An Online Learning Method for Microgrid Energy Management Control*

Vittorio Casagrande[1], Martin Ferianc[1], Miguel Rodrigues[1] and Francesca Boem[1]

***Abstract*— We propose a novel Model Predictive Control (MPC) scheme based on online-learning (OL) for microgrid energy management, where the control optimisation is embedded as the last layer of the neural network. The proposed MPC scheme deals with uncertainty on the load and renewable generation power profiles and on electricity prices, by employing the predictions provided by an online trained neural network in the optimisation problem. In order to adapt to possible changes in the environment, the neural network is online trained based on continuously received data. The network hyperparameters are selected by performing a hyperparameter optimisation before the deployment of the controller, using a pretraining dataset. We show the effectiveness of the proposed method for microgrid energy management through extensive experiments on real microgrid datasets. Moreover, we show that the proposed algorithm has good transfer learning (TL) capabilities among different microgrids.**

I. INTRODUCTION

Microgrids are small-scale power systems consisting of an interconnection of loads, generators and storage systems operating together to reliably supply electricity to local consumers. The uncertainty of renewable energy sources, as well as dynamic and stochastic load demand pose a major challenge to the scheduling of the microgrid operation. The Energy Management System (EMS) is the controller that computes the power flows in order to provide stable delivery of power to loads while achieving other operational goals, such as economic benefits. Model Predictive Control (MPC) has been widely used in the literature to deal with scheduling problems, e.g. in finance and for EMSs [3, 18], due to its ability to enforce constraints and compensate for uncertainties. In the EMS application, MPC requires accurate predictions of system variables such as the load power profile. These profiles are often difficult to predict and may change over time, for example due to unforeseen circumstances such as revamping of some elements. In this paper, we present a learning-based MPC control for microgrids energy management, adaptable to changes in the environment.

Related work. MPC has been often adopted in the literature for energy management purposes [27]. In this context, it requires estimate of power and price profiles; therefore it is often combined with a predictor forecasting these quantities. In [16] least-square support vector machine regression is used to predict the unknown profiles. Neural networks (NNs) are adopted in [21, 25] to obtain predictions of renewable power, load demand and to compute the optimal energy scheduling. The main drawback of the methods proposed in [16, 21, 25] is that the predictors used in those works are trained offline on a dataset composed of past observations and subsequently the controller is deployed to the real system, thus not being able to update the prediction model where potential changes arise, such as aging, installation of new equipment or sensors calibration [10]. Moreover, it is assumed that a sufficiently long recording of the past power profiles is available in the design phase. Reinforcement Learning (RL) provides an answer to online model adaptation and there are several proposed methods in the literature for the energy management application. In [14, 22] RL is used to compute the schedule of a battery in a microgrid system while dealing with the stochastic nature of load demands, renewable generators and electricity price. In [19] RL is proposed as an alternative to MPC not requiring a prediction model. Although RL is able to adapt to possible changes in the system, there is in general no formal guarantee that it may not drive the system state to unsafe conditions [24]. In this paper, the integration of OL algorithms with MPC allows to overcome the aforementioned issues, by adapting the learnt model and guaranteeing safety and constraints satisfaction. There are several implementations of OL in the literature which do not employ the obtained prediction for control purposes. The authors of [23] propose an ensemble that combines offline and online learning (OL) used for load forecasting using passive-aggressive regression. In [10] a Long Short-Term Memory (LSTM) model that is learnt online together with the tuning of the hyperparameters is proposed, but not allowing changes in the NN structure, for load prediction. Similarly in [2] a method for online load forecasting with uncertainty estimation is proposed. In [7] a predictor is proposed for renewable energy generation characterized by a high computational overhead since it is retrained at the end of each day with all available data.

Contributions. In this paper, we address the challenge of dealing with uncertainty in microgrid systems under time-varying conditions, by designing an innovative OL-based control algorithm for microgrid energy management based on a LSTM NN and MPC. In particular, the NN is trained to forecast the future profiles of the renewable generator, load and electricity price given their past values. The predictions are employed by the MPC optimisation layer to compute the amount of energy to be stored in the storage system

*This work has been supported by the Engineering and Physical Sciences Research Council (grants references: EP/W024411/1 and EP/R513143/1). Martin Ferianc was sponsored through a scholarship from the Institute of Communications and Connected Systems at UCL.

