Machine Learning for Ranking Day-ahead Decisions in the Context of Short-term Operation Planning

Laurine Duchesne, Efthymios Karangelos, Antonio Sutera, Louis Wehenkel
Montefiore Institute - Department of EE&CS
University of Liège, Liège, Belgium
{l.duchesne, e.karangelos, a.sutera, l.wehenkel}@uliege.be

***Abstract*—In operation planning, probabilistic reliability assessment consists in evaluating, for various candidate planning decisions, the induced probability of meeting a reliability target and the expected operating cost over a certain future time period. In this paper, we propose to exploit Monte-Carlo simulation and machine learning to predict operation costs for various day-ahead unit commitment and economic dispatch decisions and a range of realisations of uncertain loads and renewable generations over the next day. We describe how to generate a database, how to apply supervised machine learning to it, and how to use the learnt proxies to rank candidate day-ahead decisions in terms of the expected operating cost they induce over the next day. We illustrate the approach on the IEEE-RTS96 benchmark where we use the DC power-flow approximation and the N-1 criterion to simulate real-time operation and to generate generation schedules in the day-ahead operation planning stage.**

***Index Terms*—Machine learning, Monte Carlo simulation, Operation planning, Probabilistic reliability assessment**

I. Introduction

In operation planning, reliability management aims at taking decisions ahead in time so that the system can be later on operated according to its reliability targets. For example, day-ahead decisions may be taken to adjust market clearing outcomes, to postpone planned outages, *etc*, so as to enable the operator to meet his reliability targets over the next day while minimising the cost of operating the system. In general, this reliability management problem is decomposed into *reliability assessment* and *reliability control* [1].

In this paper we consider as 'template' the day-ahead operation planning context, and we address the problem of ranking various candidate day-ahead planning decisions in terms of the *expected* next-day operating cost they induce, while considering exogenous uncertainties such as load or renewable generation. The final purpose of this would be to help selecting a day-ahead decision among a given set of candidate such decisions, while considering the impact of uncertainties on operation. More specifically, we propose a method to evaluate, for a list of candidate day-ahead decisions, their corresponding *expected* cost of real-time operation and then use this evaluation to rank the decisions to, *in fine*, ease the identification of a 'good' day-ahead decision.

To state our problem, we model in a probabilistic way the exogenous uncertainties about load and renewable generation that would be faced by the operation planner. We also model the behaviour of the real-time operator via a *real-time decision making simulator* using the next-day information state to compute next-day decisions and the different terms of next-day operation cost.

To evaluate the *expected* cost of operation, a possible solution would exploit a crude Monte-Carlo (CMC) approach using the real-time decision making simulator, for many next-day scenarios drawn by using the probabilistic model of exogenous uncertainties. However, this approach is most of the time impractical. Indeed, simulating real-time operation consists often in running Security Constrained Optimal Power Flow (SCOPF) types of problems with a very large number of variables [2] especially for large-scale power systems. In this regard, given that in day-ahead the operator has only a few hours to select a day-ahead decision, running a very large number of SCOPFs for each candidate day-ahead decision may not be realistic from a practical point of view.

To reduce the computational burden of the CMC approach, we proposed in [3] to replace the real-time decision making simulator by a much faster *proxy* built with machine learning algorithms, and we also leveraged variance reduction techniques, such as control variates, to yield unbiased estimates of the expected operating cost. In that work, the problem was tackled in a set-up of assessing only one single already selected day-ahead decision.

In the present paper, we propose a generalisation of the approach presented in [3], to further help choosing among day-ahead decisions. We thus address the problem of building a *proxy* predicting operation costs with acceptable performances for several candidate day-ahead decisions, even unseen ones. This proxy could replace the real-time decision-making simulator in the CMC approach, allowing one to much more rapidly assess the expected cost of next-day operation for various candidate day-ahead decisions. We propose a methodology to automatically build a database combining different candidate day-ahead decisions with a sample of scenarios representing the expected range of possible next-day conditions, and show how to exploit such a database to learn and validate a proxy of real-time operation. In particular we investigate the use of neural networks and multitask learning [4] to assess different day-ahead decisions and to rank them in terms of their induced next-day operating cost. We test this approach on the three-area IEEE-RTS96 system, while using the DC power flow model and the N-1 security criterion in order to simulate real-time operation as the resolution of a sequence of DC-

SCOPF problems. In our case study, we consider as candidate day-ahead decisions several unit commitments and market dispatches, as well as provisional wind curtailment, and we focus on the estimation of the expected value of the real-time preventive control costs.

The idea of exploiting machine learning in power system security and reliability assessment dates back to the 1970s [5]. More recently, due to the advances in the field of machine learning, an increasing number of papers exploiting this approach have been published (see, for example [6]–[9]). We refer the interested reader to [10] for a comprehensive survey of recent works applying machine learning in the context of reliability management. Closer to the subject of the present work, several papers propose to use machine learning to learn the outputs of a SCOPF. For instance, [11]–[13] exploit deep learning to predict the generators set-points. Finally, several other papers propose machine learning approaches to build proxies of shorter-term decision-making contexts to be used when solving longer-term reliability assessment problems, but in other contexts than short-term operation. For example, in [14], [15], the authors propose, in the context of outage scheduling, to build proxies of short-term operation based on the nearest neighbor algorithm.

Our approach also shares strong similarities with analytical two-stage optimisation such as stochastic unit commitment [16]. However, we see several differences. First, our approach can be used to rank under uncertainties various sets of candidate first-stage decisions, for which we do not need to know how they have been chosen. In contrast, solving a two-stage stochastic program returns a (locally or globally) optimal candidate decision but no quantitative information on the merit of this decision with respect to the other available alternatives. Furthermore, our methodology relies essentially only on massively parallel simulations of the operator's real-time behaviour and induced costs along different scenarios. On the other hand, stochastic unit commitment approaches require that the real-time operation strategy of the operator is explicitly modelled in the form of a second-stage optimisation problem, and typically impose strong restrictions on such second-stage models in order to yield computationally tractable two-stage stochastic formulations.

