Computers and Chemical Engineering 173 (2023) 108222

Contents lists available at ScienceDirect

Computers and Chemical Engineering

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

A deep learning-based predictive controller for the optimal charging of a lithium-ion cell with non-measurable states

Andrea Pozzi [a,*], Scott Moura [b], Daniele Toti [a]

[a] *Catholic University of Sacred Heart, Faculty of Mathematical, Physical and Natural Sciences, Via della Garzetta 48, 25133 Brescia, Italy*
[b] *University of California, Berkeley, 625 Davis Hall, Berkeley, CA 94720, USA*

ARTICLE INFO

Keywords:
Machine learning
Model predictive control
Battery management systems
Computational complexity
Deep learning

ABSTRACT

Battery charging is a complex task, which needs to be addressed by a proper control methodology to find the highest charging current while guaranteeing safety. Among the different approaches, model predictive control appears particularly suitable due to its ability in dealing with nonlinear systems and constraints. However, its use in a realistic scenario is limited due to the high computational burden required by the online solution of an optimal control problem. A neural network-based algorithm is here proposed to significantly reduce the real-time computational effort by approximating the predictive control law. In addition, for the first time to the authors' knowledge, an adaptation of the proposed deep learning-based algorithm is presented for the case in which the battery's internal states are not measurable. The superiority of proposed methodology is highlighted in simulation by comparing it with a predictive controller coupled with a properly designed state observer.

1. Introduction

The management of Lithium-Ion batteries is a complex activity indeed, since it requires a number of often mutually conflicting considerations and strategies, including the definition of suitable current profiles for charging and discharging them and the application of mechanisms for preventing their performance from degrading with the passage of time; besides, their safety over the course of their lifespan must be guaranteed as well. It is therefore apparent that taking into account all of these elements is a challenge that needs to be tackled very carefully (Lu et al., 2013).

In industry, common practices revolve around the implementation of rule-based algorithms for managing this kind of batteries. One class of such algorithms is the one based on the so-called constant-current/constant-voltage principle, whose main purpose is to ensure a charging time for the batteries that is as practically useful as possible within the boundaries of voltage specified accordingly (Shen et al., 2012). These mechanisms, however, tend to apply a rather conservative approach and are usually not able to fully exploit the capability of batteries in terms of either speeding up their charging time or preventing negative consequences for their safety. The result is therefore often far from ideal: given the fixed nature of the constraints related to the voltage, such basic algorithms fall short in considering how batteries gradually deteriorate and how their inner characteristics get progressively altered as they get repeatedly charged and discharged for a significant number of times (Chaturvedi et al., 2010).

As a matter of fact, mechanisms that improve over these algorithms do exist and are meant to provide a better control strategy for the batteries. A notable technique in this regard is the Model Predictive Control (MPC) (Camacho and Alba, 2013), owning its success and subsequent wide adoption in the battery context (see for instance Klein et al. (2011), Zou et al. (2018), Torchio et al. (2015) and Pozzi et al. (2018b)) to its ability in managing multi-variable, nonlinear processes under constraints in terms of input and states. Nevertheless, MPC *per se* is not a silver bullet either: while it succeeds in showing a good performance during the simulations, its more practical use in real-world scenarios is generally hindered by being forced to deal with computational complexities, making it far less effective and efficient than it should in principle be. The core issue lies in the nature of the algorithm that tries, in real-time at each time step, to solve a constrained optimal control problem: when it is not able to computationally be on par with a fast sampling time of the control law, MPC, for all intents and purposes, becomes useless.

This has something to do with how MPC is implemented, usually via mostly linear, simplistic models that consider a limited number of states and a prediction horizon as limited as the former. This in general leads to a poor performance when the system to be controlled is particularly complex. There is of course a reason behind these implementation choices: the use of highly non-linear models within an optimization-based controller may lead to convergence issues while retrieving the optimal control action an therefore failure of the controller.

* Corresponding author.
E-mail address: andrea.pozzi@unicatt.it (A. Pozzi).

https://doi.org/10.1016/j.compchemeng.2023.108222
Received 31 July 2022; Received in revised form 2 March 2023; Accepted 8 March 2023
Available online 15 March 2023

In order to try and come up with a solution that could enable scientists to get out of this conundrum, a specific type of MPC, namely an "explicit" MPC (Alessio and Bemporad, 2009), has been brought to the attention of literature and industry, based on the concept of reducing real-time operations to merely evaluating a simple function. In principle, this explicit MPC should have a reduced computational cost, since it pre-calculates the optimal control action in terms of a piece-wise function of the state and reference vectors, mainly needing, in real-time, to detect a region in which the states are located. However, the detection of such a region may be very computationally-demanding in the presence of a large number of constraints and a large horizon of predictions, even beyond acceptable levels (Karg and Lucia, 2020).

As a consequence, several works in literature (for instance, Parisini and Zoppoli, 1995; Bemporad et al., 2011 and Csekő et al., 2015) have tried to address the problem of abating this computational cost. What the majority of them proposes is to carry out an approximation of the control law; in this regard, the use of machine learning models has recently caught the attention of researchers, thus creating the concept of learning-based model predictive control. Among the different learning models that can be used as approximators, deep neural networks have achieved a large success thanks to their representational capabilities, giving birth to the so-called deep MPC (Chen et al., 2018).

For the sake of clarity, it is important to underline how the strategy of using a machine learning model as a model for the prediction in a standard MPC scheme differs from the concept of learning-based MPC. The former strategy models the plant via a machine learning model (*i.e.* performing a "black-box" identification phase) and proceeds to solve a constrained optimization problem in real-time to obtain the control action; in the case in which deep neural networks are used, this can actually be impracticable, due to the strong nonlinearities of the considered model. The latter, instead, relies on a learning model for mapping the states into each optimal action, after it has been trained beforehand on a dataset generated by applying the predictive controller that this mechanism tries to approximate. The learning-based MPC, as it stands, only needs to carry out in real-time a prediction step of the machine learning model, therefore keeping the computational cost at a significantly more manageable level.

Related works in deep MPC include Hertneck et al. (2018), where the authors describe a model that is robust to input errors, and Karg and Lucia (2020), where a deep predictive controller is shown to be able to exactly represent, with a sufficiently high number of neurons and layers, an explicit MPC control law. For controlling batteries, the work described in Tian et al. (2019) has used an explicit MPC based on a piece-wise function for computing the optimal action, so as to guarantee an optimal performance of charging in real-time. Other works (Park et al., 2020, 2022; Attia et al., 2020) have discussed the exploitation of charging techniques relying either on deep reinforcement learning or on a Bayesian optimization algorithm.

In this paper, an extension is proposed of the works presented by the same authors in Pozzi et al. (2022b,a). In particular, a learning-based predictive controller is here developed to optimally charge a battery described by electrochemical and thermal equations, providing a realistic description of several phenomena that occur inside a Lithium-Ion cell. Firstly, a comparison of the performance of several machine learning algorithms in approximating the predictive control law is conducted, demonstrating that the use of deep neural networks allows these models to achieve the lowest mean squared error in validation.

The proposed approach then undergoes a comparison with a standard predictive control strategy, in terms of their respective computational complexity required "online". This comparison, which is conducted under the assumption of full state measurability, shows that the proposed approach manages to display a significantly lower computational time, by keeping the computational cost basically constant with the increase of the prediction horizon, while at the same time providing a charging performance similar to the standard MPC's.

Finally, a more realistic scenario is considered in which only current, voltage, and temperature are assumed to be measurable. Due to the fact that both the standard MPC and the learning-based algorithm are based on the concept of state feedback, suitable modifications need to be adopted to reproduce the experiment when the assumption of the state's availability is dropped. In particular, the coupling of the standard MPC with a state observer able to reconstruct the battery's internal states from the noisy measurements is required at each time step. Within this context, to provide the readers an insight into the possibility of using a well-known observer that achieves reasonable performance, an extended Kalman filter is included in the standard predictive control scheme. Although the same observer could be similarly used for the learning-based algorithm, the implementation of an output-based deep MPC is here considered as a low-effort alternative that does not require the time-consuming design of a suitable observer. The two adapted algorithms are then compared with an ideal MPC that has access to the battery's internal states and it is here considered as a benchmark. The results show that the output-based deep MPC can accurately reproduce the benchmark performance, while the observer-based MPC exhibits oscillations in the applied current, which may lead to sub-optimal thermal management and violations of the voltage constraint.

In summarizing, in this paper, a learning-based predictive control is presented to reduce the online computational burden of fast-charging algorithms. In addition, for the first time to the knowledge of the authors, a computationally-efficient neural network-based predictive controller is successfully employed for the optimal charging of a lithium-ion battery in the presence of non-measurable states. Finally, a minor contribution is also provided to the field of battery modeling, since the heat generation term is here properly approximated to make the model equations twice continuously differentiable and hence suitable to be employed in an optimization framework. In this way, it is possible to guarantee a fair comparison in terms of real-time computational cost between the standard predictive controller (which relies on the online solution of an optimal control problem) and the one based on deep learning (for which the optimizations are solved offline).

This work is structured as follows. Section 2 details how the battery is modeled. Section 3 presents the formulation of the learning-based predictive controller. Section 4 shows the training approach. Section 5 shows the experimental results. Finally, in Section 6 conclusions are drawn.

2. Model

Among the different models available to illustrate the behavior of lithium-ion cells, the one based on simplified electrochemical dynamics has gained a high level of relevance, since it represents an optimal compromise between both accuracy and computational burden. In particular, the Single Particle Model (SPM) (Santhanagopalan et al., 2006) and its variants, which have originated from the Doyle–Fuller-Newman model by simplifying the two electrodes as spherical particles, have proven promising results when employed within a control framework (see *e.g.* Perez et al., 2016 and Pozzi and Toti, 2022). Moreover, they have also led to interesting conclusions within the context of the identifiability of parameters (Pozzi et al., 2020) and the observability of the states (Moura et al., 2017).

In this paper, the SPM variant considered by the authors is the one proposed in Pozzi et al. (2018a), in which, in order to further diminish the computational burden, the Fick's law is reduced into an ordinary differential equation according to the polynomial approximation proposed in Subramanian et al. (2005). With regards to the thermal dynamics, the equations presented by Perez et al. (2017) are adopted, in which the SPM dynamics are integrated with a two-state temperature model.