[1]Department of Electronic and Electrical Engineering, University College London, London, United Kingdom {vittorio.casagrande.19,martin.ferianc.19, m.rodrigues,f.boem}@ucl.ac.uk

to minimise the cost due to energy trading. This work builds on the preliminary work, presented in [5], where the EMS design problem is addressed by embedding the optimisation problem as the last NN layer [1] and training the NN, online, end-to-end to optimise the ultimate control performance instead of the prediction accuracy. In this paper we further explore the capabilities of the OL algorithm by designing a platform which allows to: (i) rigorously define pre-training, validation and online training datasets; (ii) define different loss functions for the validation and online training phase, considering for example profiles prediction mean squared error (MSE), control task performance or a combination of both; (iii) optimise learning hyperparameters, including the structure of the layers of the NN. Moreover, we design an online Stochastic Weight Averaging (SWA) algorithm to improve the generalization performance of the NN over time and make the NN more robust to changing conditions or data outliers. We also explore the Transfer Learning (TL) [26] capabilities of the algorithm. TL reuses knowledge from a pre-trained model on a related problem to save time, resources and data. In fact, in a real-world scenario, the designer may not have access to data for each specific microgrid beforehand during design phase. Extensive simulation results on real microgrid datasets show the effectiveness of the proposed architecture. We first demonstrate the advantages of performing the training online, then we show that our algorithm has good TL capabilities from one microgrid dataset to another. Finally, we show that when considering a novel microgrid system, by training online a NN predictor that has already been trained on a different microgrid dataset allows us to achieve overall lower total electricity cost. The OL-based MPC code is available on GitHub `https://github.com/vittpi/ol-ems`.

The remainder of the paper is structured as follows. Section II describes the microgrid model and EMS. Section III gives an overview of the OL-based control algorithm, the online SWA and the hyperparameters optimisation. In Section IV we present the simulation results on different microgrid datasets. In Section V we draw the conclusions.

Notation. We use subscripts to denote time instants, i.e. v_t is the vector v at time t. We denote $v_{k|t}$ the value of the variable v, k steps ahead of the time step t (i.e. at $t+k$). The estimation of the variable v is denoted as $\hat{v}$. So, the estimation of the variable v, available at time t, k steps ahead of time step t (i.e. at $t+k$) is denoted as $\hat{v}_{k|t}$. We use bold variables to denote time sequences of N samples, namely $\mathbf{v}_{N|t} = \{v_{k|t}\}_{k\in\{0,\dots,N-1\}}$ is the sequence v from time step t to $t+N-1$.

II. ENERGY MANAGEMENT SYSTEM ARCHITECTURE

We start by giving an overview of the model of the microgrid in Fig. 1. We then describe the online learning-based controller which is composed of an online trained NN predictor and an MPC optimisation layer.

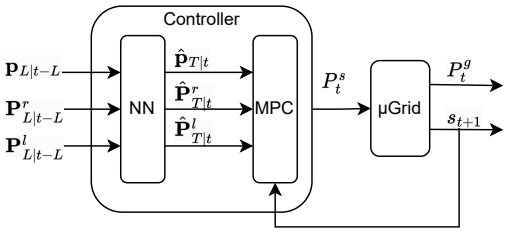

Fig. 1. The architecture of the EMS controller and the microgrid system. The controller uses a NN to predict the future profiles of the electricity price $\hat{\mathbf{p}}_{T|t}$, the load $\hat{\mathbf{P}}^l_{T|t}$ and the renewable generator $\hat{\mathbf{P}}^r_{T|t}$ based on their past values over the look-back window ($\mathbf{p}_{L|t-L}$, $\mathbf{P}^l_{L|t-L}$ and $\mathbf{P}^r_{L|t-L}$). The predictions are used by MPC to compute the power exchanged with the storage system P^s_t. The storage state is then fed back to the MPC.

