learning with multiple machine learning methods and metrics using diverse drug discovery data sets. Mol Pharm 2017;14:4462–75. doi:10.1021/acs.molpharmaceut.7b00578.

[25] Giunta AA, Wojtkiewicz SF, Eldred MS. Overview of modern design of experiments methods for computational simulations. 41st Aerospace Sciences Meeting and Exhibit; 2003. doi:102514/62003-649.

[26] D'Ambrosio S, Finesso R, Spessa E. Calculation of mass emissions, oxygen mass fraction and thermal capacity of the inducted charge in SI and diesel engines from exhaust and intake gas analysis. Fuel 2011;90:152–66. doi:10.1016/j.fuel.2010.08.025.

[27] Finesso R, Spessa E, Venditti M. Cost-optimized design of a dual-mode diesel parallel hybrid electric vehicle for several driving missions and market scenarios. Appl Energy 2016;177:366–83. doi:10.1016/j.apenergy.2016.05.080.

[28] Burhenne S, Jacob D, Henze GP. Sampling based on sobol' sequences for monte carlo techniques applied to building simulations. In: Proceedings of the International Conference Building Simulation; 2011. p. 1816–23.

[29] Ladislav K, William J. Computational investigations of low-discrepancy sequences. ACM Trans Math Softw 1995;266:294 -23. doi:10.1145/264029.264064.

[30] Bellman RE, Lee ES. History and development of dynamic programming. IEEE Control Syst Mag 1984;4:24–8. doi:10.1109/MCS.1984.1104824.

[31] Singh D, Singh B. Investigating the impact of data normalization on classification performance. Appl Soft Comput 2020;97:105524. doi:10.1016/j.asoc.2019.105524.

[32] Nayak SC, Misra BB, Behera HS. Impact of data normalization on stock index forecasting. Int J Comput Inf Syst Ind Manag Appl 2014;6:257–69.

[33] Shen X, Gong X, Cai Y, Guo Y, Tu J, Li H, et al. Normalization and integration of large-scale metabolomics data using support vector regression. Metabolomics 2016;12. doi:10.1007/s11306-016-1026-5.

[34] Bhanja S., Das A. Impact of data normalization on deep neural network for time series forecasting. ArXiv 2018. arXiv:1812.05519v2.

[35] Sharma S. Epoch vs Batch Size vs Iterations. https://towardsdatascience.com/epoch-vs-iterations-vs-batch-size-4dfb9c7ce9c9; 2017 [accessed 29 December 2020].

[36] Chicco D, Jurman G. The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation. BMC Genomics 2020;21:1–14. doi:10.1186/s12864-019-6413-7.

[37] Ozer DJ. Correlation and the coefficient of determination. Psychol Bull 1985;97:307–15. doi:10.1037/0033-2909.97.2.307.

[38] Cameron AC, Windmeijer FAG. An R-squared measure of goodness of fit for some common nonlinear regression models. J Econom 1997;77:329–42. doi:10.1016/s0304-4076(96)01818-0.

[39] Zhang C., Vinyals O., Munos R., Bengio S. A study on overfitting in deep reinforcement learning. ArXiv 2018. arXiv:1804.06893v2.