2.1. Equations of the adopted model

The battery state of charge is represented by the variable $\mathrm{soc}(t) \in [0, 1]$ whose dynamics is described as

$$\frac{d\,\mathrm{soc}(t)}{d\,t} = \frac{I(t)}{3600C} \tag{1}$$

where $I(t)$ is the applied current with the convention that a positive current charges the cell and C is the cell capacity expressed in [Ah]. The condition in which the battery is fully charged occurs when $\mathrm{soc}(t) = 1$, while $\mathrm{soc}(t) = 0$ indicates a complete discharged status. The average stoichiometry levels in the electrodes can be computed from the state of charge as follows:

$$\theta_n(t) = \Delta\theta_n \mathrm{soc}(t) + \theta_n^0 \tag{2a}$$

$$\theta_p(t) = \frac{\theta_n(t) - \theta_n^0}{\Delta\theta_n}\Delta\theta_p + \theta_p^0 \tag{2b}$$

where for $i \in \{n, p\}$ $\Delta\theta_i = \theta_i^1 - \theta_i^0$ is defined, with θ_i^1 and θ_i^0 the stoichiometry values when the cell is completely charged or discharged, respectively. The surface stoichiometry can be obtained thanks to the approximation in Subramanian et al. (2005) as

$$\theta_n^*(t) = \theta_n(t) + \frac{8R_{p,n}}{35c_{s,n}^{\max}}q_n(t) + \frac{R_{p,n}}{35a_n D_{s,n}(T_{\mathrm{avg}}(t))AL_n c_{s,n}^{\max}}I(t) \tag{3a}$$

$$\theta_p^*(t) = \theta_p(t) + \frac{8R_{p,p}}{35c_{s,p}^{\max}}q_p(t) - \frac{R_{p,p}}{35a_p D_{s,p}(T_{\mathrm{avg}}(t))AL_p c_{s,p}^{\max}}I(t) \tag{3b}$$

where $T_{\mathrm{avg}}(t) = \frac{T_c(t)+T_s(t)}{2}$ is the average cell temperature and, for $i \in \{n, p\}$, $q_i(t)$ is defined as the average concentration flux, $R_{p,i}$ as the particle radius, $c_{s,i}^{\max}$ as the maximum ions concentration in the electrodes, a_i as the specific active surface area, A as the current collector area, L_i as the section length, $D_{s,i}(T_{\mathrm{avg}}(t))$ as the solid phase diffusion coefficient which depends on the average cell temperature according to the Arrhenius law. The specific active surface area for $i \in \{n, p\}$ is computed as $a_i = \frac{3\epsilon_i}{R_{p,i}}$ where ϵ_i is the active material volume fraction. The dynamics of the average concentration flux is given by

$$\frac{d\,q_n(t)}{d\,t} = -\frac{30D_{s,n}(T_{\mathrm{avg}}(t))}{R_{p,n}^2}q_n(t) + \frac{45}{2a_n R_{p,n}^2 FAL_n}I(t) \tag{4a}$$

$$\frac{d\,q_p(t)}{d\,t} = -\frac{30D_{s,p}(T_{\mathrm{avg}}(t))}{R_{p,p}^2}q_p(t) - \frac{45}{2a_p R_{p,p}^2 FAL_p}I(t) \tag{4b}$$

where F is the Faraday constant. The terminal voltage of the cell is computed as

$$V(t) = U_p(t) - U_n(t) + \eta_p(t) - \eta_n(t) \tag{5}$$

where, for $i \in \{n,p\}$, $U_i(t)$ and $\eta_i(t)$ are defined as the open circuit potential and overpotential, respectively. The open circuit potentials are generally represented as nonlinear functions of the surface stoichiometry which depends on the considered cell (see 2.2). The overpotentials are computed as follows

$$\eta_n(t) = \frac{2RT_{\mathrm{avg}}(t)}{F}\sinh^{-1}\left(\frac{I(t)}{2a_n AL_n i_{0,n}(t)}\right) \tag{6a}$$

$$\eta_p(t) = \frac{2RT_{\mathrm{avg}}(t)}{F}\sinh^{-1}\left(-\frac{I(t)}{2a_n AL_n i_{0,n}(t)}\right) \tag{6b}$$

where $i_{0,i}(t)$ is the exchange current density and for $i \in \{n, p\}$ is given by

$$i_{0,i}(t) = Fk_i(T_{\mathrm{avg}}(t))c_{s,i}^{\max}\sqrt{c_e\theta_i(t)(1-\theta_i(t))} \tag{7}$$

where c_e is the average electrolyte concentration, here assumed to be constant in the spatial and temporal domain since the electrolyte dynamics is neglected, and $k_i(T_{\mathrm{avg}}(t))$ is the kinetic rate reaction which is dependent on the cell average temperature according to the Arrhenius law.

Finally, a two states model is adopted for the thermal dynamics, in which the core and the surface temperatures are represented by $T_c(t)$ and $T_s(t)$, respectively. In particular, it holds that

$$C_c\frac{d\,T_c(t)}{d\,t} = Q(t) - \frac{T_c(t) - T_s(t)}{R_{c,s}} \tag{8a}$$

$$C_s\frac{d\,T_s(t)}{d\,t} = \frac{T_c(t) - T_s(t)}{R_{c,s}} - \frac{T_s(t) - T_{env}}{R_{s,e}} \tag{8b}$$

where $Q(t)$ is the generated heat, while C_c and C_s are, respectively, the heat capacity of the core and the surface of the cell, $R_{c,s}$ is the thermal resistances between the core and the surface, and $R_{s,e}$ is the thermal resistance between the surface and the external environment. The heat generation is usually modeled in the literature as follows

$$Q(t) = |I(t)(V(t) - U_p(t) + U_n(t))| \tag{9}$$

in order to describe the Joule effect as well as the heat dissipated by the electrodes' overpotential. However, due to its discontinuous derivative, (9) is not suitable to be employed in an optimal control framework that relies on the implementation of the interior point method provided by IPOPT (Wächter and Biegler, 2006), such as the one considered as a benchmark in this paper. In particular, the presence of a discontinuous derivative may lead to both numerical issues and convergence to a point of local infeasibility, thus making not fair the comparison of computational costs conducted in the present work. Therefore, the absolute value is approximated in (9) with a twice continuously differentiable function as follows

$$Q(t) \simeq \sqrt{(I(t)(V(t) - U_p(t) + U_n(t)))^2 + \epsilon} \tag{10}$$

with chosen as small as $\epsilon = 10^{-10}$. It is interesting to notice that also Eqs. (6) and (7) are discontinuous, however, these points of discontinuity are outside the admissible region for the state of charge, and therefore no approximation is required.

2.2. Model parameters

The electrochemical parameters are taken from Ecker et al. (2015b,a), where a commercial cell (in particular, the Kokam SLPB 75106100) is experimentally characterized. The nonlinear functions for the open circuit potentials are chosen as follows

$$U_p(t) = 18.45\theta_p(t)^6 - 40.7\theta_p(t)^5 + 20.94\theta_p(t)^4 + 8.07\theta_p(t)^3 - 7.84\theta_p(t)^2 + 0.024\theta_p(t) + 4.57 \tag{11a}$$

$$U_n(t) = \frac{0.1261\theta_n(t) + 0.00694}{\theta_n(t)^2 + 0.6995\theta_n(t) + 0.00405} \tag{11b}$$

where both the structure and the parameters are selected in order to maximize the fitting with the data collected in Ecker et al. (2015b).

The thermal parameters here considered are chosen as the ones used by the authors in Perez et al. (2017), except for the thermal resistance between the surface and external environment, which is here set as $R_{s,e} = 10$ K W^{-1} to represent a cell in the center of a battery pack (*i.e.* in less favorable conditions in terms of the heat dissipation).

3. Methodology

In the following section, the methodology on which the learning-based predictive controller relies is described as listed afterwards. Firstly, in 3.1, the equations of a general nonlinear model are introduced. Then, in Section 3.2, the main features of the standard predictive control that needs to be approximated are recalled. Finally, in 3.3, the main characteristics of the learning-based MPC are illustrated.

3.1. General nonlinear model

In the present paper, a set of continuous-time nonlinear equations is used to model the real plant dynamics as follows:

$$\frac{d\,\mathbf{x}(t)}{d\,t} = f(\mathbf{x}(t),\,\mathbf{u}(t)) \tag{12a}$$

$$\mathbf{y}(t) = g(\mathbf{x}(t),\,\mathbf{u}(t)) \tag{12b}$$

where the time is described by the variable $t \in \mathbb{R}_+$, while $\mathbf{x}(t) \in \mathbb{R}^{n_x}$, $\mathbf{u}(t) \in \mathbb{R}^{n_u}$ and $\mathbf{y}(t) \in \mathbb{R}^{n_y}$ are the states, inputs and outputs vectors, respectively. In addition, the state and input pairs are mapped into the state derivatives and outputs through the functions $f : \mathbb{R}^{n_x}\times\mathbb{R}^{n_u} \to \mathbb{R}^{n_x}$ and $g : \mathbb{R}^{n_x}\times\mathbb{R}^{n_u} \to \mathbb{R}^{n_y}$, respectively.

Since in the following discussion it is assumed that a digital controller is adopted, which applies a piecewise constant input at the discrete times t_k, $k \in \mathbb{N}$ with sample time t_s, the nonlinear system in (12) is discretized accordingly:

$$\mathbf{x}(t_{k+1}) = f_d(\mathbf{x}(t_k),\,\mathbf{u}(t_k)) \tag{13a}$$

$$\mathbf{y}(t_{k+1}) = g(\mathbf{x}(t_{k+1}),\,\mathbf{u}(t_k)) \tag{13b}$$

where $f_d : \mathbb{R}^{n_x}\times\mathbb{R}^{n_u} \to \mathbb{R}^{n_x}$ maps the input and state pairs into the state at the next time step. Note that the outputs vector $y(t)$ may be discontinuous in $t = t_k$, $k \in \mathbb{N}$ due to the piece-wise nature of the applied inputs vector. Therefore, the value $y(t_k)$ is not uniquely defined, since it exhibits different left and right limits for $t \to t_k$. As clearly highlighted in (13), in this paper the convention for which $y(t_k)$ coincides with the left limit of $y(t)$ $t \to t_k$ is adopted, *i.e.*

$$y(t_k) = \lim_{t\to t_k^-} y(t) \tag{14}$$

As a consequence, in the rest of the paper, the generic input sequence applied in the time interval $[t_k,\,t_{k+H}]$, with $H \in \mathbb{N}$, is represented as

$$\mathbf{u}_{[t_k,t_{k+H}]} = \left[\mathbf{u}(t_k)^\top\,\mathbf{u}(t_{k+1})^\top\,\ldots\,\mathbf{u}(t_{k+H-1})^\top\right]^\top. \tag{15}$$

3.2. Model predictive control

Among the different control methodologies, the ones relying on a predictive model and on a receding horizon framework (Bemporad and Morari, 1999) have been proven particularly successful in dealing with nonlinear processes subject to states and inputs constraints (Camacho and Alba, 2013). Within this context, the model predictive control scheme computes at each time step t_k the optimal control sequence $\mathbf{u}^\star_{[t_k,t_{k+H}]}$ over the prediction horizon H, solving a constrained optimization problem with the cost function depending on the predictions made by a mathematical model of the plant. Then, according to the receding horizon paradigm only the first element $\mathbf{u}^\star(t_k)$ of the resulting optimal input sequence is applied, while the remaining future optimal moves are discarded.