A. Microgrid model

In this Section we detail all the agents connected to the microgrid, the block μGrid of Fig. 1, for more details see [6]. The renewable generators are modelled as power sources, collectively producing the power P^r_t at each time step in the microgrid. The loads are modelled as power sinks, collectively drawing the power P^l_t at each time step. We assume that both the loads and renewable generators cannot adjust the power they exchange with the microgrid. The method can be easily extended to cover this case. The storage is modelled as a first-order linear system [18]:

$$s_{t+1} = (1-\sigma)s_t + \eta_{c/d} T_s P^s_t \tag{1}$$

where s_t is the charge level at time t, P^s_t is the power exchanged with the microgrid, $\sigma \in [0,1]$ is the self-discharge rate of the battery, $\eta_{c/d}$ is the energy conversion efficiency of the battery for charging/discharging ($\eta_c < 1$ and $\eta_d > 1$) and T_s is the controller sampling time. The power and the charge level of the storage are constrained as follows:

$$-P^s_M \le P^s_t \le P^s_M, \quad s_m \le s_t \le s_M \tag{2}$$

where P^s_M is the maximum power that can be exchanged with the microgrid, s_M is the maximum charge level of the battery and s_m is the minimum charge level of the battery. We assume the microgrid is connected to the utility grid and it exchanges a power P^g_t at each time step. All agents are coupled by the power balance constraint:

$$P^s_t + P^g_t = P^r_t - P^l_t \tag{3}$$

where it is assumed that P^l_t and P^r_t are positive, P^s_t is positive when the storage system is charging and P^g_t is positive when power is sold to the utility grid. The goal of the EMS is to minimise the energy cost due to energy trading with the utility grid, in mathematical terms:

$$\sum_{t=0}^{\infty} -p_t P^g_t \tag{4}$$

where p_t is the price of the energy at time t.

B. Neural network predictor

The proposed controller is based on a NN that is trained to predict the future profiles of the electricity price, of the load demand and of the renewable generator production given their past values defined by a look-back window of length L. In particular, we employ a standard LSTM-based recurrent NN [12] (RNN) with a hidden size H and a number of layers n_{layers}. We consider RNN in comparison to other types of NNs because of their ability to capture dependencies in time-series data, while having a relatively small number of trainable parameters and a flexible input time window. We design a single RNN for jointly forecasting all three uncertain time series: load demand, renewable power generation and electricity price profiles, so that the RNN is able to learn correlations between the variables which can lead to richer representations and more accurate predictions. We are aware that different network architecture compositions can affect the prediction performance, for example, using a separate NN to predict each profile or input the NN's profile prediction into the next one in a cascaded fashion. Nevertheless, our aim is to develop a general framework that can be applied even to other architectures of networks. At each time step we feed the RNN with the past values of the electricity price $\mathbf{p}_{L|t-L}$, the load $\mathbf{P}^l_{L|t-L}$ and the renewable generator $\mathbf{P}^r_{L|t-L}$ over the look-back window of length L and output the predictions $\hat{\mathbf{p}}_{T|t}$, $\hat{\mathbf{P}}^l_{T|t}$ and $\hat{\mathbf{P}}^r_{T|t}$ over the MPC prediction horizon of length T. The prediction over the time horizon is obtained by collecting the last hidden state of the RNN and feeding it to a fully connected layer with a linear activation function. The NN output dimension is $T\times 3$, containing the predictions of the profiles over the MPC prediction horizon, while the input dimension is $L \times 3$.

C. Model predictive control

The controller computes the power that is bought from the utility grid and the energy to store in the storage system such that the total operating cost (4) of the microgrid is minimised. At each time step the optimisation problem is reformulated for the next optimisation horizon given the feedback measures from the microgrid and the predictions of the unknown profiles. The resulting problem is:

$$\min_{\mathbf{P}^s_t} \quad \sum_{k=0}^{T-1} -\hat{p}_{k|t} P^g_{k|t} \tag{5a}$$

$$\text{s.t.} \quad s_{k+1|t} = (1-\sigma)s_{k|t} + \eta_{c/d} T_s P^s_{k|t} \tag{5b}$$

