2022 IEEE 17th International Conference on Control & Automation (ICCA)
June 27-30, 2022 (Hybrid) Naples, Italy

A Neural Network-Based Approximation of Model Predictive Control for a Lithium-Ion Battery with Electro-Thermal Dynamics

Andrea Pozzi, Scott Moura and Daniele Toti

2022 IEEE 17th International Conference on Control & Automation (ICCA) | 978-1-6654-9572-1/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICCA54724.2022.9831878

Abstract— **Lithium-ion batteries are complex systems that require suitable management strategies to work properly, achieve fast charging, mitigate ageing mechanisms and guarantee safety. Among the different model-based charging strategies, the use of predictive control has shown promising results, due to its ability to deal with nonlinear systems subject to safety constraints. However, although many implementations have been proposed in the literature, little attention has been paid to their practical feasibility, which is limited by the high computational cost required online. In this paper, we exploit, for the first time in the batteries field, an approximation of predictive control obtained through the use of a deep neural network. The proposed solution is suitable for real-time battery charging, due to the fact that most of the computational burden is addressed offline. The results highlight the effectiveness of the presented methodology in approximating a standard model predictive control solution.**

I. Introduction

The proper management of lithium-ion batteries is a complex task that involves the design of suitable charging and discharging current profiles to guarantee safety and mitigate degradation mechanisms [1]. Battery management systems currently employed in the industry mainly rely on rule-based charging algorithms, see e.g. the constant-current/constant-voltage one, which is heuristically designed to achieve reasonable charging time while satisfying voltage constraints [2]. However, it has been shown in the literature that standard charging procedures, which conservatively rely only on voltage bounds, usually lead to battery underutilization and possible safety hazards as the battery ages and its characteristics change [3]. As a possible solution to these issues, the scientific community has focused on the exploitation of a mathematical description of the battery for the development of optimal control methodologies. Among the different model-based strategies, Model Predictive Control (MPC) [4] has been largely adopted in the context of battery management due to its ability to deal with multi-variable nonlinear processes subject to states and input constraints. In particular, the authors in [5] exploit a one-step predictive controller to solve the minimum charging problem, while the works in [6], [7] and [8] aim at finding an optimal trade-off between charging time and battery state-of-health. Moreover, the use of nonlinear MPC for the state-of-charge balancing of series-connected cells is proposed in [9], while the charge of a whole battery pack is addressed in [10] by using a sensitivity-based MPC scheme.

Andrea Pozzi and Daniele Toti are with Faculty of Mathematics, Physics and Natural Science, Catholic University of Sacred Heart, Via della Garzetta, 48, 25133 Brescia BS, Italy (E-mail: {andrea.pozzi, daniele.toti}@unicatt.it).

Scott Moura is with the Energy, Controls and Applications Lab (eCAL) at the University of California, Berkeley, CA 94720, USA (E-mail: smoura@berkeley.edu).

However, although the aforementioned predictive controller formulations have shown successful results in simulation, their reliability in a practical scenario is limited due to computational issues [11]. This is because the core idea of MPC lies on the on-line solution of a constrained optimal control problem at each time step, thus making unsuitable models whose computational cost is not compatible with the sampling time of the control law. Indeed, most of the practical implementations of model predictive control rely on simple linear models with a limited prediction horizon and number of states. As a possible solution to the computational issues, the concept of explicit MPC has been proposed to reduce the on-line operation to a simple function evaluation [12]. In particular, explicit MPC computes the optimal control action off-line as a piecewise function of the state and reference vectors, and hence the only real-time computational cost is to detect in which region the states are located. However, such computational burden still becomes intractable when the number of constraints and the prediction horizon increase, since this corresponds to a high number of regions [13].

To overcome the aforementioned issues, the idea of approximating the predictive control law has been proposed by several authors (see [14], [15], [16] for instances). Among the different approximations, that obtained through deep neural networks have received a lot of attention due to their high representation capability, giving birth to the so-called deep model predictive control framework [17]. It is important to highlight the difference between exploiting a deep neural network as a model for the control within a standard predictive control scheme and the concept of deep MPC. The former approach consists in modelling the plant through a deep neural network (*i.e.* conducting a black-box identification phase) and then solving a constrained optimization problem on-line to retrieve the control action, which appears to be impracticable due to the strong nonlinearities of the identified model. The latter relies on a deep learning model to map the states into the optimal action, after a training process that is conducted off-line on a dataset generated by applying the predictive controller that the procedure aims to approximate. With the deep MPC algorithm, only an evaluation of the neural network is required on-line, thus allowing for a reasonable computational burden. Among the works which deal with deep MPC it is worth mentioning the one conducted by the authors in [18], where a deep MPC robust to input error is developed, and the ones in [13], where a deep predictive controller is proven able to exactly represent an explicit MPC control law with a sufficiently high number of layers and neurons.