Specifically, the following optimization problem is solved at each time step in order to compute the control action:

$$\mathbf{u}^\star_{[t_k,t_{k+H}]} = \underset{\mathbf{u}_{[t_k,t_{k+H}]}}{\operatorname{argmin}}\, J(\mathbf{x}(t_k)) \tag{16}$$

subject to

system dynamic in (12) (17a)

$$\mathbf{u}^{lb} \le \mathbf{u}(t_i) \le \mathbf{u}^{ub}, \quad i = k,\,k+1,\,\ldots,\,k+H-1 \tag{17b}$$

$$\mathbf{x}^{lb} \le \mathbf{x}(t_i) \le \mathbf{x}^{ub}, \quad i = k+1,\,k+1,\,\ldots,\,k+H \tag{17c}$$

$$\mathbf{y}^{lb} \le \mathbf{y}(t_i) \le \mathbf{y}^{ub}, \quad i = k+1,\,k+1,\,\ldots,\,k+H \tag{17d}$$

with $\mathbf{u}^{lb}, \mathbf{u}^{ub} \in \mathbb{R}^{n_u}$ being the limits for the input vector, $\mathbf{x}^{lb}, \mathbf{x}^{ub} \in \mathbb{R}^{n_x}$ the ones for the state vector and $\mathbf{y}^{lb}, \mathbf{y}^{ub} \in \mathbb{R}^{n_x}$ the ones for the output vector. The cost function $J(\mathbf{x}(t_k))$ to be minimized is formulated as follows

$$J(\mathbf{x}(t_k)) = \|\mathbf{x}(t_{k+H}) - \mathbf{x}_{\text{ref}}\|^2_{Q_H} + \sum_{i=k+1}^{k+H} \times \left[\|\mathbf{x}(t_i) - \mathbf{x}_{\text{ref}}\|^2_{Q_x} + \|\mathbf{y}(t_i) - \mathbf{y}_{\text{ref}}\|^2_{Q_y}\right] + \sum_{i=k}^{k+H-1} \|\mathbf{u}(t_i) - \mathbf{u}_{\text{ref}}\|^2_R \tag{18}$$

where the vectors $\mathbf{x}_{\text{ref}} \in \mathbb{R}^{n_x}$, $\mathbf{y}_{\text{ref}} \in \mathbb{R}^{n_y}$ and $\mathbf{u}_{\text{ref}} \in \mathbb{R}^{n_u}$ correspond to the reference point that the MPC aims to track and the matrices $Q_x \in \mathbb{R}^{n_x\times n_x}$, $Q_y \in \mathbb{R}^{n_y\times n_y}$,$Q_H \in \mathbb{R}^{n_x\times n_x}$ and $R \in \mathbb{R}^{n_u\times n_u}$ are design parameters, with $Q_x, Q_y, Q_H \ge 0$ and $R > 0$. Note that the term $\|\mathbf{x}(t_{k+H}) - \mathbf{x}_{\text{ref}}\|^2_{Q_H}$ represents a suitably tuned terminal penalty used to improve the controller stability.

The MPC control law is defined as follows as:

$$\mathbf{u}_{\text{mpc}}(\mathbf{x}(t_k)) = \mathbf{u}(t_k)^\star \tag{19}$$

which is computed by solving the problem in (16) for the states vector $\mathbf{x}(t_k)$.

3.3. Learning-based model predictive control

Learning-based predictive control relies on the representation capabilities of machine learning models to obtain an accurate approximation of the feedback law of a standard predictive controller. The methodology taken under consideration appears as an alternative to the concept of "explicit" MPC, with the aim of further reducing the online computational burden of the control scheme. In fact, the "explicit" MPC, which is based on a piece-wise function to represent the control law, requires finding the polytopic region in which the states are located online, which may be computationally expensive if the number of regions that define the corresponding piece-wise function is high. The learning-based methodologies solve this issue by directly mapping the current states to the optimal action through a machine learning model.

The concept of learning-based MPC was proposed for the first time by Åkesson and Toivonen (2006), but its application in the design of fast MPC techniques has become reliable only in recent times, thanks mainly to new progress in the theoretical description of neural networks. In particular, robustness guarantees against imprecise inputs have been proposed by Hertneck et al. (2018), while the width and depth requisites that a deep learning model should have to accurately represent the feedback law of a given linear MPC have been identified by the authors in Karg and Lucia (2020). In the remaining part of this section, the deep MPC formulation employed in this work is presented.

Let consider in the following a machine learning model $\mathcal{N} = \mathcal{N}(\mathbf{x},\,\mathbf{r},\,\theta)$ with n_x inputs and n_u outputs, where θ represents the parameters of the data-driven model, while $\mathbf{r}$ describes the vector of the MPC references, *i.e.* $\mathbf{r} = [\mathbf{x}^\top_{\text{ref}}\,\mathbf{y}^\top_{\text{ref}}\,\mathbf{u}^\top_{\text{ref}}]^\top$. The training data are obtained by solving (16) for n_{tr} different states and references samples, *i.e.* $\mathbf{x}_{\text{tr},i}$ and $\mathbf{r}_i$ respectively for $i = 1, 2, \ldots, n_{\text{tr}}$, and storing the tuples $(\mathbf{x}_{\text{tr},i},\,\mathbf{r}_i,\,\mathbf{u}_{\text{mpc}}(\mathbf{x}_{\text{tr},i}))$ in the dataset $\mathcal{B}_{\text{tr}}$, where $\mathbf{u}_{\text{mpc}}(\cdot)$ is the MPC control action.

The model $\mathcal{N}$ is then trained off-line on the dataset $\mathcal{B}_{\text{tr}}$ by solving the following optimization through the back-propagation method:

$$\theta^\star = \underset{\theta}{\operatorname{argmin}}\,\frac{1}{n_{\text{tr}}}\sum_{i=0}^{n_{\text{tr}}} \|\mathbf{u}_{\text{mpc}}(\mathbf{x}_{\text{tr},i}) - \mathcal{N}(\mathbf{x}_{\text{tr},i},\,\mathbf{r}_i,\,\theta)\|^2_2 \tag{20}$$

where the loss function is the mean squared error between the model prediction and the target. Finally, the actual control action is obtained as follows:

$$\mathbf{u}_{\text{l-mpc}}(\mathbf{x}(t_k)) = \mathcal{N}(\mathbf{x}(t_k),\,\mathbf{r},\,\theta^\star) \tag{21}$$

where the subscript "l-mpc" indicates the learning-based MPC algorithm.

4. Training of the controller

This section is devoted to the definition of the optimal charging problem for a lithium-ion cell in 4.1, as well as to the phases of dataset generation and model training in 4.2 and 4.3, respectively.

4.1. Optimal battery charging

The rapid charging of a battery is articulated in this section as an optimal control problem with state-of-charge tracking and current minimization as the target function, while setting safety limits on voltage and temperature. In first place, the battery dynamics (see Eqs. (1)–(10)) is expressed using the model in (12) and then discretized as the model in (13), with input chosen as the applied current. Then, the optimal control problem in (16) is formulated as follows:

$$\min_{I_{[t_k,t_{k+H}]}} q_{\text{soc}} \sum_{i=k+1}^{k+H} (\text{soc}(t_k) - \text{soc}_{\text{ref}})^2 + r \sum_{i=k}^{k+H-1} I(t_k)^2 + q_H (\text{soc}(t_{k+H}) - \text{soc}_{\text{ref}})^2 \tag{22}$$

that for $i = k, k+1, \dots, k+H-1$ are subject to

$$\text{battery dynamics in (1)–(9)} \tag{23a}$$

$$0 \le I(t_i) \le 10\,\text{A} \tag{23b}$$

$$0 \le \text{soc}(t_i) \le 1 \tag{23c}$$

$$T_c(t_i) \le 313.15\,\text{K} \tag{23d}$$

$$T_s(t_i) \le 313.15\,\text{K} \tag{23e}$$

$$V(t_i) \le 4.2\,\text{V} \tag{23f}$$

with $q_{\text{soc}} = 1$, $r = 10^{-6}$ and $q_H = 1$. Moreover, the sample time is taken as $t_s = 10$ s. Important to note is that the value of the state of charge reference (soc_{ref}) is not specified here as it varies depending on specific charging preferences, contrary to the current, temperature and voltage limits which are determined by battery chemistry. Furthermore, during the generation phase of the dataset used for the learning-based algorithm, the reference state of charge is considered as a random variable with uniform probability within a specific range. This enables the learning-based MPC to adapt the charging profile to the various potential values of the reference. Finally, the current reference, *i.e.* the input reference, is always taken as $I_{\text{ref}} = 0$ since the system is marginally stable.

4.2. Dataset generation

The training phase of the learning-based algorithm has to be conducted on a sufficiently wide dataset, in which each sample needs to include information on both the state of the system and the optimal action applied by the predictive controller to be approximated. It is fundamental to underline that poor data quality can negatively impact the performance of a data-driven algorithm, as the one proposed in the manuscript. If the data is inaccurate, inconsistent, or otherwise flawed, the algorithm may produce incorrect or biased results. This can lead to incorrect conclusions, poor performance, and a lack of trust in the algorithm and the results it produces. In addition to data quality, the quantity of data is also an important factor in avoiding overfitting in a machine learning model. Overfitting occurs when a model is too complex and has too many parameters relative to the amount of training data, causing it to memorize the training data and perform poorly on new, unseen data. Increasing the quantity of data helps to alleviate overfitting by providing the model with more examples to learn from and generalize to new data. However, simply increasing the quantity of data is not sufficient to prevent overfitting. The data must also be diverse and representative of the real-world data distribution to prevent the model from memorizing specific features or patterns in the data. Therefore, a combination of both high-quality and large-quantity data is needed to train robust machine learning models that generalize well to new data.

Table 1
Mean Square Error (MSE) over the validation set of the machine learning models considered as possible candidates for the MPC law approximation under the assumption of measurable states.