$$-P^s_M \leq P^s_{k|t} \leq P^s_M \tag{5c}$$

$$s_m \leq s_{k|t} \leq s_M \tag{5d}$$

$$P^g_{k|t} + P^s_{k|t} = \hat{P}^r_{k|t} - \hat{P}^l_{k|t} \tag{5e}$$

$$s_{0|t} = s_t \tag{5f}$$

where T is the MPC prediction horizon and $\hat{p}_{k|t}$, $\hat{P}^l_{k|t}$ and $\hat{P}^r_{k|t}$ are respectively the k steps-ahead predictions the price, the load and the renewable generator. The objective function (5a) is the finite horizon approximation of the cost (4) computed using the predicted price profile over the next T steps. The constraints (5b)-(5d) represent the storage dynamics (1) and the storage power and charge limits (2). Constraint (5e) is the power balance (3). Lastly (5f) is the feedback. The decision variable $\mathbf{P}^s_{T|t}$ is the future power schedule of the storage system predicted at time t. Once computed $\mathbf{P}^{s,\star}_{T|t}$, the control law is defined as $P^s_t = P^{s,\star}_{0|t}$.

The storage system state at $t+1$ and power exchanged with the utility grid at time t are computed using (1) and (3):

$$s_{t+1} = (1-\sigma)s_t + \eta T_s P^{s,\star}_{0|t} \tag{6}$$

$$P^g_t = P^r_t - P^l_t - P^{s,\star}_{0|t} \tag{7}$$

where P^r_t and P^l_t are the actual powers produced by the renewable generator and drawn from the load respectively.

III. ONLINE TRAINING AND VALIDATION ALGORITHM

In this Section, we present the online training algorithm and the procedure for hyperparameter optimisation.

A. Online training

The weights of the NN are updated at each step using the OL algorithm. The NN outputs a prediction of the values of the unknown profiles over T steps given their past L values. We now define a training set, validation set and a test set to be used at each time step. We denote the input tensor of the NN at time step t as $\psi^*_t \in \mathbb{R}^{B^*\times L\times F_{\text{in}}}$ where $* \in \{tr, val, te\}$ is the training, validation or test data, B^* is the size of the batch of data used for $*$ and F_{in} is the number of input features. In this work, by *feature* we mean the profile of price, load or renewable. In the general case, a feature can be any piece of data that improves the prediction performance. The corresponding output tensor is denoted as $\omega^*_t \in \mathbb{R}^{B^*\times T\times F_{\text{out}}}$, where F_{out} is the number of output features; for example, $F_{\text{out}} = 3$ if we predict the price, the load and the renewable generator profiles. The test targets ω^{test}_t are not required for learning purposes at time step t since they are only used to assess the prediction performance. By setting $B^{test} = 1$, we define the test input and output tensors for feature f as $\psi^{test}_t = \left[\mathbf{f}_{L|t-L}\right]$, $\omega^{test}_t = \left[\mathbf{f}_{T|t}\right]$. Hence, at time step t, the NN predicts the future values of the unknown profiles $\hat{\omega}^{test}_t$. Such values are passed to the optimiser to compute the current control action. As opposed to the test targets, the training and validation targets are required to update the network weights and hyperparameters. Hence the training and validation samples must be selected among the past values. The training and validation input and output tensors are defined as $\psi^{val}_t = \left[\mathbf{f}_{L|t-T-L-i}\right]$, $\omega^{val}_t = \left[\mathbf{f}_{T|t-T-i}\right]$ and $\psi^{tr}_t = \left[\mathbf{f}_{L|t-T-L-B^{val}-j}\right]$, $\omega^{tr}_t = \left[\mathbf{f}_{T|t-T-B^{val}-j}\right]$, where $i = 1, \ldots, B^{val}-1$ and $j = 1, \ldots, B^{tr}-1$. Since ω^{tr}_t and ω^{val}_t are composed only of past samples, these tensors are available at time t and thus can be used for learning purposes. The network can be trained by minimising any properly designed loss function, such as (i) the standard prediction mean square error (MSE) loss function, or (ii) the control cost function, or (iii) a combination of these, as proposed in [5]. At time step $t+1$ the training, validation and