Within the battery context, explicit MPC based on a piecewise function for the optimal action computation has been exploited in [19] to achieve real-time optimal charging, while learning-based

charging techniques have been proposed for instance by the authors in [20], [21], where deep reinforcement learning is employed.However, no contributions related to the use of deep MPC are present in the literature.

Therefore, in this paper, we propose for the first time in literature to the authors' best knowledge, a deep predictive controller for the optimal management of a lithium-ion cell, here represented through an electro-thermal model. Suitable neural network architecture is proposed to intrinsically consider the bounds on the input. A comparison with a standard MPC approach is then presented, with a particular focus on the approximation of the optimal control action. The results highlight that the proposed deep MPC can achieve similar performance when compared to the standard approach while reducing the computational burden.

The rest of the paper is organized as follows. In Section II the battery modelling is discussed, while in Section III the deep predictive controller formulation is recalled. The results are presented in Section IV, while Section V concludes the paper.

II. Model

The most used models for a lithium-ion cell usually belong to the following two categories: equivalent circuit models [22] and electrochemical models [23]. The former, which are simple and intuitive, are often used in control schemes because their parameters are easily identifiable and their computational cost is relatively low. The latter, instead, describe in detail the internal phenomena of a lithium-ion cell, such ion diffusion as well as material degradation. However, they exhibit a high computational burden together with identifiability [24], [25] and observability issues [26], and therefore they appear to be more suited for simulation purposes rather than control ones.

In this paper, we adopt the nonlinear equivalent circuit implemented by [27], which presents two RC blocks and it is empowered with two states of thermal dynamics, in order to achieve a reasonable trade-off between a sufficiently realistic description of the cell and computational complexity. Note that the electrical parameters of the considered model are time varying, since they are expressed as a function of the battery states.

A. Equations of the Adopted Model

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

$$\frac{dsoc(t)}{dt} = \frac{I(t)}{3600C} \tag{1}$$

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]. Note that $soc(t)=1$ represents the condition in which the cell is fully charged, while $soc(t)=0$ indicates a complete discharged status. Considering the equivalent circuit model in Figure 1, the output voltage of the cell is given by adding the following terms: the open circuit potential ($V_{\text{ocp}}(t)$), the voltage drop on the series resistance ($V_0(t)=R_0(t)I(t)$) and the voltage of the two RC blocks which represent, respectively, the ions diffusion in the solid phase ($V_1(t)$) and in the electrolyte ($V_2(t)$). In particular it holds that

$$V(t) = V_{\text{ocp}}(t) + V_1(t) + V_2(t) + R_0(t)I(t) \tag{2}$$

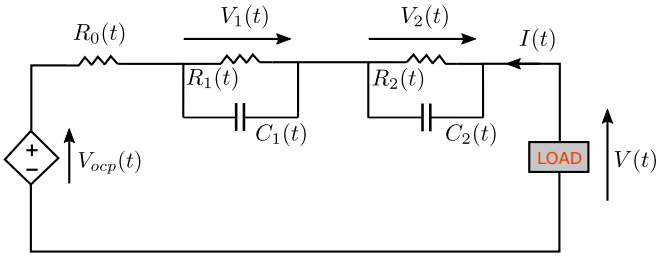

Fig. 1. Circuital scheme of the adopted battery model (also known as Thévenin model).

where $R_0(t)$ is the series resistance and $V_{\text{ocp}}(t)$ is a nonlinear function of the state of charge. The dynamics of the RC blocks are described by

$$\frac{dV_1(t)}{dt} = -\frac{V_1(t)}{R_1(t)C_1(t)} + \frac{I(t)}{C_1(t)} \tag{3a}$$

$$\frac{dV_2(t)}{dt} = -\frac{V_2(t)}{R_2(t)C_2(t)} + \frac{I(t)}{C_2(t)} \tag{3b}$$

where the resistance $R_i(t)$ and the capacitance $C_i(t)$ are equivalent elements used to approximate the ions diffusion phenomena for the i-th RC block.