Model	Degree of polynomials	MSE
Linear regression	1	2.37
Linear regression	5	0.37
Support vector machine (Linear Kernel)	5	0.51
Random Forest (*max_depth* = 7)	4	0.31
DNN (1×10)	–	0.8
DNN $(1 \times 50, 1 \times 10)$	–	0.10
DNN (3×10)	–	0.15
DNN $(1 \times 100, 1 \times 50, 1 \times 10)$	–	0.09
DNN $(1 \times 100, 1 \times 50, 1 \times 10,$ *tanh*$)$	–	0.23
RNN $(LSTM\ 16, 1 \times 100, 1 \times 50, 1 \times 10)$	–	0.12
RNN $(LSTM\ 16, 1 \times 100, 1 \times 50, 1 \times 10,$ *tanh*$)$	–	0.14
DNN $(3 \times 50, 3 \times 10)$	–	0.11
DNN $(2 \times 100, 2 \times 50, 2 \times 10)$	–	0.15
DNN $(3 \times 100, 3 \times 50, 3 \times 10)$	–	0.08
DNN $(3 \times 100, 3 \times 50, 3 \times 10,$ *tanh*$)$	–	**0.05**
RNN $(LSTM\ 16, 3 \times 100, 3 \times 50, 3 \times 10)$	–	0.16
RNN $(LSTM\ 16, 3 \times 100, 3 \times 50, 3 \times 10,$ *tanh*$)$	–	0.15

As a result, firstly, an MPC controller which executes the control task specified in 4.1 is implemented. The dataset is synthetically generated by performing 500 simulations of 200 time-steps each, and storing, for each sample time, the tuple $([\text{soc}_i\ q_p\ q_n\ T_{c,i}\ T_{s,i}]^\top, \text{soc}_{\text{ref},j}, I_i^\star)$, which fully describes the ith time step of the jth simulation. For each simulation j, an initial condition is randomly extracted, and then for each step i the optimal current $I_i^\star$ is computed by solving the optimization problem in (22) for $\text{soc}_{\text{ref},j}$, which is also randomly extracted. It is important to notice that, in every simulation, the evolution of the battery dynamics is computed by adding to the MPC control action a Gaussian noise, with a standard deviation of 2 A, to increase exploration. Finally, the dataset is split into three sets according to training, validation and testing, as usual in machine learning.

4.3. Training phase and model selection

In order to justify the use of a deep learning model for the approximation of the predictive control law, the performance of the following models has been compared against the dataset defined in 4.2:

- a linear regression with and without polynomial features;
- a support vector machine with polynomial features;
- a random forest with polynomial features;
- a Deep feed-forward Neural Network (DNN) with several layers configurations;
- a deep Recurrent Neural Network (RNN) with several layers configurations.

To enhance the learning capabilities of all the considered models, a preprocessing pipeline is considered which involves both the scaling and the standardization of the dataset's features. The results achieved by the different models are shown in Table 1, where the superiority of a deep neural network with 9 fully-connected hidden layers (3 layers of 100 neurons each, 3 layers of 50 neurons each, 3 layers of 10 neurons each), *ReLu* with activation function in the hidden layers and *tanh* activation function in the output layer, is highlighted. Note that the *tanh* activation function on the last layer is used to constrain the output of the network within a specific range, which means, in the context of battery charging, constraining the optimal applied current within its range of operation. While it is true that the *tanh* function may not always improve the performance of the model (or may even worsen it due to some non-trivial interactions between the model nonlinearities, as for the case of the DNN with 1×100, 1×50, 1×10 layers and

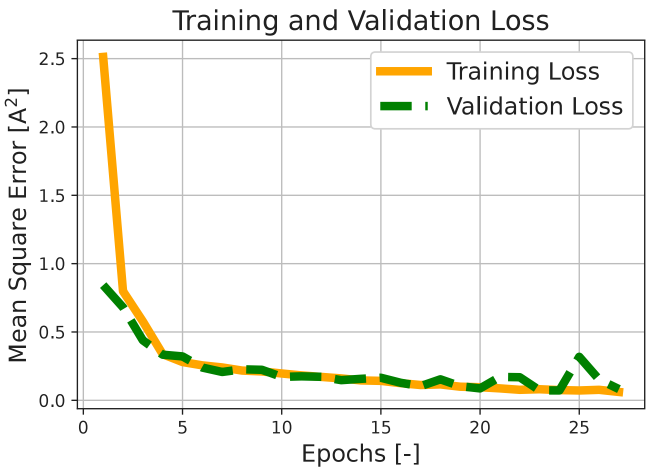


Fig. 1. Training and validation loss profiles against epochs.

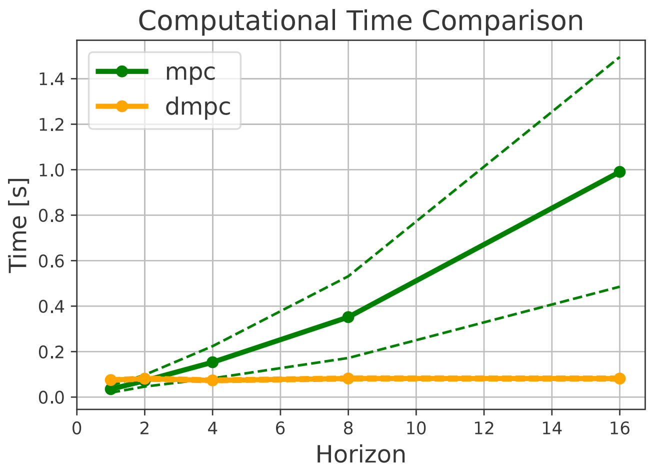


Fig. 2. Comparison of the online computational time for MPC (green) and deep MPC (orange) when the prediction horizon is increased. The results are averaged over 50 simulations with different initial conditions and SOC reference, consisting of 400 samples each. The dashed lines represent the standard deviation ranges for the computational time.

RNN with LSTM 16, 1 × 100, 1 × 50, 1 × 10 layers), it may prevent the model from producing outputs that are unrealistic or invalid for the given task. Note that an alternative to the use of the *tanh* function is to clip the output of the neural network a posteriori. However, this may result in the loss of gradient information during backpropagation, which can lead to a suboptimal performance of the model. In contrast, the *tanh* activation function provides a smooth and continuous way of constraining the output within a specific range, which can help to avoid such problem. Therefore, while there may be alternative approaches to constraining the output, the use of *tanh* activation function can be a reasonable choice depending on the specific problem being solved and the structure of the neural network, thus explaining the reason why in some configurations the use of the *tanh* function is more effective (see DNN with 3 × 100, 3 × 50, 3 × 10 layers) than in others. For all these reasons, such a configuration is selected in this paper to properly approximate the predictive control law, although the solution with 3 hidden layers (100 hidden neurons, 50 hidden neurons, 10 hidden neurons) also provides interesting results and could be considered as a viable option to address computational complexity concerns.

It is noteworthy to mention that Table 1 displays the analysis of the model's performance under the assumption of full state measurability, meaning that the optimal control action is based solely on the current battery state. In this scenario, the neural network aims to learn the relationship between the state and the optimal action by using training samples derived from the solution of a constrained optimization problem. As a result, the input features to the network do not incorporate sequential information. For this reason, the use of recurrent layers does not seem to be particularly useful in this context, as can be seen from Table 1. In fact, no improvement is obtained with respect to the DNN (1 × 100, 1 × 50, 1 × 10) and DNN (3 × 100, 3 × 50, 3 × 10, *tanh*) when adding a Long Short-Term Memory (LSTM) layer with 16 units.

Finally, it is important to emphasize that, concerning the hyperparameter selection, only a subset of the tested configurations has been reported in Table 1, omitting those that have not produced significant variations in the model's performance, with the aim of lightening the discussion and make the paper more readable.

For the training of the deep learning model, the stochastic gradient descent algorithm is employed, and, in particular, the *Adam* optimizer is adopted, with the mean squared error as the loss function and learning rate equal to $5 \cdot 10^{-4}$. It is important to consider that the loss in validation converges after 28 epochs, as depicted in Fig. 1, and that no over-fitting is present, due to the fact that the prediction error during the testing phase ($e_{\text{test}} = 0.053$) is coherent with that achieved on the training set.

5. Results

This section is dedicated to the illustration of the results of the comparison between standard and deep MPC as regards the case of battery charging. More specifically, 5.1 shows a comparison of the computational burden of the two methodologies considered at the variation of the forecasting horizon. Then, in 5.2 the effectiveness of the neural network-based algorithm in tracking the optimal charging profile, under the assumption of full states availability, are highlighted. Such an assumption is removed in 5.3 where only voltage and surface temperature are considered measurable. Within this setting, we propose, for the first time to the knowledge of the authors, the use of an output-based deep model predictive control for the battery fast charging, and we compare its performance with a standard MPC coupled with a properly tuned state observer. Finally, in 5.4 details on the software implementation are provided.

5.1. Computational time comparison

In the following subsection, the online computational performance of the deep learning-based algorithm is compared with that of a standard predictive controller, which is here considered as a benchmark. More particularly, the computational time of the two methodologies is analyzed for different prediction horizons, *i.e.* $H \in \{1, 2, 4, 8, 16\}$ and then 50 simulations are run for each value of the horizon using, firstly, the deep MPC and, secondly, the benchmark controller. Following this procedure, statistical information on the computational burden can be obtained. The results are depicted in Fig. 2, where the solid lines represent the average computational cost (orange for the neural network-based approach, and green for the MPC) and the dashed lines delimit the standard deviation range. The figure illustrates that the computation time of the standard predictive controller presents a superlinear behavior in relation to the prediction horizon, while that of the deep MPC is almost constant (about 80 ms). This is related to the fact that the only online effort demanded by the deep learning-based methodology is the evaluation of the neural network in the measured state, which is independent from the prediction horizon, as the network is identical for every H value. Besides, the standard predictive controller necessitates the solution of an optimization problem whose amount of control variables and constraints increments as the horizon increases.

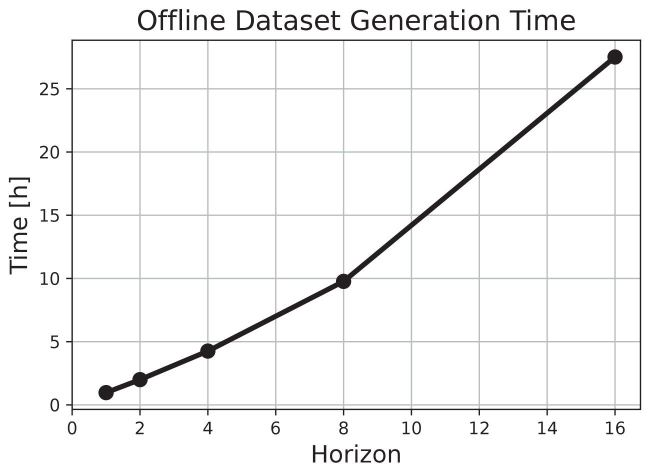


Fig. 3. Analysis of the computational time required to generate the dataset used for the training of the dMPC with respect to the prediction horizon.