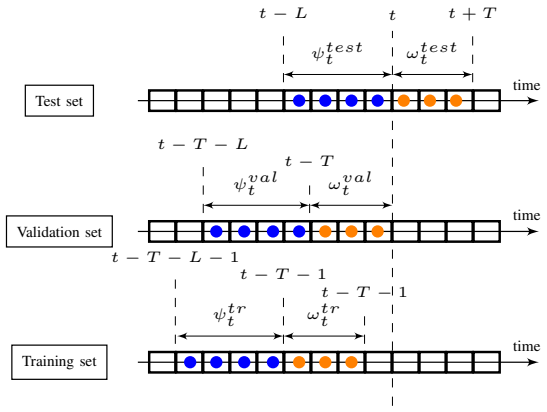

Fig. 2. Visualisation of the data splitting in training, validation and test in the case $L = 4$, $T = 3$, $B_{tr} = 1$ and $B_{val} = 1$ at time step t. Each black square represents a feature sample f_t. Samples highlighted in blue are the inputs of the NN, samples highlighted in orange are the NN targets.

test sets are shifted forward by one step. The data splitting is visualised in Fig. 2.

B. Online Weight Averaging

The goal of the online learning algorithm is to react to sudden changes in the environment (e.g. in the electricity price), while retaining the knowledge acquired in the previous time steps. Hence, to make the learning process more robust to sudden changes in the environment and in the input signal, we implement an online variant of the SWA [13] applied to OL-based controllers. In [13] a method to improve the generalization capabilities of NNs for offline learning, by considering the final model as the average over the last N training iterations, is proposed. Here we adapt the algorithm to an OL scenario, where an average model is updated at each step. In particular, we denote θ_t the NN weights at times t and θ_t^{∇} the weights after the gradient update at time step t, γ is the moving average decay factor, and $\theta_t^{avg}, \theta_{t+1}^{avg}$ are the moving averages of the weights. We update the weights for the next time step θ_{t+1} every δ time steps. During initialization, we set $\theta_0^{avg} = \theta_0$ and then at each step we update the average model and the current model through a moving average as $\theta_{t+1}^{avg} = \gamma\theta_t^{avg} + (1-\gamma)\theta_t^{\nabla}$ The OL deployment scenario requires updating θ^{avg} at every time step, since we do not know the last N training iterations in advance as in [13]. To adjust the learning algorithm's sensitivity, one can tune the hyperparameters γ and δ. γ and δ control how much the current model relies on the average model θ_t^{avg}, which is a smoothed version of the current model θ_t. The γ regulates the content of the preserved weights e.g. $\gamma = 0.9$ means that 0.9 magnitude of the average is preserved 0.1 is the weighting of the current model at time step t. The δ regulates how often the current model is replaced with the average model e.g. $\delta = 10$ means that every 10 time steps t we reset the current model to be equal to the average model. Setting the current model θ_{t+1} to the average model θ_{t+1}^{avg} aims at promoting the generalization capabilities of the NN, by resetting its optimization pathway, which prevents the network from getting stuck in local minima or being oversensitive to abrupt changes in the input signals.

C. Hyperparameter optimisation

In addition to finding the optimal NN architecture, it is necessary to tune/set the hyperparameters of the learning algorithm to the specific problem at hand. Here we utilise the Hyper Parameter Optimisation (HPO) using the syne-tune [20] library. More specifically, we formulate the HPO in terms of the domain dataset D_1 and wall-clock time budget C to find the best hyperparameters and architecture under C by minimising a chosen loss function using the validation set. Given the winning model $\mathcal{M}^*$, this model and its associated hyperparameters can then be deployed on the continuation of D_1 for online training, as done in the experiments in Section IV. Furthermore, $\mathcal{M}^*$ and its configuration can be used to initialise the model to be learnt on a different dataset D_2, as shown in the experiments for TL validation.

IV. EXPERIMENTAL RESULTS

In this Section, we describe the experiments designed to show the effectiveness of the proposed method. All the experiments have been repeated 3 times with different random seeds and the Tables represent the mean results with a single standard deviation. The microgrids are composed of the agents described in Section II-A: load, renewable generation, storage system and connection to the utility grid. We used Python 3.6.8, PyTorch 1.10.1 [17], PyTorch Lightning 1.5.10 [9], syne-tune 0.2 [20] and cvxpylayers 0.1.5 [1]. The comparison between different simulations is based on two performance indices: (i) the total cost, computed through (4) over all the simulation steps; (ii) the MSE on the test set, computed as $\sum_{f_t \in \{p_t, P_t^r, P_t^l\}} \sum_{t=0}^{S} \sum_{k=0}^{T-1} \left(f_{t+k} - \hat{f}_{k|t}\right)^2$, where S is the number of simulation steps, f_{t+k} is the feature sample of the test set at time $t+k$ and $\hat{f}_{k|t}$ is the prediction of f.