As far as the thermal model is concerned, we consider two states to describe the core and surface temperature of the cell, $T_c(t)$ and $T_s(t)$, respectively. Specifically, we have that

$$C_c\frac{dT_c(t)}{dt} = Q(t) - \frac{T_c(t)-T_s(t)}{R_{c,s}} \tag{4a}$$

$$C_s\frac{dT_s(t)}{dt} = \frac{T_c(t)-T_s(t)}{R_{c,s}} - \frac{T_s(t)-T_{\text{env}}}{R_{s,e}} \tag{4b}$$

where $Q(t)$ is the generated heat, while C_c and C_s are the heat capacity of the core and the surface of the cell, respectively, $R_{c,s}$ 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 (whose temperature is T_{env}). Although the heat generation is usually modeled as

$$Q(t) = |(V(t) - V_{\text{ocp}}(t))I(t)| \tag{5}$$

here we choose the following approximation in order to avoid the issues related to the use of a control model with a discontinuous derivative:

$$Q(t) \simeq Q_{\text{j}}(t) + Q_{\text{e}}(t) \tag{6}$$

where the term $Q_{\text{j}}(t) = \frac{V_1(t)^2}{R_1(t)} + \frac{V_2(t)^2}{R_2(t)} + R_0(t)I(t)^2$ directly represents the Joule dissipation, while the term $Q_{\text{e}}(t) = R_{\text{e}}I(t)^2$ approximates the heat generated by the electrodes overpotential. The resistance R_{e}, which does not exhibit a physical meaning, is here identified in order to minimize the approximation error.

B. Model Parameters

The parameters of the equivalent circuit model are chosen as in [27] where an electro-thermal model for a $LiFePo_4$ cell *A123 26650* is parameterized and validated, with the electrical parameters dependent on the battery states.

The open circuit potential is modeled as follows

$$V_{\text{ocp}}(t) = p_{l,0} + p_{l,1} soc(t) - \frac{p_{a,1} soc(t) + p_{a,2}}{soc(t)^2 - p_{a,3} soc(t) - p_{a,4}} + \\ + \frac{p_{b,1} soc(t) + p_{b,2}}{soc(t)^2 + p_{b,3} soc(t) + p_{b,4}} \tag{7}$$

with the coefficients fitted from the data depicted in Figure 7 of [28], where the same cell is considered. In particular, the open circuit potential is defined by three terms: a linear function and two hyperbolae, which are able to describe the cell behavior near the state of charge limits. Although the hyperbolic terms are discontinuous with respect to the state of charge, this does not affect the solver operation since such discontinuities are outside the state of charge limits.

The thermal parameters are chosen equal to the ones used by the authors in [27], with the exception of the thermal resistance between surface and external environment which has been here adapted to the case of a cell in the center of a battery pack, $R_{s,e} = 30 KW^{-1}$ (*i.e.* in a less favorable case in terms of the heat dissipation). Finally, the identified value for the resistance used in the approximation of the heat generation is $R_{\text{e}} = 2.5 m\Omega$.

III. Methodology

In this section, the proposed methodology is described. In particular, we start by considering the equations of a general nonlinear model in III-A. Then, in III-B we define the concept of model predictive control, which is used in this paper both for the dataset generation and as a benchmark. Finally, the deep MPC approach is discussed in III-C.

A. General Nonlinear Model

In this paper, the real plant to be controlled is modelled through the following continuous time nonlinear equations:

$$\frac{d\mathbf{x}(t)}{dt} = f(\mathbf{x}(t), \mathbf{u}(t)) \tag{8a}$$

$$\mathbf{y}(t) = g(\mathbf{x}(t), \mathbf{u}(t)) \tag{8b}$$

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

Since in the following we assume 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 (8) is discretized accordingly.

Note that in the rest of the paper we consider the generic input sequence applied in the time interval $[t_k, t_{k+H}]$, with $H \in \mathbb{N}$, as

$$\mathbf{u}_{[t_k, t_{k+H}]} = \left[\mathbf{u}(t_k)^\top \mathbf{u}(t_{k+1})^\top \cdots \mathbf{u}(t_{k+H-1})^\top\right]^\top. \tag{9}$$

B. Model Predictive Control

Model predictive control has proven effective in the management of multi-variable nonlinear processes subject to states and inputs constraints [4]. In particular, MPC provides a control action based on the solution of a constrained finite horizon optimal control problem repeatedly within a *receding horizon* fashion [29]: at each time step t_k, only the first element $\mathbf{u}^\star(t_k)$ of the resulting optimal input sequence $\mathbf{u}^\star_{[t_k, t_{k+H}]} = [\mathbf{u}^\star(t_k) \mathbf{u}^\star(t_{k+1}) \cdots \mathbf{u}^\star(t_{k+H-1})]$ is applied, while the remaining future optimal moves are discarded. The optimal input sequence is obtained by solving the following optimization problem over the horizon H

$$\mathbf{u}^\star_{[t_k, t_{k+H}]} = \underset{\mathbf{u}_{[t_k, t_{k+H}]}}{\operatorname{argmin}} \; J(\mathbf{x}(t_k)) \tag{10a}$$

subject to

system dynamic in (8)

$$\mathbf{u}^{lb} \le \mathbf{u}(t_i) \le \mathbf{u}^{ub}, \qquad i = k, k+1, \cdots, k+H-1 \tag{10b}$$

$$\mathbf{x}^{lb} \le \mathbf{x}(t_i) \le \mathbf{x}^{ub}, \qquad i = k+1, k+1, \cdots, k+H \tag{10c}$$

$$\mathbf{y}^{lb} \le \mathbf{y}(t_i) \le \mathbf{y}^{ub}, \qquad i = k+1, k+1, \cdots, k+H \tag{10d}$$