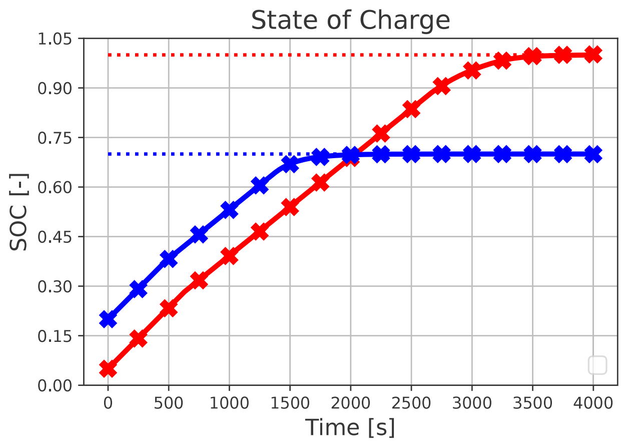


Fig. 4. State of charge profiles of the two considered methodologies (solid line for the standard predictive controller and crossed line for the deep MPC), with *sim 1* in blue and *sim 2* in red. The blue and red dotted lines represent the SOC references for the first and second simulation, respectively.

Interestingly, for a very small horizon, the computational cost of the standard predictive controller may be inferior to that of the proposed approach, since the evaluation of the neural network has a certain predetermined computational cost. However, a short prediction horizon is hardly ever used in a practical scenario, as it can lead to stability problems and breaches of constraints. As a result, the proposed methodology appears to be appropriate for all those situations where a large prediction horizon and a small sampling time are required.

5.1.1. Offline computational burden for the deep MPC

Interestingly, while the use of the neural network-based algorithm presents a lower online computational cost, a great computational effort is required for the offline generation of the dataset, as illustrated in Fig. 3. This is due to the fact that the dataset generation phase requires the execution of the standard MPC repeatedly (in particular 500 simulations each with a length of 200 steps). In fact, each sample of the training dataset is obtained by solving the optimal control problem in Eqs. (22)–(23) of the revised manuscript, therefore creating a larger training set necessarily takes a longer time. Note that since the computational burden of the standard MPC, i.e. the time required to solve Eqs. (22)–(23), depends on the length of the prediction horizon H, the latter affects the duration of the dataset generation procedure as well.

A disadvantage of the proposed methodology is that the generation of the dataset is crucial for the performance of the algorithm, thus making the design and tuning phases more time-consuming. Nevertheless, it is important to underline that, in many real-world applications, the online cost is much more significant than the offline cost because the former must be incurred repeatedly and in real-time, whereas the latter is incurred only once before deployment. This is because the control inputs must be constantly updated as the system evolves over time, and the real-time computation required to apply the updated inputs must be done quickly and accurately to ensure stability and reliability of the system. On the other hand, the offline computational cost is often considered a minor problem because the primary concern is ensuring real-time performance and control stability. Reducing the online computational cost is a key goal, and often the limiting factor in practical implementation. From this perspective, the deep learning-based method is effective as it allows most of the computational cost to be shifted offline, thus enabling real-time applications.

5.2. Approximation of the MPC charging profile with measurable states

In the following subsection it will be evaluated the effectiveness of the neural network-based approach in approximating the charging profile achieved by the standard predictive controller, with particular attention to two different simulation settings. Specifically, the first simulation (*sim 1*) is characterized by $\text{soc}_{\text{ref}} = 0.7$ and initial states given by: $\text{soc}(t_0) = 0.2$, $q_n(t_0) = 0$, $q_p(t_0) = 0$ and $T_s(t_0) = T_c(t_0) = 305.15$ K. The second simulation (*sim 2*) is defined by $\text{soc}_{\text{ref}} = 1$ and initial states given by: $\text{soc}(t_0) = 0.05$, $q_n(t_0) = 0$, $q_p(t_0) = 0$ and $T_s(t_0) = T_c(t_0) = 300.15$ K. For both the simulations $H = 4$ is considered.

The results are illustrated in the Figs. 4–8, with crossed lines representing the deep MPC profiles and solid lines showing the ones of the standard predictive controller. Furthermore, blue lines are used for the profiles which refer to *sim 1* and red lines are used for the ones related to *sim 2*. As can be seen, the trajectories of the system controlled by the two methodologies present very similar characteristics. In particular, it is possible to see from Fig. 4 that both the controllers can track the state of charge references, while Figs. 5–7 show that both the constraints on voltage and temperature are always satisfied. Furthermore, in Fig. 8 the applied current is shown, where the constraints are satisfied by the design for the deep MPC, while in the case of the standard predictive controller they are imposed as bounds on the input. It is interesting to notice that the control strategy obtained with both the controllers for *sim 2* is similar to the so-called constant-current/constant-temperature/constant-voltage protocol which has been demonstrated to be effective in reducing charging time in the presence of current, voltage and temperature constraints (as shown in Patnaik et al. (2018) where multiple PID controllers have been suitably tuned to implement such protocol).

It should be noted that a similar approximation capability for the deep MPC can be achieved for initial conditions and references that falls in the intervals of variation considered during the training phase, which can be made sufficiently wide to account for the entire range of realistic scenarios. In support of this assertion, it is provided in the following a statistical analysis of the difference between standard and deep MPC profiles over 50 simulations, with starting states and references randomly extracted. In particular, it emerges that the approximation error for the applied current is in average -0.60 mA, with a standard deviation of 79.8 mA, as depicted by the histogram in Fig. 9, where the distribution of the samples resembles that of a skew normal one. Moreover, for the state of charge profile the approximation error is in average $-0.109 \cdot 10^{-3}$, with a standard deviation of $0.713 \cdot 10^{-3}$. Finally, the approximation error for the voltage profile is 0.061 mV with a standard deviation of 1.90 mV, while the core temperature exhibits an error of -5.59 mK, with a standard deviation of 50.6 mK. Such results are summarized in Table 2. Of interest is that the learning-based model has more probability of making errors in the approximation of

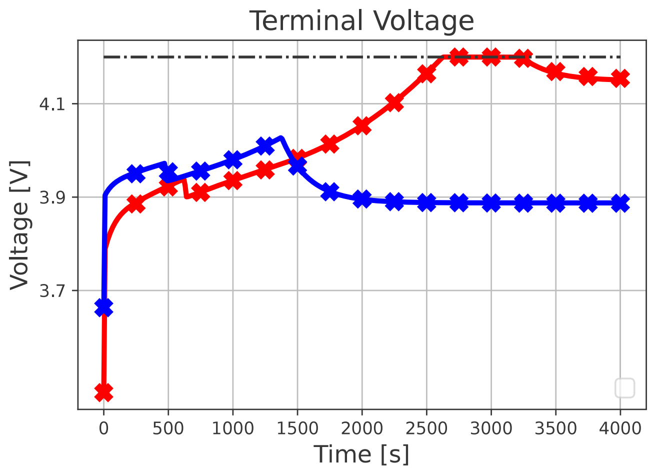


Fig. 5. Voltage profiles of the two considered methodologies. The dash-dotted black line represents the voltage upper bound.

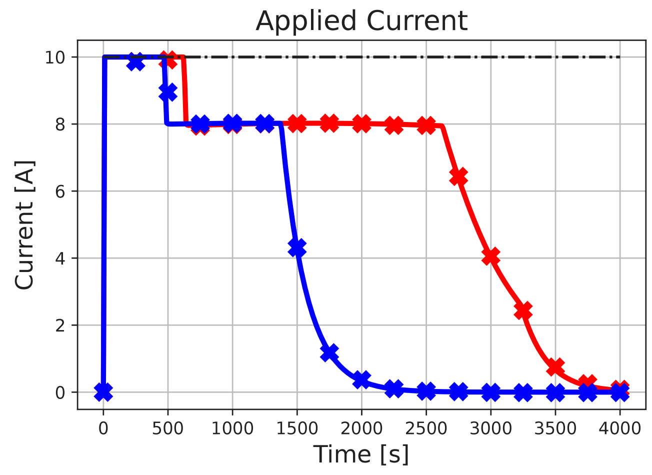


Fig. 8. Applied current profiles of the two considered methodologies.

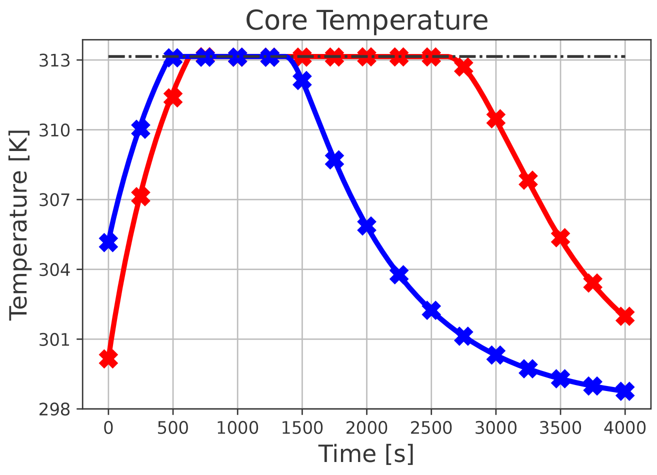


Fig. 6. Core temperature profiles of the two considered methodologies.

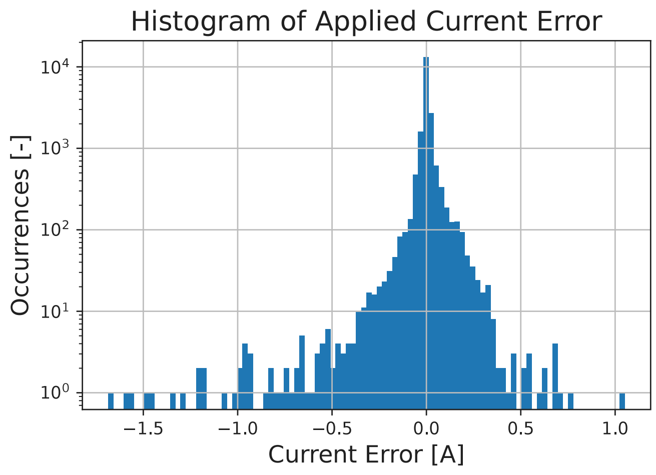


Fig. 9. Histogram of the errors committed over 50 simulations by the dMPC approach in approximating the standard predictive control law.

Table 2
Statistical description over 50 simulations of the difference between the trajectory of the model variables obtained with the two considered control methodologies.

	soc $[10^{-3}]$	V [mV]	T_c [mK]
Mean	−0.109	0.061	−5.59
Std. Dev.	0.713	1.90	50.6

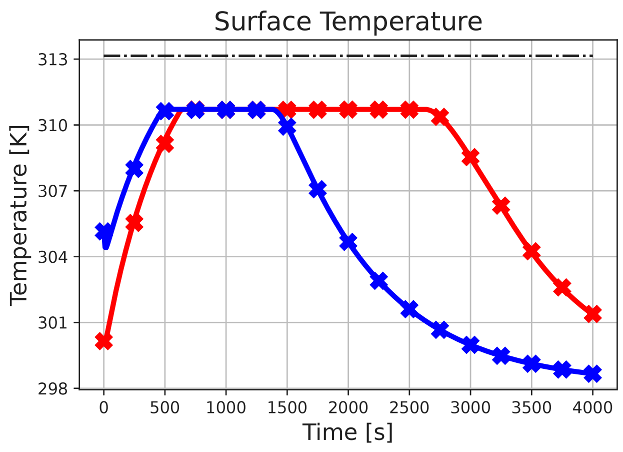


Fig. 7. Surface temperature profiles of the two considered methodologies.

the MPC control law when the system state is located in regions near the reference or constraints. Even though these errors cause the system to make small, damped oscillations around the reference, they do not influence the stability of the control law, at least not from a practical standpoint. However, one possible resolution to this problem could be the inclusion of additional samples corresponding to the most critical points in the dataset.