Datasets description. We paired two data sources to create datasets for our experiments. The first provides is the *EMSx* benchmark dataset [15] that collects the power profiles for renewable production and load demand, as well as storage characteristics of two different microgrids, we considered the recordings of industrial sites 10 and 12 between 01/01/2016 and 07/10/2017. The second data source is the *ENTSO-E Transparency Platform* dataset [8, 11] that collects the day-ahead electricity price, for each European bidding zone, we employed the price profile of *IT-Centre-North* of the same time range as the power profiles. The storage parameters are provided in the first dataset: $(s_M[\text{kW h}], \eta_c, \eta_d, P_M^s[\text{kW}])$ are $(400, 0.95, 1.05, 100)$ and $(800, 0.95, 1.05, 200)$ for site 10 and 12. All data samples have been normalised in the range 0-1, we recognize that, in an OL scenario, the normalisation factors are not available beforehand, nevertheless the normalisation factors can be inferred from past recordings. The other numerical values used for simulations are as: $T_s = 1\text{h}$, $T = 24$, $L = 168$, $s_m = 0.1 s_M$, $\sigma = 0.0042$, $s_0 = 200\text{kW h}$, $B^{tr} = B^{val} = 1$, $n_{\text{layers}} = 1$. We

TABLE I

COMPARISON OF PERFORMANCE OF DIFFERENT CONTROLLERS

Site	Controller	Cost [€]	MSE
10	OL-MPC	285,681 ± 36	1.007 ± 0.028
	1yOL-MPC	285,962 ± 150	1.563 ± 0.028
	noSWA	285,829 ± 368	0.990 ± 0.006
12	OL-MPC	59,770 ± 45	1.203 ± 0.019
	1yOL-MPC	60,741 ± 345	3.530 ± 0.094
	noSWA	59,745 ± 182	1.082 ± 0.013

TABLE II

TRANSFER LEARNING RESULTS

D_1	D_2		Cost [€]	MSE
12	10	NO TL	286,124 ± 13	1.185 ± 0.044
		Only TL	286,617	1.597
		OL+TL	285,295	0.345
10	12	NO TL	59,387 ± 342	1.066 ± 0.021
		Only TL	59,867	9.113
		OL+TL	57,995	3.460

selected the hyperparameters of the NN for each industrial site using the procedure explained in Section III-C using random search [4] with 4 CPU workers for a maximum time of 3 hours. The optimised hyperparameters are the learning rate, the weight decay, δ, γ and H which are $(3.38\times10^{-4}, 1.88\times10^{-4}, 400, 7.07\times10^{-1}, 48)$ and $(3.31\times10^{-3}, 8.88\times10^{-5}, 50, 8.00\times10^{-1}, 48)$ for site 10 and 12.

Online learning experiments. The goal of these experiments is to show how the OL algorithm is able to adapt to changing environment conditions. To this end, we compare our controller, denoted as OL-MPC, with another controller denoted as 1yOL-MPC. For this controller the weights θ_t are updated until time step $\bar{t}$. After this the training is stopped and the network weights are not updated anymore $\theta_{\bar{t}+k} = \theta_{\bar{t}}$. For this simulation we set the learning stop time after the first year of simulation and we continue the simulation for the next 10 months. Simulation results are shown in Table I. Notably, the proposed controller OL-MPC outperforms the 1yOL-MPC due to its ability to adapt to substantial changes in the profiles. For example, consider the price profile in Fig. 3 between 20 December 2016 and 14 January 2017. The blue and green profiles are respectively the first predicted sample $\hat{p}_{1|t}$ by the OL-MPC and the 1yOL-MPC. The vertical red line indicates the moment in which the training is stopped. While our OL-MPC algorithm is able to adapt to this change, the values predicted by the 1yOL-MPC remain in the same range.