The rest of this paper is organised as follows. Section II states the problem studied and presents methodologies to generate automatically a database of real-time operation trajectories, to build proxies with supervised learning and to rank candidate day-ahead decisions with the help of these proxies. Section III reports the case-study on the IEEE-RTS96 system, where we analyse the generalisation capability of the proxies and we exploit them to rank candidate day-ahead decisions. Section IV concludes and suggests future works.

II. Problem statement and proposals

A. Problem statement

In this work, we consider the standard setting wherein in day-ahead (da) the operation planner seeks to ensure N-1 secure operation over every time period of the forthcoming day, and subject to uncertainty on renewable power generation and demand. In particular, the mission of the operation planner is to select in advance (i) the commitment and dispatch of generating units, and, (ii) provisional curtailment of wind power generation. To help in his decision-making, we assume that he has access to a simulator of real-time operation along the next day (nd), modelling both the physical behaviour of the power system and the decision-making of control room operators in response to the resolution of uncertainties in real-time. We use a sequence of 24 SCOPF computations following the N-1 criterion while minimising the costs of real-time operation to this end.

We also assume that the operation planner has at his disposal a generative model of day-ahead uncertainties, allowing him to sample scenarios of next-day load and renewable generations, denoted $\{\xi^1_{nd}, \xi^2_{nd}, ...\}$. For a given candidate decision δ^i_{da} and a given scenario ξ^j_{nd}, the application of the real-time operation simulator allows the planner to anticipate real-time operation along the next day and in particular to evaluate the costs $y^{i,j}$ of real-time operation. Furthermore, we suppose that the operation planner is interested in evaluating, for any given day-ahead decision, the consequence over next-day operation by the expected value of the cost of operating the system the next day.

The problem addressed in this paper then amounts to *screening candidate day-ahead decisions to select a good day-ahead decision in terms of its expected impact on next-day operating costs, while exploiting the available generative model and real-time operation simulator*.

B. Generating a database of day-ahead decisions and next-day scenarios

We extend the methodology described in [3] to generate both a set of day-ahead decisions and a set of next-day scenarios. In this approach, m next-day scenarios $\{\xi^1_{nd}, \xi^2_{nd}, ..., \xi^m_{nd}\}$ are sampled with the generative model of day-ahead uncertainties available to the operation planner. They are then combined with k day-ahead decisions $\{\delta^1_{da}, \delta^2_{da}, ..., \delta^k_{da}\}$ generated as described in the following subsection.

1) Generating k day-ahead decisions: Here we assume that the planner has a day-ahead decision-making support software, e.g. in the form of a deterministic multi-period Unit Commitment and Economic Dispatch (UCED) program.

To generate the k day-ahead decisions with such a tool, we first generate a large sample ($n \gg k$) of next-day scenarios by using our generative model of uncertainties, and then apply to them the *k-means* clustering algorithm [17].[1] We then compute k day-ahead decisions $\{\delta^1_{da}, \delta^2_{da}, ..., \delta^k_{da}\}$ from these k next-day scenarios $\{\xi^1_{nd}, \xi^2_{nd}, ..., \xi^k_{nd}\}$ by applying to each scenario the available day-ahead decision-making support software.

2) Simulating real-time operation: For a given pair $(\delta^i_{da}, \xi^j_{nd})$ combining a day-ahead decision and a next-day scenario (we call this combination the *trajectory* $\tau^{i,j}$), we

[1]If n is sufficiently large, the resulting k real-time scenarios will cover as well as possible the uncertainty space, for a given budget k.

apply the real-time operation simulator to compute the corresponding next-day operation costs, denoted by $y^{i,j}$. This gives us a database $\{(\tau^{i,j}, y^{i,j})\}^N$ of size $N = k \times m$, with features describing day-ahead decisions and next-day scenarios as inputs and real-time operation costs as outputs, that we can exploit to learn proxies of the real-time operation simulator.

C. *Learning and generalising the proxies*

A proxy is a simplified model of real-time operation, allowing to predict the real-time operation costs $y_{i,j}$ for a given day-ahead decision δ^i_{da} and a given next-day scenario ξ^j_{nd}. We propose to build proxies with supervised learning, which is a branch of machine learning that consists in finding a function $h(\cdot)$ that maps some vector of inputs x to (a vector) of outputs y, given a database $\{(x^i, y^i)\}_{i=1}^N$ of input-output pairs [17].

1) Splitting the database in training and test sets: We are interested in evaluating the accuracy of our proxies to unseen day-ahead decisions, to unseen next-day scenarios, and to combinations of both. For that, we need to split the database into a learning set $\mathcal{L}$ and a validation set $\mathcal{V}$ used both to train the proxies, and three test sets $\mathcal{T}$ used to evaluate their generation capabilities. Figure 1 shows this database decomposition in a graphical way.

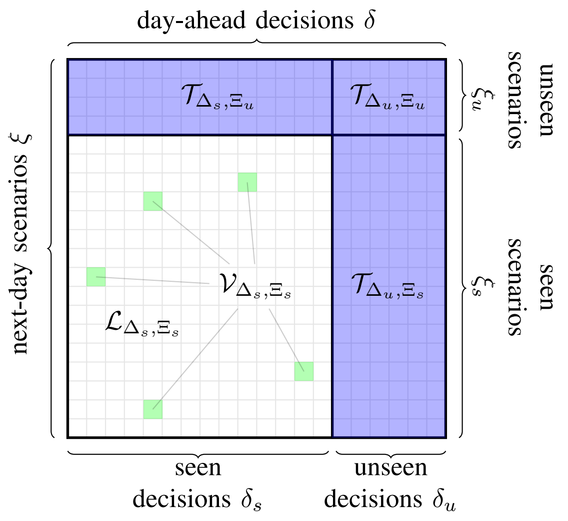

day-ahead decisions δ
$\mathcal{T}_{\Delta_s,\Xi_u}$
$\mathcal{T}_{\Delta_u,\Xi_u}$
ξ_u
unseen scenarios
next-day scenarios ξ
$\mathcal{V}_{\Delta_s,\Xi_s}$
$\mathcal{L}_{\Delta_s,\Xi_s}$
$\mathcal{T}_{\Delta_u,\Xi_s}$
ξ_s
seen scenarios
seen decisions δ_s
unseen decisions δ_u

Fig. 1. Schematic representation of the database partition between train ($\mathcal{L}$, in white), validation ($\mathcal{V}$, in green) and test ($\mathcal{T}$, in blue) sets. Rows and columns respectively represent scenarios and decisions.