5.3. Performance evaluation with non-measurable states

In the discussion above the battery internal states have been assumed to be measurable, while in a practical scenario they need to be indirectly reconstructed through the available noisy measurements. The aim of this section is to address the battery charging problem for the case in which the assumption of full states availability is dropped. Specifically, in the following, only the voltage and the surface temperature are made available to the controller, affected by a Gaussian noise with standard deviation of 20 mV for the voltage and 1 K for the temperature. Since both the methodologies discussed in 5.2 rely on the concept of states feedback, the adoption of specific modifications becomes necessary to exploit them in such a realistic situation. The most immediate and natural solution for the standard MPC algorithm is the implementation of an observer to reconstruct

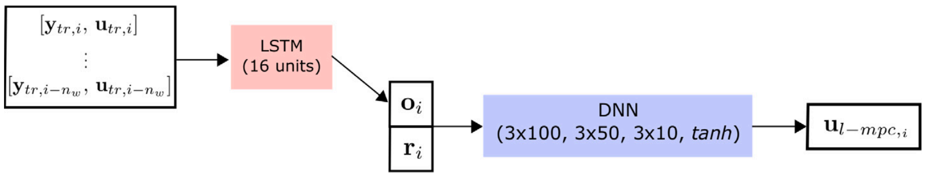

$[\mathbf{y}_{tr,i}, \mathbf{u}_{tr,i}]$
$\vdots$
$[\mathbf{y}_{tr,i-n_w}, \mathbf{u}_{tr,i-n_w}]$
LSTM (16 units)
$\mathbf{o}_i$
$\mathbf{r}_i$
DNN (3x100, 3x50, 3x10, *tanh*)
$\mathbf{u}_{l-mpc,i}$

Fig. 10. Conceptual scheme of the proposed RNN architecture, where $\mathbf{o}_i$ refers to the output vector of the LSTM when the ith sample of the dateset is provided as input of the network.

the state profiles from the available outputs. In particular, among the different state observers previously proposed in literature in the context of lithium-ion batteries (Waag et al., 2014), in this paper, an Extended Kalman Filter (EKF) for continuous-time model with discrete-time measurements (Jazwinski, 2007) is adopted and coupled with the standard predictive controller. In this context, the observer is implemented to evaluate the charge state and core temperature, two essential pieces of information for calculating an appropriate charging protocol, and to filter out noisy surface temperature. The reason for the idea of estimating only a subset of the states is derived from the fact that the average concentration fluxes display asymptotically stable behavior (Moura et al., 2017), whose starting values can be set equal to their equilibrium conditions. The EKF parameters used in the following simulations are described by:

$$P^0_{\mathrm{soc}} = 0.04, \quad P^0_{T_c} = 25, \quad P^0_{T_s} = 25 \tag{24a}$$

$$Q_{\mathrm{soc}} = 0.6 \cdot 10^{-3}, \quad Q_{T_C} = 10^{-3}, \quad Q_{T_s} = 1 \tag{24b}$$

$$R_V = 0.25 \cdot 10^{-4}, \quad R_T = 0.25 \tag{24c}$$

where P^0_{soc}, $P^0_{T_c}$ and $P^0_{T_s}$ are the initial variances of the state of charge, the core temperature and the surface temperature, respectively; Q^0_{soc}, $Q^0_{T_c}$ and $Q^0_{T_s}$ are the variances of the process noise that are assumed to affect the model dynamics; and, finally, R_V and R_{T_s} are the variances of the measurement noise on voltage and surface temperature, respectively.

Although the same observer could be used also for the deep controller, an alternative solution (referred to in the following as output-based dMPC) is here proposed which directly relies on the available measurements. In particular, inspired by the techniques used in the context of the partially observable Markov decision process, a fixed window of historical measurements is used as input for the neural network during the training phase, thus allowing the model to learn a map from the available measurements to the optimal applied current. Therefore, the training phase is reformulated in a way such that each row in the training dataset $\mathcal{B}^{out}_{\mathrm{tr}}$, with $|\mathcal{B}^{out}_{\mathrm{tr}}| = n_{tr} - n_w$, consists of the tuple

$$\mathrm{row}_i = \left(\mathbf{y}_{\mathrm{tr},i-n_w}, \dots, \mathbf{y}_{\mathrm{tr},i}, \mathbf{u}_{\mathrm{tr},i-n_w}, \dots, \mathbf{u}_{\mathrm{tr},i}, \mathbf{r}_i, \mathbf{u}_{\mathrm{mpc}}(\mathbf{x}_{\mathrm{tr},i})\right) \tag{25}$$

where $\mathbf{y}_{\mathrm{tr},i}$ and $\mathbf{u}_{\mathrm{tr},i}$ are the samples of the system's outputs and inputs, respectively, while n_w is the length of the window of the considered historical measurements. The neural network $\mathcal{N}$ is then trained off-line on the dataset $\mathcal{B}^{out}_{\mathrm{tr}}$ by solving the following optimization through the back-propagation method:

$$\theta^\star = \operatorname*{argmin}_{\theta} \frac{1}{n_{\mathrm{tr}} - n_w} \sum_{i=n_w}^{n_{\mathrm{tr}}} \|\mathbf{u}_{\mathrm{mpc}}(\mathbf{x}_{\mathrm{tr},i}) - \mathcal{N}(\mathrm{row}_i, \theta)\|_2^2 \tag{26}$$

where the loss function is the mean squared error between the network prediction and the target. In particular, a window of historical measurements equal to 7 is here adopted (i.e. $n_w = 7$).

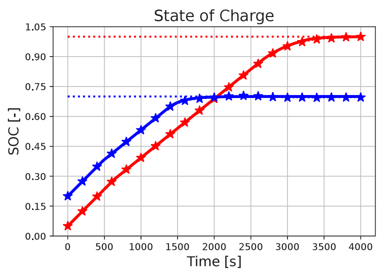
As far as the selection of the machine learning model is concerned, the use of a recurrent neural network, such as the one depicted in Fig. 10, is found to be particularly valuable due to its capability in handling sequential data. This is also demonstrated by the results in Table 3, where the superiority of the RNN-based architecture is highlighted when compared with the different feed-forward solutions, in the case of non-measurable states.

Table 3
Mean Square Error (MSE) over the validation set of the machine learning models considered as possible candidates for the MPC law approximation in the case of non-measurable states.

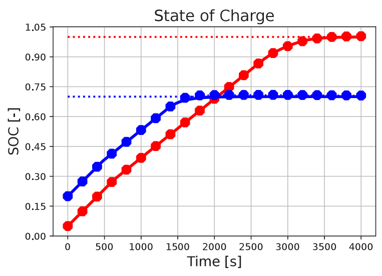
Model	MSE
DNN (1 × 100, 1 × 50, 1 × 10)	0.25
DNN (1 × 100, 1 × 50, 1 × 10, *tanh*)	0.20
DNN (3 × 100, 3 × 50, 3 × 10)	0.22
DNN (3 × 100, 3 × 50, 3 × 10, *tanh*)	0.19
RNN (*LSTM* 16, 3 × 100, 3 × 50, 3 × 10)	0.19
RNN (LSTM 16, 3 × 100, 3 × 50, 3 × 10, *tanh*)	**0.16**

In Figs. 11–15, the standard MPC coupled with the EKF (on the left indicated with "⋆") and the output-based deep MPC (on the right represented with "o") are compared with an ideal MPC with full state measurability (solid line) considered as a benchmark. Since the initial states of the battery are assumed to be unknown, the following values are adopted as a guess for the EKF: $\mathrm{soc}^g(t_0) = 0.3$, $T^g_c(t_0) = 302$ K and $T^g_s(t_0) = 302$ K. Although the output-based dMPC does not imply an initial guess for the battery's states, a guess for the first n_w control action is required. The latter is computed offline as the average applied current at the beginning of the charging phase according to the available data in our training set: $I^g(t_k) = 9.7$ A, with $k = 0, \dots, n_w$. On the one hand, it can be noticed that the profiles of state of charge (Fig. 11) are very similar for all the discussed algorithms. On the other hand, the profiles of terminal voltage obtained by the standard predictive controller coupled with the observer (left part of Fig. 12) exhibit an oscillating behavior around the trajectory achieved by the benchmark. This is mainly because the states estimated by the observer at each time step are affected by a certain estimation error, which consequently causes oscillations in the value of the applied current computed by the optimizer as depicted by the left part of Fig. 15. Such current's oscillations are filtered in the plots related to the state of charge due to its slow dynamics, while they lead to an increase in both the core and surface temperature at the end of the charge (see left part of Figs. 13 and 14). Interestingly, no oscillations are present in the voltage and current profiles obtained by employing the output-based dMPC (right graph in Figs. 15 and 12), which can track the optimal current applied by the benchmark with a much higher accuracy if compared with the observer-based MPC.

With the aim of providing further validation of the performance of the proposed algorithm, a statistical analysis has also been extended to the case of non-measurable states. Specifically, Table 4 compares the tracking performance of the output-based dMPC and observer-based MPC against an ideal MPC with measurable states over 50 simulations. The analysis focuses on the main battery variables, including state of charge, voltage, and core temperature. The results show that, while both the algorithms are able to approximate the optimal policy in expectation, the observer-based one exhibits an oscillating behavior that affects its performance, especially in terms of thermal management. In fact, using the observer-based approach, the core temperature ends up being 580 mK higher in expectation than the one of the benchmark, with a standard deviation of 591 mK, while adopting the output-based algorithm only a 35.6 mK increase of the temperature occurs (with a standard deviation 168 mK). This can be further highlighted by evaluating the errors committed by both the algorithms in the

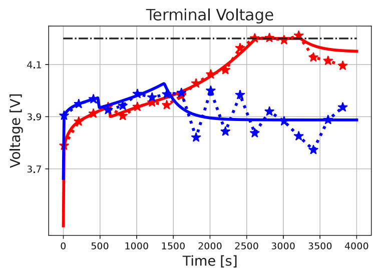


(a) Comparison between the standard MPC + EKF and the benchmark.