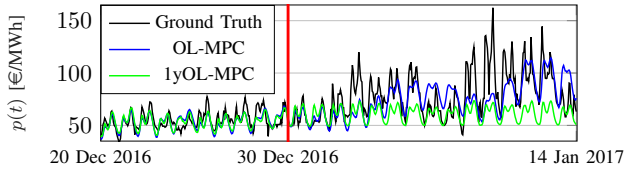

Fig. 3. Ground truth and predicted price profiles obtained using the OL-MPC and the 1yOL-MPC.

SWA experiments. The experiments reported in this Section are related to the online implementation of the SWA algorithm. In particular, we show that enabling the SWA throughout the simulation decreases the variability of the obtained performance. Table I reports the cost and the MSE obtained simulating the EMS in site 10 and 12, the controller is denoted as noSWA. In order to compare fairly the results of these experiments with the ones presented in the previous Section, we re-run the HPO optimising only the learning rate, the weight decay and H and disabling the SWA algorithm. The mean value of the results, for both MSE and cost, are similar to the OL-MPC controller, however, the variability increases by disabling the SWA.

Transfer learning experiments. We now show that the proposed algorithm is suitable for TL, by showing that the control performance improves if the NN is pre-trained on an available dataset before the controller deployment. We show here that the combination of TL and OL allows to achieve the lowest cost. We assume to have two different industrial site datasets, we use the same price profile for both the datasets, D_1 which is available before the controller deployment for EMS design and D_2 which is not available before the controller deployment and represents the new microgrid for which the controller has been commissioned. We run three different experiments: (i) we use the dataset D_1 only for hyperparameters optimisation and train the NN on D_2 using our algorithm. This experiment is denoted in Table II as NO TL; (ii) we optimise hyperparameters and train the NN on D_1 and we then deploy the controller on D_2 disabling the OL. This experiment is denoted in Table II as only TL; (iii) we test the combination of OL and TL. The dataset D_1 is used for both hyperparameters optimisation and pre-training. The controller is then deployed to dataset D_2 where the training continues. This experiment is denoted in Table II as TL+OL. For all the experiments data samples are normalised using normalisation factors of D_1. Simulation results are reported in Table II. It is clear that pre-training the NN on an available dataset has a beneficial effect on the control performance since results of Table II are similar to the results of the OL-MPC in Table I even though the hyperparameters have not been optimised for dataset D_2 specifically. However, it is clear that the combination of OL and TL allows to achieve the lowest cost. It should be noted that the total cost of OL+TL in Table II is even lower than the one obtained using OL-MPC in Table I even though the NN hyperparameters in the OL+TL case are not optimised for D_2. We indirectly attribute this result to the online SWA procedure which led the network's weights towards generalization beyond a single dataset.

V. CONCLUSION

The paper proposes a novel learn-based MPC scheme for microgrid energy management that uses an online trained NN with SWA. The NN predicts load, renewable generation, and electricity prices, and optimizes the control actions accordingly. The proposed method is tested on real microgrid datasets and shows promising results in terms of

cost reduction and transfer learning capabilities. As a future work, we plan to investigate the role of the loss function both for the HPO and for online training, by comparing prediction optimisation and end-to-end optimisation, thus directly minimising the control performance instead of the prediction error, or combining different loss functions.

REFERENCES

[1] Akshay Agrawal et al. "Differentiable convex optimization layers". In: *Advances in neural information processing systems* 32 (2019).

[2] Verónica Álvarez, Santiago Mazuelas, and José A Lozano. "Probabilistic load forecasting based on adaptive online learning". In: *IEEE Transactions on Power Systems* 36.4 (2021), pp. 3668–3680.

[3] Alberto Bemporad, Laura Puglia, and Tommaso Gabbriellini. "A stochastic model predictive control approach to dynamic option hedging with transaction costs". In: *Proceedings of the 2011 American control conference*. IEEE. 2011, pp. 3862–3867.

[4] James Bergstra and Yoshua Bengio. "Random search for hyper-parameter optimization." In: *Journal of machine learning research* 13.2 (2012).

[5] Vittorio Casagrande and Francesca Boem. "Learning-based MPC using Differentiable Optimisation Layers for Microgrid Energy Management". In: *2023 European Control Conference (ECC) (Accepted)*. IEEE. 2023.

[6] Vittorio Casagrande et al. "Resilient Microgrid Energy Management Algorithm Based on Distributed Optimization". In: *IEEE Systems Journal* ().