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

$$J(\mathbf{x}(t_k)) = \sum_{i=k+1}^{k+H} \left[\|\mathbf{x}(t_i) - \mathbf{x}_{\text{ref}}\|^2_{Q_x} + \|\mathbf{y}(t_i) - \mathbf{y}_{\text{ref}}\|^2_{Q_y} \right] + \\ + \sum_{i=k}^{k+H-1} \|\mathbf{u}(t_i) - \mathbf{u}_{\text{ref}}\|^2_R \tag{10e}$$

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 $Q_x \in \mathbb{R}^{n_x \times n_x}$, $Q_y \in \mathbb{R}^{n_y \times n_y}$ and $R \in \mathbb{R}^{n_u \times n_u}$ are design parameters, with $Q_x, Q_y \ge 0$ and $R > 0$.

In the following we will define the MPC control law as follows as

$$\mathbf{u}_{\text{mpc}}(\mathbf{x}(t_k)) = \mathbf{u}(t_k)^\star \tag{11}$$

which is computed by solving the problem in (10) for the states vector $\mathbf{x}(t_k)$.

C. Deep Model Predictive Control

The core idea behind deep MPC is to approximate the feedback law through the use of deep neural networks, as a possible alternative to the use of explicit MPC formulations. These latter rely on the representation of the control law as a piecewise function, and the only computational requirement for their on-line application consists in finding the polytopic region in which the states are currently located. However, it has been shown that the number of regions increases with the prediction horizon and the number of constraints, making explicit MPC reliable only for small linear systems. Although the use of neural networks as approximators for the MPC control law was proposed for the first time by [30], only recently they have become a real opportunity for the development of fast MPC methodologies, especially due to new advances on the theoretical description of neural network representation capabilities. In particular, the work in [18] has formulated a deep MPC framework with guarantees on robustness against inaccurate inputs, while the authors in [13] have derived bounds for the required size (width and depth) that a deep neural network should have to exactly represent a given linear MPC control action. In the following, we discuss the deep MPC formulation adopted in this work.

Consider a neural network $\mathcal{N}=\mathcal{N}(\mathbf{x},\mathbf{r},\theta)$ composed by an input layer with n_x neurons, $N_h \in \mathbb{N}$ fully connected layers with *ReLu* activation function, each of them constituted of $n_{h,j}$ number of neurons, with $j=1,2,...,N_h$, and an output layer with n_u neurons and activation function *tanh*. The neural network parameters vector is represented by θ, while $\mathbf{r}$ represents the vector of the MPC references, *i.e.* $\mathbf{r}=[\mathbf{x}_{\text{ref}}^\top \mathbf{y}_{\text{ref}}^\top \mathbf{u}_{\text{ref}}^\top]^\top$. The training data are obtained by solving (10) for n_{tr} different states and references samples, *i.e.* $\mathbf{x}_{\text{tr},i}$ and $\mathbf{r}_i$ respectively for $i=1,2,...,n_{\text{tr}}$, and storing the tuples $(\mathbf{x}_{\text{tr},i},\mathbf{r}_i,\bar{\mathbf{u}}_{\text{mpc}}(\mathbf{x}_{\text{tr},i}))$ in the dataset $\mathcal{B}_{\text{tr}}$, where $\bar{\mathbf{u}}_{\text{mpc}}(\cdot)$ is the MPC control action normalized with respect to the input bounds such that $\bar{\mathbf{u}}_{\text{mpc}}(\cdot) \in [-1,1]$. The neural network $\mathcal{N}$ is then trained off-line on the dataset $\mathcal{B}_{\text{tr}}$ by solving the following optimization through the back-propagation method:

$$\theta^\star = \underset{\theta}{\operatorname{argmin}} \frac{1}{n_{\text{tr}}} \sum_{i=0}^{n_{\text{tr}}} \|\bar{\mathbf{u}}_{\text{mpc}}(\mathbf{x}_{\text{tr},i}) - \mathcal{N}(\mathbf{x}_{\text{tr},i},\mathbf{r}_i,\theta)\|_2^2 \qquad (12)$$

where the loss function is the mean squared error between the network prediction and the target.