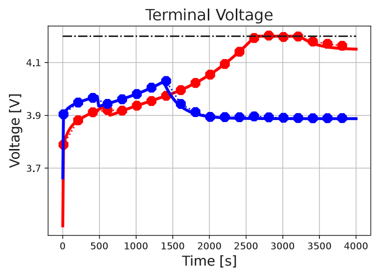


(b) Comparison between the output-based dMPC and the benchmark.

Fig. 11. Comparison between the state of charge profiles obtained by the two discussed methodologies (standard MPC coupled with EKF on the left indicated with "⋆" and output-based dMPC on the right represented with "o") and the benchmark profile achieved by using an ideal MPC with full state measurability. Specifically, the two simulations described in 5.2 are considered: *sim 1* is represented by the blue plot while *sim 2* is depicted in red.

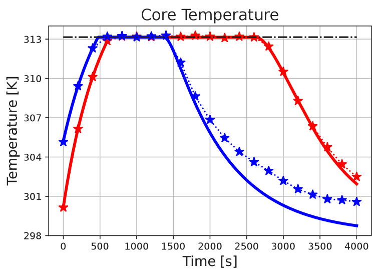


(a) Comparison between the standard MPC + EKF and the benchmark.

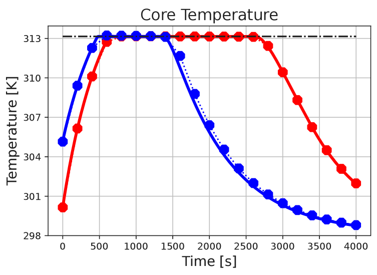


(b) Comparison between the output-based dMPC and the benchmark.

Fig. 12. Comparison between the voltage profiles obtained by the two discussed methodologies and the benchmark profile achieved by using a standard MPC with full state measurability. As it can be noticed, the MPC coupled with the observer (left) leads to voltage oscillations, which are instead mitigated using the output-based dMPC (right).



(a) Comparison between the standard MPC + EKF and the benchmark.



(b) Comparison between the output-based dMPC and the benchmark.

Fig. 13. Comparison between the core temperature profiles obtained by the two discussed methodologies and the benchmark profile achieved by using a standard MPC with full state measurability. The oscillations in the current applied by the observer-based MPC lead to a slight increase of the core temperature at the end of the charge (left), while the output-based dMPC (right) is able to accurately reproduce the temperature profile achieved by the benchmark.

Table 4
Statistical description over 50 simulations of the difference between the trajectory of the model variables obtained with the two considered control methodologies.

	soc $[10^{-3}]$		V [mV]		T_c [mK]	
	dMPC	EKF+MPC	dMPC	EKF+MPC	dMPC	EKF+MPC
Mean	0.116	−0.264	−0.012	−0.413	35.6	580
Std. Dev.	4.25	2.44	7.16	43.4	168	591

approximation of the control action computed by the benchmark, over the 50 simulations, as depicted in Fig. 16, where the superiority of the output-based dMPC is apparent.

It is important to clarify that slight violation of voltage and temperature constraints may occur in a realistic scenario such as the one considered in this section, as a result of the limitations of unknown internal states and the measurement noise affecting the available output. It is acknowledged that in practical scenarios, it is not always possible to guarantee the satisfaction of hard constraints on states and outputs. Instead, these constraints should be treated as soft constraints, meaning they should not be violated if at all possible. To minimize the chance of a constraint violation, appropriate countermeasures can be taken. For example, using a more accurate observer (in the case of observer-based standard MPC) or a more accurate machine learning model (in the case of output-based deep MPC) may help reduce constraint violations. As far as the satisfaction of input constraints is concerned, the inclusion of a *tanh* activation function on the output layer of the neural network model ensures that the applied current does not violate the safety bounds.

The performance of the observer alone is shown in Fig. 17, where the error in estimating the state of charge is depicted for a current input corresponding to the one applied by the observer-based MPC. As it can be noticed, the error becomes small in a few iterations, even if

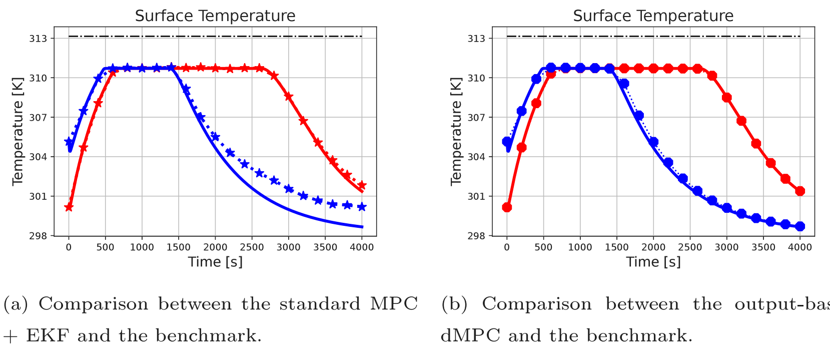


Fig. 14. Comparison between the surface temperature profiles obtained by the two discussed methodologies and the benchmark profile achieved by using a standard MPC with full state measurability. The oscillations in the current applied by the observer-based MPC lead to a slight increase of the surface temperature at the end of the charge (left), while the output-based dMPC (right) is able to accurately reproduce the temperature profile achieved by the benchmark.

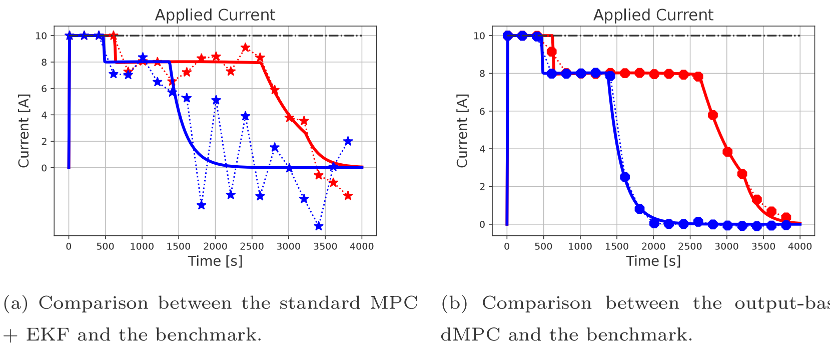


Fig. 15. Comparison between the applied current profiles computed by the two discussed methodologies and the benchmark profile computed by a standard MPC with full state measurability. As it can be noticed, the MPC coupled with the observer (left) leads to current oscillations, which are instead mitigated using the output-based dMPC (right).

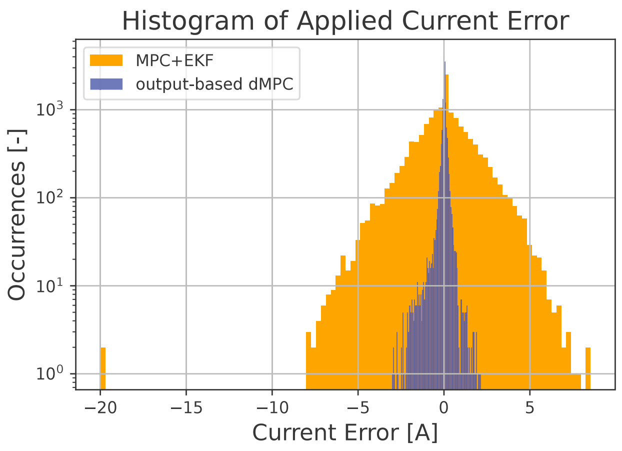


Fig. 16. Statistical analysis of the errors committed by the observer-based MPC and the output-based dMPC in approximating the control law applied by the benchmark algorithm over 50 different simulations. As it can be noticed, the errors of both the algorithms have nearly zero mean, but the standard deviation achieved by the output-based MPC (0.32 A) is much lower than the one obtained by the MPC coupled with the EKF (1.69 A).

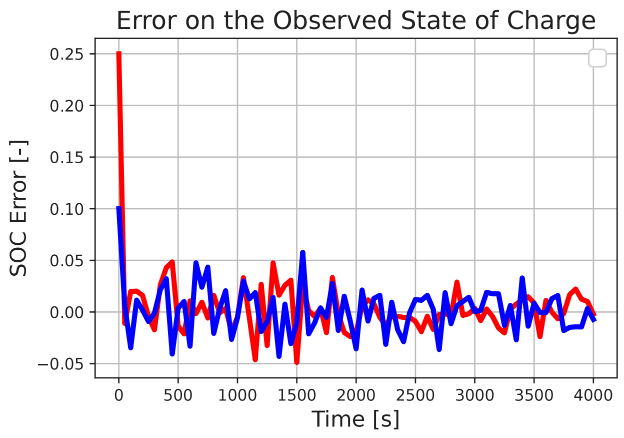


Fig. 17. Evaluation of the observer's performance in estimating the state of charge. Specifically, the figure shows the errors between the actual state values and the observed ones, for an applied current profile equal to the one computed by the observer-based model predictive control.

some slight oscillations remain present, thus confirming the analysis conducted previously.

Finally, it is important to notice that also the coupling of the deep MPC algorithm discussed in 3.3 with the developed state observer is a possible solution. However, it has been noticed in simulation that such a control scheme exhibits the same issues which affect the observer-based MPC (i.e. oscillating variables), which are strictly related to the observer's performance.

5.4. Implementation details

The simulations described in this paper were executed on a Windows 10 personal computer with 16 GB of RAM and a I7-8750H processor and implementation was made in a *Python* environment (version *3.7*). Moreover, the equations of the model are integrated through *CasADi* (Andersson et al., 2019), which consists of a symbolic framework for automatic differentiation. *CasADi* is used also to solve the optimal control problem in (16). Finally, *TensorFlow 2.0* (Abadi et al., 2016) is used to build and train the deep learning model, since it constitutes a well-known framework for large-scale machine learning.

6. Conclusions and future work

In this paper, the optimal charging of a lithium-ion battery modeled with electrochemical and thermal dynamics is addressed by exploiting a control scheme consisting of a machine learning-based approximation of a predictive controller. In particular, a deep neural network is used as an approximator thanks to its high representation capabilities. A first analysis is conducted in order to demonstrate the ability of the proposed methodology to significantly reduce the online computational cost which is known to be a limiting factor for the practical implementation of nonlinear model predictive control algorithms. Subsequently, a more realistic setting is considered in which the assumption of availability of the battery's internal state is dropped. In such a scenario, the learning-based algorithm is adapted to retrieve the optimal charging profile by relying only on current, voltage and temperature measurements, and a comparison with a standard-predictive controller coupled with a suitably tuned state observer is presented. The results show the effectiveness of the learning-based methodology in tracking the reference state of charge and guaranteeing safety while applying a very stable current profile, differently from the observer-based predictive controller that exhibits an oscillating current profile due to the unavoidable inaccuracies in the estimation of the internal battery states which may potentially lead to thermal issues and constraints violations. Finally, it is important to recall that other contributions include the approximation of the equation that represents the generation of heat in the battery to make the model twice differentiable and the generation of a synthetic dataset that was sufficiently informative for the training phase of the deep MPC algorithm.