2) Assessing the accuracy of proxies: In order to assess the accuracy of a proxy $h_p(\cdot)$ based on a sample $\mathcal{S}$ of size $|\mathcal{S}|$, we consider the square-loss $\ell(\hat{y}, y) = (\hat{y} - y)^2$ and compute the *empirical loss* by

$$L(h_p, \mathcal{S}) = \frac{1}{|\mathcal{S}|} \sum_{(x^i, y^i) \in \mathcal{S}} \left(h_p(x^i) - y^i\right)^2 .$$

From there, we say that $h_p(\cdot)$ generalises well to unseen day-ahead decisions if $L(h_p, \mathcal{T}_{\Delta_u,\Xi_u}) \approx L(h_p, \mathcal{T}_{\Delta_s,\Xi_u})$.

D. *Using the proxies for ranking day-ahead decisions*

Once one has shown that the proxies generalise well to unseen decisions, one could use them in order to identify a good day-ahead decision. To do so, we propose to first select (randomly) a subset Δ_s of candidate day-ahead decisions that will be used to build a proxy of the real-time operation simulator. This subset should be large enough for the proxy to be able to generalise well to unseen decisions but small enough to avoid as much as possible the computational burden stemming from the use of the heavy real-time simulator. For all $\delta^i_{da} \in \Delta_s$ and $\xi^j_{nd} \in \{\xi^1_{nd}, \xi^2_{nd}, ..., \xi^m_{nd}\}$, we compute the next-day operation costs $y^{i,j}$ with the real-time simulator and then we build the proxy as described in section II-C. Then we apply the proxy to predict the next-day operation cost $y^{i,j}$ for all $\delta^i_{da} \in \Delta_u$ and $\xi^j_{nd} \in \{\xi^1_{nd}, \xi^2_{nd}, ..., \xi^l_{nd}\}$[2]. After that we average the predictions over the l scenarios to have an estimate $\bar{y}^i_p$ of the expected next-day operation cost for each decision in $\delta^i_{da} \in \Delta_u$ and we average the $y^{i,j}$ over the m scenarios to have an estimate $\bar{y}^i$ for each decision in $\delta^i_{da} \in \Delta_s$. Finally, these estimates are used to rank the candidate decisions according to their expected next-day operation cost.

III. Case-study on the IEEE-RTS96 benchmark

In order to test our proposed methodology, we place ourselves in the context of day-ahead operation planning, when the operation planner has to select a unit commitment and economic dispatch for the next day.

Our real-time operation simulator as well as the scenario generator are implemented in JULIA. We use Python for learning: *scikit-learn* for the clustering algorithms and *Pytorch* as the deep learning framework.

A. *Test system, uncertainties, day-ahead decision-making and real-time operation simulator*

1) Test system: We consider as test system the 3-area IEEE-RTS96 benchmark [18], where 19 wind farms have been added, as was proposed in [19]. We consider as initial scenario the demand and wind generation of the first day of the year, for a peak demand of 3135 MW per area, and favourable wind.

2) Day-ahead decision-making: A day-ahead decision corresponds to the commitment status and economic dispatch of all dispatchable generators as well as the provisional curtailment of wind power generation. To simulate day-ahead decision-making, we apply to the forecasted scenarios a multi-period SCOPF, under the DC-approximation. We consider that the market dispatch should satisfy the N-1 security criterion on transmission system elements, using provisional wind curtailment as a measure of last resort. We also impose a minimum upward and downward spinning reserve capacity constraint of 300 MW in each area of the system. The full formulation can be found in the appendix of [3].

3) Uncertainties and next-day scenarios: We place ourselves at noon, the day ahead. We consider uncertainties coming from errors in demand and wind generation forecast, modelled by Gaussian distributions. In particular, for each element (load or wind farm) we consider two terms of error, a local one and a global one (see [3]). We exploit this model

[2]Note that that these l scenarios can be different from the m scenarios generated for learning the proxy.

coupled with Monte-Carlo simulation to generate plausible next-day scenarios of realized demand and wind generation. Since we chose a one-hour time step, each next-day scenario corresponds to 24 snapshots of one day. The precise methodology is described in our previous work [20].

4) Real-time operation simulator: The real-time operation simulator is modelled as a sequence of 24 SCOPF computations (using as well the N-1 criterion and the DC-approximation), to adapt to the realization of uncertainties via preventive and corrective generator redispatch, load shedding and wind curtailment. These recourse decisions are chosen so that it is always possible to come back to the market dispatch decided for the next hour, given the ramping constraints of generating units, and with an emphasis on the preventive control cost so as to favour corrective actions whenever they can help. We refer the reader to the appendix of [3] for the mathematical formulation of this SCOPF.

B. Description of the database

For our study, we generated $k = 20$ candidate day-ahead decisions, applying the methodology of Section II-B with $n = 20,000$ next-day scenarios. We combined further $m = 600$ next-day scenarios with each one of these $k = 20$ day-ahead decisions to yield $12,000$ trajectories of 24 hourly time-steps.