IV. Simulative Results

In this section, the results of the comparison between standard and deep MPC is shown for the case of battery charging. In particular, we first define the concept of optimal charging of a lithium-ion cell in IV-A, while in IV-B the training phase of the neural network is presented. Then, in IV-C the results of the simulation are discussed. Finally, in IV-D details on the software implementation are provided.

A. Optimal Battery Charging

The optimal management of a battery involves the tracking of a reference state of charge while minimizing the control effort and satisfying safety constraints on voltage and temperature. First, we express the battery dynamics (see equations (1)- (7)) through the model in (8), where the input is chosen as the applied current. Then, the constrained optimization in 10, which needs to be solved on-line at each time step t_k by the MPC, is adapted as follows:

$$\min_{I_{[t_k,t_{k+H}]}} q_{\text{soc}} \sum_{i=k+1}^{k+H} (soc(t_k) - soc_{\text{ref}})^2 + r \sum_{i=k}^{k+H-1} I(t_k)^2 \qquad (13a)$$

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

$$\text{battery dynamics in (1)-(7)}$$
$$0 \le I(t_i) \le 10A, \qquad 0 \le soc(t_i) \le 1 \qquad (13b)$$
$$T_c(t_i) \le 313.15K, \qquad V(t_i) \le 3.6V \qquad (13c)$$

with $q_{\text{soc}}=1$, $r=10^{-5}$ and $H=5$. Moreover, the sample time is taken as $t_s=10s$. Note that the state of charge reference (soc_{ref}) remains symbolic here since its value depends on the specific charging preferences, unlike the current, temperature and voltage constraints that are determined by the battery manufacturer. Furthermore, when generating the dataset for the training of the deep MPC, the state of charge reference is extracted with uniform probability within a specific interval to allow the deep predictive controller to learn how to charge the battery for each state of charge value in that interval. On the other hand, the current reference, *i.e.* the input reference, is always taken as $I_{\text{ref}}=0$ since the system is marginally stable.

It is important to notice that, although the internal states of the cell are not measurable in practice, in this paper we draw the assumption of availability of all the relevant states. Therefore, in a practical scenario, a suitable observer needs to be developed and coupled with the presented control scheme. The observer design, however, goes beyond the scope of this paper and will be objective of future work.

B. Dataset Generation and Training Phase

One of the crucial points in the development of a suitable approximation of MPC through deep neural networks is the dataset generation phase. First, we need to implement an MPC controller according to the settings specified in IV-A. Then 2000 simulations are executed to generate the training dataset, in which each sample is constituted by the tuple $([soc_i V_{1,i} V_{2,i} T_{c,i} T_{s,i}]^\top, soc_{\text{ref},j}, \bar{I}_i^\star)$, which fully describes the i-th time step of the j-th simulation. For each simulation j, an initial condition is randomly extracted, and then for each step i the normalized optimal current $\bar{I}_i^\star$ is computed by solving the optimization problem in (13) for $soc_{\text{ref},j}$, which is also randomly extracted. Note that, in each simulation, we compute the evolution of the battery dynamics until 50 steps are executed or the state of charge achieves a neighborhood of the target, by applying as input the computed optimal current with an additive Gaussian noise to enhance exploration. Such noise in particular is chosen with a zero mean and a standard deviation of $2A$.

The considered deep learning model consists of a feed-forward network with 6 hidden layers (three couples of layers with 100, 50 an 10 neurons, respectively). The optimizer used during training is the well known *Adam* optimizer (with learning rate $l_r=0.0005$), which is a stochastic gradient descent method that is based on adaptive estimation of first-order and second-order moments

C. Results Discussion

In this section, we describe the results of the performed simulations. In particular, we graphically show in figures 2-5 the comparison between the performance of the standard predictive controller (solid line) and the deep MPC (crossed line) in charging the battery in two different simulation settings. Specifically, in blue we indicate the first simulation (*sim 1*) with $soc_{\text{ref}}=0.8$ and initial states given by: $soc(t_0)=0.3$, $V_1(t_0)=V_2(t_0)=0$ and $T_s(t_0)=T_c(t_0)=301.15K$. In red we represent the second simulation (*sim 2*) with $soc_{\text{ref}}=0.95$ and initial states given by: $soc(t_0)=0.05$, $V_1(t_0)=V_2(t_0)=0$ and $T_s(t_0)=T_c(t_0)=298.15K$.