While the proposed paper and methodology offer a novel contribution, they are not without limitations and issues that must be acknowledged. In particular, the following aspects should be taken into account in future research:

- the discussed analysis relies on the assumption of accurate battery parameter estimation. It is widely recognized in the field that the electrochemical parameters of a lithium-ion cell can vary greatly, even among cells of the same type, and may change as the cell ages. Additionally, determining these parameters often requires time-intensive and intrusive experiments. A possible solution could be the development of an algorithm able to derive the optimal action directly from available measurements, in the case in which not only the battery states but also the model parameters are unknown, under the assumption that only the model structure is known beforehand;
- the output-based dMPC predicts the optimal input for charging a battery based on previous measurements. However, at the start of the charging process, there are only a limited number of previous measurements available, which makes it difficult to use the proposed algorithm effectively. In order to overcome this challenge, a rough guess for the first few control actions is used. The latter is computed offline as the average current applied at the beginning of the charging phase according to the available data in our training set. Unfortunately, the performance of the controller during the first stage of charging will be impacted if the guess input deviates from the optimal one, potentially leading to safety issues. To address this issue, multiple output-based dMPC algorithms may be used during the first stage of charging. Each algorithm would use a different number of previous measurements, starting with an algorithm that uses no previous measurements, then moving on to an algorithm that uses one previous measurement, and so on until a certain amount of measurements are available;
- neural networks are widely criticized for their dependence on large amounts of data for training and their lack of interpretability. Possible solutions include using pre-trained models for similar tasks and relying upon attention and feature activation methods to gain insight into how the model makes its predictions.

It is important to address these limitations in future work to further improve the proposed approach.

CRediT authorship contribution statement

Andrea Pozzi: Conceptualization, Methodology, Software, Investigation, Writing – original draft, Writing – review & editing, Supervision. **Scott Moura:** Writing – review & editing. **Daniele Toti:** Conceptualization, Writing – original draft, Writing – review & editing.

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.

Data availability

Data will be made available on request.

References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al., 2016. Tensorflow: A system for large-scale machine learning. In: 12th $\{USENIX\}$ Symposium on Operating Systems Design and Implementation ($\{OSDI\}$ 16). pp. 265–283.

Åkesson, B.M., Toivonen, H.T., 2006. A neural network model predictive controller. J. Process Control 16 (9), 937–946.

Alessio, A., Bemporad, A., 2009. A survey on explicit model predictive control. In: Nonlinear Model Predictive Control. Springer, pp. 345–369.

Andersson, J.A., Gillis, J., Horn, G., Rawlings, J.B., Diehl, M., 2019. CasADi: a software framework for nonlinear optimization and optimal control. Math. Program. Comput. 11 (1), 1–36.

Attia, P.M., Grover, A., Jin, N., Severson, K.A., Markov, T.M., Liao, Y.-H., Chen, M.H., Cheong, B., Perkins, N., Yang, Z., et al., 2020. Closed-loop optimization of fast-charging protocols for batteries with machine learning. Nature 578 (7795), 397–402.

Bemporad, A., Morari, M., 1999. Robust model predictive control: A survey. In: Robustness in Identification and Control. Springer, pp. 207–226.

Bemporad, A., Oliveri, A., Poggi, T., Storace, M., 2011. Ultra-fast stabilizing model predictive control via canonical piecewise affine approximations. IEEE Trans. Automat. Control 56 (12), 2883–2897.

Camacho, E.F., Alba, C.B., 2013. Model Predictive Control. Springer Science & Business Media.

Chaturvedi, N.A., Klein, R., Christensen, J., Ahmed, J., Kojic, A., 2010. Algorithms for advanced battery-management systems. IEEE Control Syst. 30 (3), 49–68.

Chen, S., Saulnier, K., Atanasov, N., Lee, D.D., Kumar, V., Pappas, G.J., Morari, M., 2018. Approximating explicit model predictive control using constrained neural networks. In: 2018 Annual American Control Conference. ACC, IEEE, pp. 1520–1527.

Csekő, L.H., Kvasnica, M., Lantos, B., 2015. Explicit MPC-based RBF neural network controller design with discrete-time actual Kalman filter for semiactive suspension. IEEE Trans. Control Syst. Technol. 23 (5), 1736–1753.

Ecker, M., Käbitz, S., Laresgoiti, I., Sauer, D.U., 2015a. Parameterization of a physico-chemical model of a lithium-ion battery II. Model validation. J. Electrochem. Soc. 162 (9), A1849–A1857.

Ecker, M., Tran, T.K.D., Dechent, P., Käbitz, S., Warnecke, A., Sauer, D.U., 2015b. Parameterization of a physico-chemical model of a lithium-ion battery I. Determination of parameters. J. Electrochem. Soc. 162 (9), A1836–A1848.

Hertneck, M., Köhler, J., Trimpe, S., Allgöwer, F., 2018. Learning an approximate model predictive controller with guarantees. IEEE Control Syst. Lett. 2 (3), 543–548.

Jazwinski, A.H., 2007. Stochastic Processes and Filtering Theory. Courier Corporation.

Karg, B., Lucia, S., 2020. Efficient representation and approximation of model predictive control laws via deep learning. IEEE Trans. Cybern. 50 (9), 3866–3878.

Klein, R., Chaturvedi, N.A., Christensen, J., Ahmed, J., Findeisen, R., Kojic, A., 2011. Optimal charging strategies in lithium-ion battery. In: 2015 American Control Conference. ACC, IEEE, pp. 382–387.

Lu, L., Han, X., Li, J., Hua, J., Ouyang, M., 2013. A review on the key issues for lithium-ion battery management in electric vehicles. J. Power Sources 226, 272–288.

Moura, S.J., Argomedo, F.B., Klein, R., Mirtabatabaei, A., Krstic, M., 2017. Battery state estimation for a single particle model with electrolyte dynamics. IEEE Trans. Control Syst. Technol. 25 (2), 453–468.

Parisini, T., Zoppoli, R., 1995. A receding-horizon regulator for nonlinear systems and a neural approximation. Automatica 31 (10), 1443–1451.

Park, S., Pozzi, A., Whitmeyer, M., Perez, H., Joe, W.T., Raimondo, D.M., Moura, S., 2020. Reinforcement learning-based fast charging control strategy for li-ion batteries. In: 2020 IEEE Conference on Control Technology and Applications. CCTA, IEEE, pp. 100–107.

Park, S., Pozzi, A., Whitmeyer, M., Perez, H., Kandel, A., Kim, G., Choi, Y., Joe, W.T., Raimondo, D.M., Moura, S., 2022. A deep reinforcement learning framework for fast charging of Li-ion batteries. IEEE Trans. Transp. Electrif..

Patnaik, L., Praneeth, A., Williamson, S.S., 2018. A closed-loop constant-temperature constant-voltage charging technique to reduce charge time of lithium-ion batteries. IEEE Trans. Ind. Electron. 66 (2), 1059–1067.

Perez, H., Dey, S., Hu, X., Moura, S., 2017. Optimal charging of li-ion batteries via a single particle model with electrolyte and thermal dynamics. J. Electrochem. Soc. 164 (7), A1679.

Perez, H.E., Hu, X., Moura, S.J., 2016. Optimal charging of batteries via a single particle model with electrolyte and thermal dynamics. In: 2016 American Control Conference. ACC, IEEE, pp. 4000–4005.

Pozzi, A., Ciaramella, G., Gopalakrishnan, K., Volkwein, S., Raimondo, D.M., 2018a. Optimal design of experiment for parameter estimation of a single particle model for Lithium-ion batteries. In: 2018 51st IEEE Conference on Decision and Control. CDC, pp. 6482–6487.

Pozzi, A., Moura, S., Toti, D., 2022a. Deep learning-based predictive control for the optimal charging of a lithium-ion battery with electrochemical dynamics. In: 2022 IEEE Conference on Control Technology and Applications. CCTA, IEEE, pp. 785–790.

Pozzi, A., Moura, S., Toti, D., 2022b. A neural network-based approximation of model predictive control for a lithium-ion battery with electro-thermal dynamics. In: 2022 IEEE 17th International Conference on Control & Automation. ICCA, IEEE, pp. 160–165.

Pozzi, A., Torchio, M., Raimondo, D.M., 2018b. Assessing the performance of model-based energy saving charging strategies in Li-ion cells. In: 2018 IEEE Conference on Control Technology and Applications. CCTA, IEEE, pp. 806–811.

Pozzi, A., Toti, D., 2022. Lexicographic model predictive control strategy in ageing-aware optimal charging procedure for lithium-ion batteries. Comput. Chem. Eng. 163, 107847.

Pozzi, A., Xie, X., Raimondo, D.M., Schenkendorf, R., 2020. Global sensitivity methods for design of experiments in Lithium-ion battery context. IFAC-PapersOnLine 53 (2), 7248–7255.

Santhanagopalan, S., Guo, Q., Ramadass, P., White, R.E., 2006. Review of models for predicting the cycling performance of lithium ion batteries. J. Power Sources 156 (2), 620–628.

Shen, W., Vo, T.T., Kapoor, A., 2012. Charging algorithms of lithium-ion batteries: An overview. In: 2012 7th IEEE Conference on Industrial Electronics and Applications. ICIEA, IEEE, pp. 1567–1572.

Subramanian, V.R., Diwakar, V.D., Tapriyal, D., 2005. Efficient macro-micro scale coupled modeling of batteries. J. Electrochem. Soc. 152 (10), A2002–A2008.

Tian, N., Fang, H., Wang, Y., 2019. Real-time optimal charging for lithium-ion batteries via explicit model predictive control. In: 2019 IEEE 28th International Symposium on Industrial Electronics. ISIE, IEEE, pp. 2001–2006.

Torchio, M., Wolff, N.A., Raimondo, D.M., Magni, L., Krewer, U., Gopaluni, R.B., Paulson, J.A., Braatz, R.D., 2015. Real-time model predictive control for the optimal charging of a lithium-ion battery. In: 2015 American Control Conference. ACC, IEEE, pp. 4536–4541.

Waag, W., Fleischer, C., Sauer, D.U., 2014. Critical review of the methods for monitoring of lithium-ion batteries in electric and hybrid vehicles. J. Power Sources 258, 321–339.

Wächter, A., Biegler, L.T., 2006. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. Math. Program. 106 (1), 25–57.

Zou, C., Manzie, C., Nešić, D., 2018. Model predictive control for lithium-ion battery optimal charging. IEEE/ASME Trans. Mechatronics 23 (2), 947–957.