[7] Michelangelo Ceci et al. "Spatial autocorrelation and entropy for renewable energy forecasting". In: *Data Mining and Knowledge Discovery* 33.3 (2019), pp. 698–729.

[8] ENTSO-E. *Transparency platform*. URL: https://transparency.entsoe.eu. (Accessed: 14.03.2022).

[9] William Falcon and The PyTorch Lightning team. *PyTorch Lightning*. Version 1.4. Mar. 2019. DOI: 10.5281/zenodo.3828935. URL: https://github.com/Lightning-AI/lightning.

[10] Mohammad Navid Fekri et al. "Deep learning for load forecasting with smart meter data: Online Adaptive Recurrent Neural Network". In: *Applied Energy* 282 (2021), p. 116177.

[11] Lion Hirth, Jonathan Mühlenpfordt, and Marisa Bulkeley. "The ENTSO-E Transparency Platform–A review of Europe's most ambitious electricity data platform". In: *Applied energy* 225 (2018), pp. 1054–1067.

[12] Sepp Hochreiter and Jürgen Schmidhuber. "Long short-term memory". In: *Neural computation* 9.8 (1997), pp. 1735–1780.

[13] Pavel Izmailov et al. "Averaging weights leads to wider optima and better generalization". In: *arXiv preprint arXiv:1803.05407* (2018).

[14] Ying Ji et al. "Real-time energy management of a microgrid using deep reinforcement learning". In: *Energies* 12.12 (2019), p. 2291.

[15] Adrien Le Franc et al. "EMSx: a numerical benchmark for energy management systems". In: *Energy Systems* (2021), pp. 1–27.

[16] Alessandra Parisio, Evangelos Rikos, and Luigi Glielmo. "A model predictive control approach to microgrid operation optimization". In: *IEEE Transactions on Control Systems Technology* 22.5 (2014), pp. 1813–1827.

[17] Adam Paszke et al. "Pytorch: An imperative style, high-performance deep learning library". In: *Advances in neural information processing systems* 32 (2019).

[18] Ionela Prodan, Enrico Zio, and Florin Stoican. "Fault tolerant predictive control design for reliable microgrid energy management under uncertainties". In: *Energy* 91 (2015), pp. 20–34.

[19] Naren Srivaths Raman et al. "Reinforcement learning-based home energy management system for resiliency". In: *2021 American Control Conference (ACC)*. IEEE. 2021, pp. 1358–1364.

[20] David Salinas et al. "Syne Tune: A Library for Large Scale Hyperparameter Tuning and Reproducible Research". In: *First Conference on Automated Machine Learning (Main Track)*. 2022. URL: https://openreview.net/forum?id=BVeGJ-THIg9.

[21] Bharatkumar V Solanki et al. "Including smart loads for optimal demand response in integrated energy management systems for isolated microgrids". In: *IEEE Transactions on Smart Grid* 8.4 (2015), pp. 1739–1748.

[22] Ganesh Kumar Venayagamoorthy et al. "Dynamic energy management system for a smart microgrid". In: *IEEE transactions on neural networks and learning systems* 27.8 (2016), pp. 1643–1656.

[23] Leandro Von Krannichfeldt, Yi Wang, and Gabriela Hug. "Online ensemble learning for load forecasting". In: *IEEE Transactions on Power Systems* 36.1 (2020), pp. 545–548.

[24] Kim P Wabersich and Melanie N Zeilinger. "Safe exploration of nonlinear dynamical systems: A predictive safety filter for reinforcement learning". In: *arXiv preprint arXiv:1812.05506* (2018).

[25] Tiancai Wang, Xing He, and Ting Deng. "Neural networks for power management optimal strategy in hybrid microgrid". In: *Neural Computing and Applications* 31.7 (2019), pp. 2635–2647.

[26] Fuzhen Zhuang et al. "A comprehensive survey on transfer learning". In: *Proceedings of the IEEE* 109.1 (2020), pp. 43–76.

[27] Muhammad Fahad Zia, Elhoussin Elbouchikhi, and Mohamed Benbouzid. "Microgrids energy management systems: A critical review on methods, solutions, and prospects". In: *Applied energy* 222 (2018), pp. 1033–1055.