The profiles depicted in figures 2-5 highlight the fact that the trajectories of the system controlled by the two methodologies are very similar. In particular, we can see from Figure 2 that both the controllers can track the state of charge references, while Figure 3 and Figure 4 show that both the constraints on voltage and temperature are always satisfied. Moreover, in Figure 5 the applied current is shown, where the constraints are satisfied by design for the deep MPC, while in the case of the standard predictive controller they are imposed as bounds on the input.

It is important to notice that similar performance can be obtained for all the initial conditions and references which belong to the intervals considered during the training phase, which can be

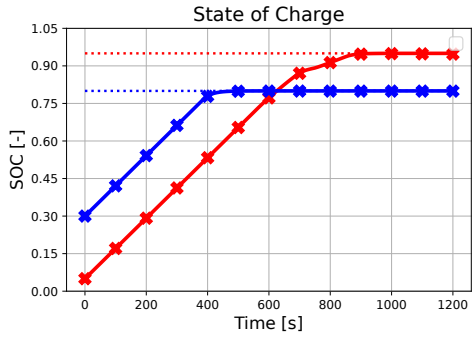

Fig. 2. Comparison of the state of charge profiles when MPC (solid line) and deep MPC (crossed line) are applied. Two simulations are considered with different references (dotted line) and initial states. The blue color is used to represent *sim 1*, while the red is used for *sim 2*.

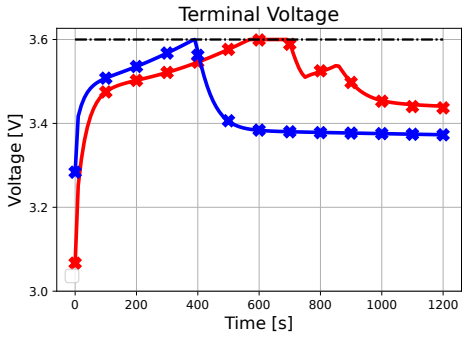

Fig. 3. Comparison of the voltage profiles when MPC (solid line) and deep MPC (crossed line) are applied.

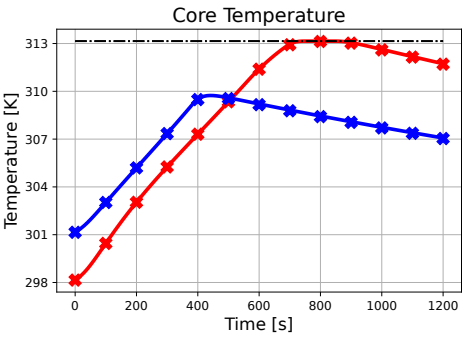

Fig. 4. Comparison of state of the core temperature profiles when MPC (solid line) and deep MPC (crossed line) are applied..

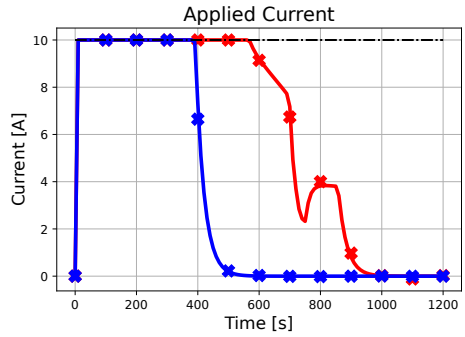

Fig. 5. Comparison of the applied current profiles when MPC (solid line) and deep MPC (crossed line) are applied.

made large enough to take into account the whole span of realistic scenarios. Nevertheless, we expect that satisfying performance of the deep MPC can be achieved also for unseen situations thanks to the generalization capability of the neural network. To demonstrate this statement we compute the mean and standard deviation of the difference between standard and deep MPC current profiles over 10 simulations with randomly extracted initial states and state of charge reference. The results are depicted in Figure 6, which provides a statistical description of the errors between the input current applied in closed loop by the two control schemes. As it can be noticed, the histogram representing the current error occurrences exhibits a bell shape with mean $6.70mA$ and standard deviation $237mA$, while the minimum and maximum errors are $-3.94A$ and $2.34A$, which however correspond only to isolated cases. Specifically, most of the errors that the deep MPC commits in tracking the standard predictive controller trajectory happen when the system states are approaching for the first time in a region which is in proximity of the constraints or near the reference point. In fact, in these situations the deep MPC algorithm presents small oscillations before stabilizing around the optimal value. A possible solution to this issue may be the enrichment of the training dataset with tuples corresponding to the aforementioned conditions.