In our database, the input-features per hourly snapshot of each trajectory are the following:

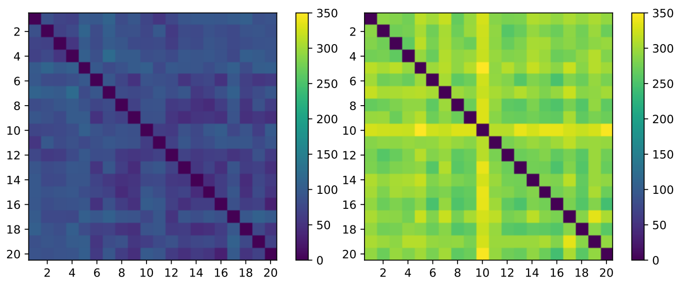
- demand realisations for each load,
- wind generation realisations for each wind farm,
- difference between the real-time scenario and the forecast scenario used to generate the corresponding day-ahead decision (in MW and in %),
- total demand, total wind generation and net load,
- maximum and minimum generation capacity,
- hour of the day.

Concerning the outputs of the database, we focus here on the total cost of preventive actions, which is the sum of the preventive generation redispatch cost, the preventive load shedding cost and the preventive wind curtailment cost along a next-day scenario.

C. Analysis of the 20 candidate day-ahead decisions

We begin our analysis with the comparison of the unit commitments (on-off statuses) and economic dispatches (active power levels) of the 20 candidate day-ahead decisions. For that, we represent each decision as a vector containing as elements either the unit commitment or day-ahead dispatch of each one of the 96 generating units for each one of the 24 hours of the day. To compare the vectors pairwise, we use the *Hamming distance*, expressed as the number of components for which the two vectors differ. The results are presented in the form of heat maps and can be seen in Figure 2.

We observe that the decisions are different, and that the distances between them are of same order of magnitude, both in the unit-commitment space and the space of economic dispatches. On average, the Hamming distance for unit



(a) Day-ahead unit commitment (b) Day-ahead economic dispatch

Fig. 2. Hamming distance between each candidate day-ahead decision for respectively the unit commitment and day-ahead dispatch.

TABLE I
MEAN, STANDARD ERROR, MINIMUM AND MAXIMUM VALUE OF THE TOTAL PREVENTIVE COST FOR THE $k = 20$ STUDIED DAY-AHEAD DECISIONS COMPUTED OVER $m = 600$ NEXT-DAY SCENARIOS. SORTED IN INCREASING ORDER OF MEAN.

δ^i_{da}	Mean ($\times 10^6$)	Standard err ($\times 10^4$)	Min ($\times 10^5$)	Max ($\times 10^6$)
2	1.641	3.439	5.278	6.339
19	1.656	3.163	4.912	7.024
1	1.661	3.350	5.072	6.672
3	1.676	3.263	5.382	7.404
11	1.689	3.384	5.136	7.466
8	1.691	3.411	5.970	7.091
6	1.692	3.150	5.546	8.139
7	1.719	3.741	5.151	7.680
10	1.728	3.615	5.153	7.664
9	1.731	3.705	4.788	7.182
4	1.732	3.264	6.359	6.215
20	1.740	3.560	5.835	7.119
16	1.746	3.412	5.270	6.781
14	1.748	3.518	5.944	7.608
18	1.788	3.700	5.061	7.287
12	1.797	4.046	5.248	8.192
5	1.798	3.485	6.345	6.317
13	1.855	3.631	5.928	6.293
15	1.872	3.793	5.412	7.547
17	1.917	3.962	6.546	7.912

commitments is equal to 3% of the vector components. This proportion rises to 12% for the economic dispatches.

Another analysis we can make to compare candidate day-ahead decisions is to look at their impact on next-day operation costs. Since we used the same 600 next-day scenarios with each candidate day-ahead decision, the results are directly comparable. We look at some statistics of the total preventive cost over the 600 scenarios, such as the average value $\bar{y}^i$, the standard error which is defined as $\frac{\sigma}{\sqrt{m}}$, with $m = 600$ and σ the standard deviation of $y^{i,j}$ over the 600 scenarios, and the minimum and maximum values of $y^{i,j}$. The results can be seen in Table I, which is sorted in increasing order of the mean total preventive cost.

When analysing the mean total preventive cost, we see that there is clear difference between decisions. For instance decision 17, the most costly decision, is in average 300,000€ more expensive in real-time than decision 2. If one analyses the components of the total preventive cost, it can be seen that this decision leads to the largest load shedding cost in average over next day. Note that given the standard error values, we cannot guarantee that decision 2 is effectively the candidate decision with the smallest expected next-day preventive control cost, but the less expensive decision on average should nevertheless be among the first few decisions of Table I.

D. Machine learning protocol

To predict the total preventive cost along a next-day trajectory we divide this trajectory in 24 hourly snapshots, predict the total preventive cost for each hour (denoted as hourly prediction in [3]) and then sum the 24 predictions. Note that we provide information about the hour of the day in two ways, one using one single input with values ranging from 1 to 24 and another one using 24 binary inputs with a one-hot-encoding. This leaves us with 240 hourly input features and one single output variable.

As supervised learning algorithm, we chose the neural network algorithm [17]. A neural network has many meta-parameters that need to be selected. We used a gridsearch to find a suitable configuration of meta-parameters among the following candidate values: 3, 4, 5 or 6 layers and 50, 100 and 200 neurons per layer. Furthermore, we tested 5 different initializations of the networks' weights. The other meta-parameters were kept constant. In particular, we used a batch size of 200, a learning rate of 10^{-3}, the Adam optimizer [21] and a weight decay of 10^{-4}. The maximum number of epochs is 200, and we keep the model corresponding to the epoch minimising the loss on the validation set.

In reference to Figure 1, 100 next-day scenarios are always kept out in order to yield the test sets. As concerns the splitting along the day-ahead decisions, we investigate different settings in the sequel. In any case, the validation set $\mathcal{V}$ corresponds to 5% of the trajectories not used in the test sets; it is used to select the meta-parameters of the learning algorithms[3], while the remaining 95% provide the learning set $\mathcal{L}$ used only to tune the parameters of the neural network predictors.

To evaluate the proxies, we always use the R^2-score, which is computed on the basis of a sample $\mathcal{S}$ by [22]:

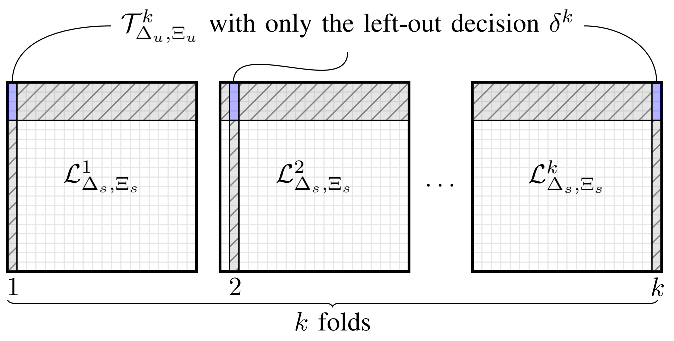
$$R^2(h_p, \mathcal{S}) = 1 - \frac{\sum_{(x^i,y^i)\in\mathcal{S}}(y^i - h_p(x^i))^2}{\sum_{(x^i,y^i)\in\mathcal{S}}(y^i - \bar{y}_\mathcal{S})^2},$$

where $\bar{y}_\mathcal{S}$ is the mean of the targets y^i over $\mathcal{S}$. The best possible R^2-score is 1 and corresponds to a proxy h_p that perfectly predicts the target values y^i over $\mathcal{S}$. The best constant model would systematically predict the sample-mean $\bar{y}_\mathcal{S}$ and obtain an R^2-score of 0.

E. Generalisation over day-ahead decisions

1) Leave-one-decision-out: For this experiment, we use samples from $k-1$ decisions to learn a proxy and we test it with data from the k^{th} (unseen) decision applied on unseen scenarios. We redo this experiment k times, with each time a different unseen decision in the test set. We then average the k test scores obtained. If this score is high, the proxy is able to well generalise to unseen decisions. Figure 3 illustrates the principle of the leave-one-decision-out experiment.

[3]We keep the combination of meta-parameters minimising the loss on the validation set.



$\mathcal{T}^k_{\Delta_u,\Xi_u}$ with only the left-out decision δ^k
$\mathcal{L}^1_{\Delta_s,\Xi_s}$
$\mathcal{L}^2_{\Delta_s,\Xi_s}$
$\dots$
$\mathcal{L}^k_{\Delta_s,\Xi_s}$
1
2
k
k folds

Fig. 3. Principle of the leave-one-decision-out experiment (unused data are in grey).

TABLE II
STATISTICS OF THE HOURLY (H.) AND TRAJECTORY-WISE (T.) R^2-SCORES OBTAINED OVER THE 20 FOLDS OF THE LEAVE-ONE-DECISION-OUT EXPERIMENT.

	H. train score Seen dec. Seen scen.	H. test score Seen dec. Unseen scen.	H. test score Unseen dec. Unseen scen.	H. test score Unseen dec. Seen scen.
Mean	0.9876	0.9262	0.9207	0.9820
Std	0.0004	0.0074	0.0144	0.0062
Min	0.9863	0.9070	0.8984	0.9648
Max	0.9883	0.9360	0.9434	0.9889
	T. train score Seen dec. Seen scen.	T. test score Seen dec. Unseen scen.	T. test score Unseen dec. Unseen scen.	T. test score Unseen dec. Seen scen.
Mean	0.9852	0.9069	0.8987	0.9791
Std	0.0015	0.0101	0.0196	0.0057
Min	0.9813	0.8798	0.8703	0.9614
Max	0.9870	0.9200	0.9395	0.9866

To select the meta-parameters, we performed a gridsearch and kept for each fold the model with the combination of meta-parameters leading to the maximum score on the validation set. We thus have 20 proxies, each with a different day-ahead decision left out. The statistics of the test scores of these 20 proxies on the different test sets can be seen in Table II.

We observe that the scores of unseen decisions are almost equal to those of seen decisions. Therefore the proxy is able to generalise to unseen decisions. Furthermore, the scores are quite good, meaning that the proxy is able to predict the total preventive cost with acceptable performances. The trajectory scores are generally a bit smaller than the hourly scores but are still good. We also see that when the scenarios have already been seen by the neural network, the score is close to 1, even for unseen decisions.

2) Machine learning improvement: multitask learning: Instead of predicting only the target output with a proxy, we can try to simultaneously predict a vector of outputs; this corresponds to multitask learning [4]. In our case, this means predicting at the same time the target (total preventive cost) and auxiliary outputs as the preventive redispatch cost, the preventive load shedding cost and the preventive wind curtailment cost. The main advantage of this method is that the model can benefit from extra knowledge brought by the additional auxiliary outputs, so as to improve the performances of the proxy on the main target output.

We repeated the leave-one-decision-out experiment while exploiting this multitask learning approach. We performed

TABLE III
STATISTICS OF THE HOURLY (H.) AND TRAJECTORY-WISE (T.) R^2-SCORES OBTAINED OVER THE 20 FOLDS OF THE LEAVE-ONE-DECISION-OUT EXPERIMENT, WITH MULTITASK LEARNING.

	H. train score Seen dec. Seen scen.	H. test score Seen dec. Unseen scen.	H. test score Unseen dec. Unseen scen.	H. test score Unseen dec. Seen scen.
Mean	0.9868	0.9444	0.9387	0.9817
Std	0.0005	0.0057	0.0134	0.0052
Min	0.9860	0.9286	0.9049	0.9704
Max	0.9874	0.9523	0.9595	0.9884
	T. train score Seen dec. Seen scen.	T. test score Seen dec. Unseen scen.	T. test score Unseen dec. Unseen scen.	T. test score Unseen dec. Seen scen.
Mean	0.9845	0.9257	0.9157	0.9784
Std	0.0017	0.0080	0.0240	0.0066
Min	0.9797	0.9021	0.8634	0.9620
Max	0.9861	0.9351	0.9519	0.9864

again a gridsearch analysis with the same meta-parameters as before, but this time predicting a vector of outputs. We keep the network configuration maximising the validation score for the total preventive cost, since it is the target of interest, for each fold. Looking only at the predictions of the total preventive cost, we obtain the results presented in Table III.

We see that with multitask learning, we can improve the test scores of the proxy by 2%, which is quite interesting given that the scores were already close to the maximum score.