Finally, the average on-line computational cost of the two methodologies is $45.4ms$ for the deep MPC (with standard deviation of $5.56ms$ and maximum value of $60.3ms$) and $110ms$ for the standard predictive controller (with standard deviation of $53.0\,ms$ and maximum value of $344\,ms$) . This difference is further accentuated if we consider a longer horizon. In fact it is known that the computational cost of the MPC exhibits a superlinear increase as the horizon grows, unlike that required by the deep algorithm, whose online computational cost depends only on the structure of the neural network, which is not affected by variations of the prediction horizon. Specifically, if one considers a horizon $H=10$, the average computational cost remains around $45.2\,ms$ for the deep MPC, while for the standard predictive controller becomes $327ms$.

D. Implementation Details

The presented framework is implemented in Python 3.7 and the simulations have been performed on a Windows 10 personal computer with 16 GB of RAM and a i7-8750H processor. We rely on *CasADi* [31] for the integration of the model equations as well as the solution of the optimization problems. In particular, the Runge-Kutta method is used to simulate the model, while we exploit the implementation of the interior-point method provided by the IPOPT solver for the optimization [32]. As far as the deep learning model is concerned, we rely on *TensorFlow 2.0* [33], which is a well known framework for large-scale machine learning.

V. Conclusions

In this paper, the concept of deep model predictive control has been applied for the first time to the optimal charging of a lithium-ion battery. The strength of such methodology lies in its ability to approximate the predictive control, thanks to the

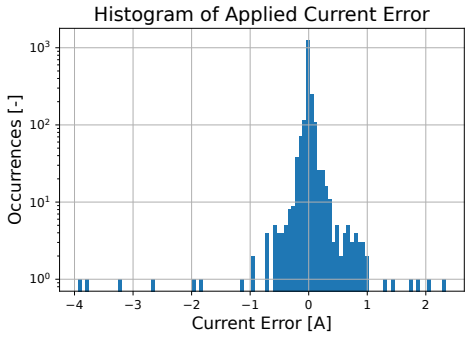

Fig. 6. Histogram with 100 bins describing the differences between the current applied in closed loop by the two considered methodologies. Note that the y-axis in logarithmic scale.

representation capabilities of neural networks, thus reducing the real-time computational cost. As it is shown by the presented results, the state and input profiles obtained by applying the deep MPC approach coincide with those of the standard methodology, except for an approximation error which is negligible from an application point of view. It is important to notice that proper dataset generation is required to enhance the generalization capabilities of the deep-learning algorithm.

REFERENCES

[1] L. Lu, X. Han, J. Li, J. Hua, and M. Ouyang, "A review on the key issues for lithium-ion battery management in electric vehicles," vol. 226, pp. 272–288, 2013.

[2] W. Shen, T. T. Vo, and A. Kapoor, "Charging algorithms of lithium-ion batteries: An overview," in *2012 7th IEEE conference on industrial electronics and applications (ICIEA)*. IEEE, 2012, pp. 1567–1572.

[3] N. A. Chaturvedi, R. Klein, J. Christensen, J. Ahmed, and A. Kojic, "Algorithms for advanced battery-management systems," *IEEE Control Systems*, vol. 30, no. 3, pp. 49–68, 2010.

[4] E. F. Camacho and C. B. Alba, *Model Predictive Control*. Springer Science & Business Media, 2013.

[5] R. Klein, N. A. Chaturvedi, J. Christensen, J. Ahmed, R. Findeisen, and A. Kojic, "Optimal charging strategies in lithium-ion battery," in *2015 American Control Conference (ACC)*. IEEE, 2011, pp. 382–387.

[6] A. Pozzi, M. Torchio, and D. M. Raimondo, "Film growth minimization in a li-ion cell: a pseudo two dimensional model-based optimal charging approach," in *2018 European Control Conference (ECC)*. IEEE, 2018, pp. 1753–1758.

[7] C. Zou, C. Manzie, and D. Nešić, "Model predictive control for lithium-ion battery optimal charging," *IEEE/ASME Transactions on Mechatronics*, vol. 23, no. 2, pp. 947–957, 2018.

[8] A. Pozzi, M. Torchio, and D. M. Raimondo, "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, 2018, pp. 806–811.

[9] A. Pozzi, M. Zambelli, A. Ferrara, and D. M. Raimondo, "Balancing-aware charging strategy for series-connected lithium-ion cells: A nonlinear model predictive control approach," *IEEE Transactions on Control Systems Technology*, vol. 28, no. 5, pp. 1862–1877, 2020.

[10] A. Pozzi, M. Torchio, R. D. Braatz, and D. M. Raimondo, "Optimal charging of an electric vehicle battery pack: A real-time sensitivity-based model predictive control approach," *Journal of Power Sources*, vol. 461, p. 228133, 2020.