3) Impact of the number of training day-ahead decisions: In this experiment, we first selected randomly $l < k$ candidate decisions that we consider as our test decisions. Then we learn $k-l$ times a proxy, each time adding a new decision (different from the l test decisions) in the training set. At the end, we compare the $k-l$ test scores obtained both on the test set with seen decisions and unseen scenarios and the test set with unseen decisions and unseen scenarios and check when the test score corresponding to unseen decisions is similar to the one corresponding to seen decisions. This would indicate that enough decisions have been taken into account for a proxy to be able to generalise well to unseen decisions. Figure 4 illustrates the principle of this experiment.

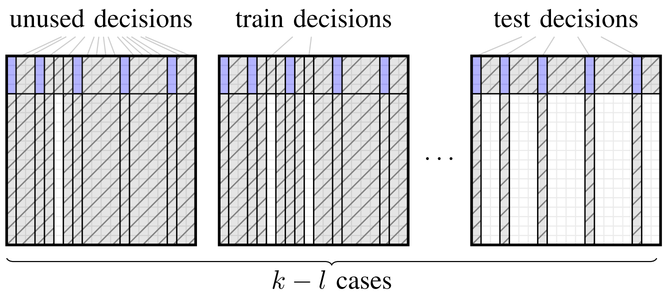


Fig. 4. Principle of the gradual increase of the learning set $\mathcal{L}$ size experiment.

Here we report the results obtained with $l = 5$. We use the best configuration of neural network on average on the validation score from the previous experiment (5 layers and 50 neurons per layer) and we repeated each experiment 5 times, with a different initialisation of the neural network weights. The results for the best initialisation (based on the validation score) are presented in Table IV. We see that the proxy is able to generalise well with only 5 decisions and that the scores are already quite good. With 10 decisions we get scores close to those reported for 19 decisions in Table II.

F. Using the proxies for ranking day-ahead decisions

In this subsection we consider the use of the learnt proxies in order to rank a set of unseen candidate decisions according to the expected next-day operating cost they would induce. Since we have seen that the proxies generalise well to unseen decisions, we conjecture that they could be used in order to identify a good day-ahead decision, while avoiding as much as possible to resort to heavy SCOPF computations over large samples of next-day scenarios combined with each candidate day-ahead decision.

We compare two methods exploiting the proxies to estimate the expected total preventive cost $\bar{y}^i$ associated to a day-ahead decision δ^i_{da}. For each method, we first select a subset Δ_s of candidate day-ahead decisions that we exploit to build a proxy. Since we noticed in Table IV that only 5 decisions are needed in the learning set for the proxies to generalise well on unseen decisions, we selected randomly 5 decisions and assigned them to the learning set. We performed this operation 5 times, each time with a different set Δ_s of size 5. We also realised this experiment with 10 decisions in the learning set, for comparison.

1) Method 1: For this method, we first generate 2000 next-day scenarios. We then apply the proxy to predict the total preventive cost $y^{i,j}$ for all $\delta^i_{da} \in \Delta_u$ and $\xi^j_{nd} \in \{\xi^1_{nd}, \xi^2_{nd}, ..., \xi^{2000}_{nd}\}$. Finally we average the predictions over the 2000 scenarios to have an estimate $\bar{y}^i_p$ of the expected total preventive cost for each decision in $\delta^i_{da} \in \Delta_u$.

2) Method 2: This method extends the previous one by using the control variates approach presented in [3] to correct a possible bias in the estimation $\bar{y}^i_p$ performed with the proxies. For that, we compute the estimated total preventive cost $\bar{y}^i_{cv}$ of decision $\delta^i_{da} \in \Delta_u$ as follows:

$$\bar{y}^i_{cv} = \bar{y}^i_p + \sum_{j=1}^{100} (y^{i,j} - y^{i,j}_p),$$

where the 100 scenarios belongs to the set Ξ_u.

3) Results: We consider as ground truth the average value of y presented in Table I and computed with 600 scenarios per decision. Note that with both methods, for each $\delta^i_{da} \in \Delta_s$, the estimated expected value of total preventive cost is the one presented in Table I, given that the SCOPF calls had to be made to build the proxy.

To analyse the quality of the ranking, we use the Kendall's tau coefficient and the Spearman's rank correlation coefficient. The Kendall's tau coefficient τ is defined as $\tau = \frac{C-D}{C+D}$, where C is the number of concordant pairs and D the number of discordant pairs. The Spearman's rank correlation coefficient ρ is defined as $\rho = \frac{cov(rg_X, rg_Y)}{\sigma_{rg_X} \sigma_{rg_Y}}$, where $cov(rg_X, rg_Y)$ is the covariance of the rank variables and σ_{rg_X} and σ_{rg_Y} are the standard deviations of the rank variables.

TABLE IV
MEAN HOURLY (H.) AND TRAJECTORY (T.) R^2-SCORES AS A FUNCTION OF THE NUMBER OF DAY-AHEAD (DA) DECISIONS IN THE LEARNING SET.

Nb of DA decisions in the training set	H. test score Seen decision Unseen scenario	H. test score Unseen decision Unseen scenario	H. test score Unseen decision Seen scenario	T. test score Seen decision Unseen scenario	T. test score Unseen decision Unseen scenario	T. test score Unseen decision Seen scenario
1	0.7329	0.6621	0.8234	0.6413	0.6272	0.7961
2	0.7888	0.7140	0.9001	0.7323	0.6840	0.8860
3	0.8241	0.8045	0.9426	0.7903	0.7582	0.9366
4	0.8611	0.8447	0.9603	0.8352	0.8131	0.9546
5	0.8620	0.8668	0.9703	0.8321	0.8384	0.9651
6	0.8861	0.8824	0.9739	0.8636	0.8581	0.9703
7	0.8969	0.8814	0.9745	0.8720	0.8544	0.9709
8	0.9046	0.8964	0.9776	0.8870	0.8737	0.9741
9	0.9103	0.9045	0.9797	0.8993	0.8829	0.9782
10	0.9120	0.9091	0.9801	0.8942	0.8881	0.9779
11	0.9239	0.9243	0.9805	0.9045	0.9015	0.9782
12	0.9241	0.9225	0.9820	0.9100	0.9073	0.9798
13	0.9265	0.9307	0.9821	0.9094	0.9049	0.9807
14	0.9257	0.9245	0.9824	0.9057	0.8963	0.9816
15	0.9257	0.9244	0.9818	0.9143	0.9099	0.9804

TABLE V
MINIMUM AND MAXIMUM KENDAL'S TAU COEFFICIENT AND SPEARMAN'S CORRELATION COEFFICIENT FOR BOTH ESTIMATION METHODS, WITH 5 OR 10 DECISIONS IN THE LEARNING SET.