[11] A. S. Kumar and Z. Ahmad, "Model predictive control (mpc) and its current issues in chemical engineering," *Chemical Engineering Communications*, vol. 199, no. 4, pp. 472–511, 2012.

[12] A. Alessio and A. Bemporad, "A survey on explicit model predictive control," in *Nonlinear model predictive control*. Springer, 2009, pp. 345–369.

[13] B. Karg and S. Lucia, "Efficient representation and approximation of model predictive control laws via deep learning," *IEEE Transactions on Cybernetics*, vol. 50, no. 9, pp. 3866–3878, 2020.

[14] T. Parisini and R. Zoppoli, "A receding-horizon regulator for nonlinear systems and a neural approximation," *Automatica*, vol. 31, no. 10, pp. 1443–1451, 1995.

[15] A. Bemporad, A. Oliveri, T. Poggi, and M. Storace, "Ultra-fast stabilizing model predictive control via canonical piecewise affine approximations," *IEEE Transactions on Automatic Control*, vol. 56, no. 12, pp. 2883–2897, 2011.

[16] L. H. Csekő, M. Kvasnica, and B. Lantos, "Explicit mpc-based rbf neural network controller design with discrete-time actual kalman filter for semiactive suspension," *IEEE Transactions on Control Systems Technology*, vol. 23, no. 5, pp. 1736–1753, 2015.

[17] S. Chen, K. Saulnier, N. Atanasov, D. D. Lee, V. Kumar, G. J. Pappas, and M. Morari, "Approximating explicit model predictive control using constrained neural networks," in *2018 Annual American control conference (ACC)*. IEEE, 2018, pp. 1520–1527.

[18] M. Hertneck, J. Köhler, S. Trimpe, and F. Allgöwer, "Learning an approximate model predictive controller with guarantees," *IEEE Control Systems Letters*, vol. 2, no. 3, pp. 543–548, 2018.

[19] N. Tian, H. Fang, and Y. Wang, "Real-time optimal charging for lithium-ion batteries via explicit model predictive control," in *2019 IEEE 28th International Symposium on Industrial Electronics (ISIE)*. IEEE, 2019, pp. 2001–2006.

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

[21] S. Park, A. Pozzi, M. Whitmeyer, H. Perez, A. Kandel, G. Kim, Y. Choi, W. T. Joe, D. M. Raimondo, and S. Moura, "A deep reinforcement learning framework for fast charging of li-ion batteries," *IEEE Transactions on Transportation Electrification*, 2022.

[22] X. Hu, S. Li, and H. Peng, "A comparative study of equivalent circuit models for Li-ion batteries," *Journal of Power Sources*, vol. 198, pp. 359–367, 2012.

[23] P. M. Gomadam, J. W. Weidner, R. A. Dougal, and R. E. White, "Mathematical modeling of lithium-ion and nickel battery systems," vol. 110, no. 2, pp. 267–284, 2002.

[24] A. Pozzi, G. Ciaramella, K. Gopalakrishnan, S. Volkwein, and D. M. Raimondo, "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)*, 2018, pp. 6482–6487.

[25] A. Pozzi, X. Xie, D. M. Raimondo, and R. Schenkendorf, "Global sensitivity methods for design of experiments in lithium-ion battery context," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 7248–7255, 2020.

[26] S. J. Moura, "Estimation and control of battery electrochemistry models: A tutorial," in *2015 54st IEEE Conference on Decision and Control (CDC)*, 2015, pp. 3906–3912.

[27] H. E. Perez, J. B. Siegel, X. Lin, A. G. Stefanopoulou, Y. Ding, and M. P. Castanier, "Parameterization and validation of an integrated electro-thermal cylindrical lfp battery model," in *Dynamic Systems and Control Conference*, vol. 45318. American Society of Mechanical Engineers, 2012, pp. 41–50.

[28] X. Lin, H. E. Perez, S. Mohan, J. B. Siegel, A. G. Stefanopoulou, Y. Ding, and M. P. Castanier, "A lumped-parameter electro-thermal model for cylindrical batteries," *Journal of Power Sources*, vol. 257, pp. 1–11, 2014.

[29] A. Bemporad and M. Morari, "Robust model predictive control: A survey," in *Robustness in Identification and Control*. Springer, 1999, pp. 207–226.

[30] B. M. Åkesson and H. T. Toivonen, "A neural network model predictive controller," *Journal of Process Control*, vol. 16, no. 9, pp. 937–946, 2006.

[31] J. A. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "Casadi: a software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, no. 1, pp. 1–36, 2019.

[32] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical programming*, vol. 106, no. 1, pp. 25–57, 2006.

[33] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, "Tensorflow: A system for large-scale machine learning," in *12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16)*, 2016, pp. 265–283.