		$\bar{y}_p$		$\bar{y}_{cv}$	
		Min	Max	Min	Max
5 decisions	τ	0.5158	0.8421	0.8211	0.9263
	ρ	0.7038	0.9534	0.9338	0.9835
10 decisions	τ	0.6211	0.8632	0.8211	0.9159
	ρ	0.7624	0.9654	0.9353	0.9820

Both metrics minimum and maximum values can be found in Table V for the different estimations. One can see that with the control variates approach, these metrics are closer to 1, meaning that there is a stronger relationship between the true ranking and the estimated one. One can also notice that the ranking is better when there is 10 decisions in the learning set, but at the cost of more SCOPF calls.

Table VI presents the different estimations of the real-time operating costs associated to a decision as well as the corresponding ranking for the best case experiment when there is only 5 decisions in the learning set. One can directly notice that the control variates approach allows to improve the estimation of the preventive total cost and thus the ranking, but it has a larger computational burden (50% more SCOPF calls) than method 1. Note that, even if the first decision in both estimated rankings is not the correct decision, one can see that the good decisions (small expected total preventive cost) are identified with both methods.

IV. CONCLUSIONS AND FUTURE WORK

With respect to the methodology proposed in [3], we took here the additional steps of (i) studying how to generalise over 'unseen' day-ahead decisions, and (ii) testing the usefulness of the learnt proxies for ranking candidate day-ahead decisions. To do so, we proposed a methodology to build automatically a database of candidate day-ahead decisions and next-day scenarios, to each combination of which we can apply a real-time operation simulator in order to compute the resulting next-day operating costs. We also proposed a methodology to build and validate proxies, exploiting such a database.

TABLE VI
TRUE AND ESTIMATED EXPECTED TOTAL PREVENTIVE COST PER DECISION AND THE ASSOCIATED RANKING $r(\cdot)$. THE DECISIONS USED TO LEARN THE PROXIES ARE COLORED IN RED.

δ^i	$\bar{y}^i$ $(\times 10^6)$	$\bar{y}^i_p$ $(\times 10^6)$	$\bar{y}^i_{cv}$ $(\times 10^6)$	$r(\bar{y}^i)$	$r(\bar{y}^i_p)$	$r(\bar{y}^i_{cv})$
2	1.642	1.639 $-$	1.635 $-$	2	19 $\wedge_1$	19 $\wedge_1$
19	1.656	1.626 $-$	1.626 $-$	19	2 $\vee_1$	2 $\vee_1$
1	1.661	1.681 $+$	1.646 $-$	1	11 $\wedge_2$	1 $=$
3	1.676	1.664 $-$	1.664 $-$	3	8 $\wedge_2$	3 $=$
11	1.689	1.643 $-$	1.672 $-$	11	3 $\vee_1$	11 $=$
8	1.691	1.663 $-$	1.706 $+$	8	1 $\vee_3$	6 $\wedge_1$
6	1.692	1.692 $\circ$	1.692 $\circ$	6	6 $=$	8 $\vee_1$
7	1.719	1.700 $-$	1.716 $-$	7	7 $=$	7 $=$
10	1.728	1.725 $-$	1.742 $+$	10	4 $\wedge_2$	9 $\wedge_1$
9	1.731	1.731 $\circ$	1.731 $\circ$	9	14 $\wedge_4$	10 $\vee_1$
4	1.732	1.711 $-$	1.761 $+$	4	10 $\vee_2$	4 $=$
20	1.740	1.728 $-$	1.788 $+$	20	20 $=$	20 $=$
16	1.746	1.728 $-$	1.796 $+$	16	16 $=$	16 $=$
14	1.748	1.716 $-$	1.807 $+$	14	9 $\vee_4$	12 $\wedge_2$
18	1.788	1.750 $-$	1.810 $+$	18	18 $=$	5 $\wedge_2$
12	1.797	1.797 $\circ$	1.797 $\circ$	12	12 $=$	14 $\vee_2$
5	1.798	1.798 $\circ$	1.798 $\circ$	5	5 $=$	18 $\vee_2$
13	1.855	1.855 $\circ$	1.855 $\circ$	13	15 $\wedge_1$	13 $=$
15	1.872	1.816 $-$	1.871 $-$	15	13 $\vee_1$	15 $=$
17	1.918	1.857 $-$	1.905 $-$	17	17 $=$	17 $=$

We showed with a case-study on the three-area IEEE-RTS96 benchmark that our proxies of real-time operation, predicting the next-day preventive costs, are able to generalise well to unseen decisions. We also showed that they can be exploited to identify candidate decisions with smallest expected induced costs in real-time operation.

There are many possible future directions of research following this work. We highlight the following ones:

- how to improve the ranking methodology;
- how to adapt the proposed methodology to evaluate, instead of the expected real-time operation costs for a given decision, the probability to meet the reliability target in real-time and in particular be able to evaluate the probability of rare events;
- how to exploit the presented methodology to reverse the problem and find hints for the operation planner about what forecast scenarios could lead him to a 'good' day-ahead decision.

Acknowledgement

The authors thank the Isaac Newton Institute for Mathematical Sciences, University of Cambridge, for support and hospitality during the programme Mathematics of Energy Systems where work on this paper was initiated. This work was supported by EPSRC grant no EP/R014604/1. They warmly thank the support of RTE-France (the French Transmission System Operator) and *Consortium des Équipements de Calcul Intensif* funded by the *Fonds de la Recherche Scientifique de Belgique* (F.R.S.-FNRS) under Grant No. 2.5020.11 and the Walloon Region for providing the computational resources. LW acknowledges the support of F.R.S.-FNRS for his sabbatical year and the Simons Foundation for his fellowship during his stay at the Isaac Newton Institute.

References

[1] GARPUR Consortium, "D2.2 - guidelines for implementing the new reliability assessment and optimization methodology," 2016, available at: http://www.garpur-project.eu/deliverables.

[2] F. Capitanescu, "Critical review of recent advances and further developments needed in AC optimal power flow," *Electric Power Systems Research*, vol. 136, pp. 57–68, 2016.

[3] L. Duchesne, E. Karangelos, and L. Wehenkel, "Using machine learning to enable probabilistic reliability assessment in operation planning," in *2018 Power Systems Computation Conference (PSCC)*. IEEE, 2018, pp. 1–8.

[4] R. Caruana, "Multitask learning," *Machine learning*, vol. 28, no. 1, pp. 41–75, 1997.

[5] L. A. Wehenkel, *Automatic learning techniques in power systems*. Springer Science & Business Media, 2012.

[6] J.-M. H. Arteaga, F. Hancharou, F. Thams, and S. Chatzivasileiadis, "Deep learning for power system security assessment," in *13th IEEE PowerTech 2019*. IEEE, 2019.

[7] B. Donnot, I. Guyon, A. Marot, M. Schoenauer, and P. Panciatici, "Optimization of computational budget for power system risk assessment," in *2018 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe)*. IEEE, 2018, pp. 1–6.

[8] J. L. Cremer, I. Konstantelos, S. H. Tindemans, and G. Strbac, "Data-driven power system operation: Exploring the balance between cost and risk," *IEEE Transactions on Power Systems*, vol. 34, no. 1, pp. 791–801, 2018.

[9] F. Thams, L. Halilbasic, P. Pinson, S. Chatzivasileiadis, and R. Eriksson, "Data-driven security-constrained opf," in *X Bulk Power Systems Dynamics and Control Symposium*, 2017.

[10] L. Duchesne, E. Karangelos, and L. Wehenkel, "Recent developments in machine learning for energy systems reliability management," *Proceedings of the IEEE*, in press.

[11] F. Fioretto, T. W. Mak, and P. Van Hentenryck, "Predicting AC optimal power flows: Combining deep learning and lagrangian dual methods," *arXiv preprint arXiv:1909.10461*, 2019.

[12] X. Pan, T. Zhao, and M. Chen, "DeepOPF: Deep neural network for DC optimal power flow," in *2019 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*. IEEE, 2019, pp. 1–6.

[13] A. Zamzam and K. Baker, "Learning optimal solutions for extremely fast AC optimal power flow," *arXiv preprint arXiv:1910.01213*, 2019.

[14] G. Dalal, E. Gilboa, S. Mannor, and L. Wehenkel, "Unit commitment using nearest neighbor as a short-term proxy," in *2018 Power Systems Computation Conference (PSCC)*. IEEE, 2018, pp. 1–7.

[15] ——, "Chance-constrained outage scheduling using a machine learning proxy," *IEEE Transactions on Power Systems*, 2019.

[16] M. Håberg, "Fundamentals and recent developments in stochastic unit commitment," *International Journal of Electrical Power & Energy Systems*, vol. 109, pp. 38–48, 2019.

[17] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed. Springer Series in Statistics, 2009.

[18] P. Wong, P. Albrecht, R. Allan, R. Billinton, Q. Chen, C. Fong, S. Haddad, W. Li, R. Mukerji, D. Patton *et al.*, "The ieee reliability test system-1996. a report prepared by the reliability test system task force of the application of probability methods subcommittee," *Power Systems, IEEE Transactions on*, vol. 14, no. 3, pp. 1010–1020, 1999.

[19] H. Pandzic, Y. Dvorkin, T. Qiu, Y. Wang, and D. Kirschen, "Unit Commitment under Uncertainty - GAMS Models," Library of the Renewable Energy Analysis Lab (REAL), University of Washington, Seattle, USA, [Online]. Available at: http://www.ee.washington.edu/research/real/gams_code.html.

[20] L. Duchesne, "Machine learning of proxies for power systems reliability management," Master's thesis, Université de Liège, Liège, Belgique, 2016.

[21] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[22] Scikit-learn developers, "Model evaluation: quantifying the quality of predictions," [Online], Available at http://scikit-learn.org/stable/modules/model_evaluation.html, accessed on 2016-05-11.

Appendix

Computing times

The average computing times of the experiments presented in this paper, with a MacBookPro (2.2GHz Intel Core i7, 16GB RAM), are presented in Table VII.

In the upper part of the Table, we highlight the CPU times needed to assess a single scenario and day-ahead decision, composed of first sampling the scenario, and then computing the costs induced by the decision either via the detailed SCOPF-wise simulation or via applying the learnt proxy. We observe a gain of a factor 10,000 with the ML proxy.

TABLE VII
Average computing times

	Average time (s)
Sampling of one scenario	0.002
Real-time SCOPF simulation for one scenario	303.600
Using the ML proxy for one scenario	0.027
Learning/validation/test dataset generation	3,643,200.000
Learning one proxy	5,660.000
Optimising the meta-parameters of the proxy	169,813.000

The CPU times in the lower part of the Table correspond to the off-line learning stage based on the leave-one-decision-out experiment. We observe that the bulk of the computations are about the dataset generation (corresponding in our simulations to 12,000 trajectories (i.e. 600 scenarios combined with 20 day-ahead decisions) times 24 hours, while optimising the meta-parameters of the machine learning method corresponded in our case to tuning 30 different proxy models. It is important to realise that the computing times of both the dataset generation and the meta-parameters' optimisation parts could be reduced by using massive parallel computations, respectively to about 303 seconds and 5,660 seconds.