Reinforcement learning for Energies of the future and carbon neutrality: a Challenge Design

Gaëtan Serré, Eva Boguslawski, Benjamin Donnot, Adrien Pavão, Isabelle Guyon, Antoine Marot

Reinforcement learning for Energies of the future and carbon neutrality: a Challenge Design

Gaëtan Serré[1]
gaetan.serre@universite-paris-saclay.fr

Eva Boguslawski[1, 3]
eva.boguslawski@rte-france.com

Benjamin Donnot[3]
benjamin.donnot@rte-france.com

Adrien Pavão[1]
adrien.pavao@gmail.com

Isabelle Guyon[1, 2]
guyon@chalearn.org

Antoine Marot[3]
antoine.marot@rte-france.com

[1]LISN/CNRS/INRIA, University of Paris-Saclay, Gif-Sur-Yvette, France, [2]ChaLearn, [3]RTE France

***Abstract*—Current rapid changes in climate increase the urgency to change energy production and consumption management, in order to reduce carbon and other greenhouse gas production. In this context, the French electricity network management company RTE (Réseau de Transport d'Électricité) has recently published the results of an extensive study outlining various scenarios for tomorrow's French power management [10]. We propose a challenge that will test the viability of such scenarios [1]. The goal is to control electricity transportation in power networks while pursuing multiple objectives: balancing production and consumption, minimizing energetic losses, keeping people and equipment safe, and particularly avoiding catastrophic failures. While the importance of the application provides a goal in itself, this challenge also aims to push the state-of-the-art in a branch of Artificial Intelligence (AI) called Reinforcement Learning (RL), which offers new possibilities to tackle control problems. In particular, various aspects of the combination of Deep Learning and RL called Deep Reinforcement Learning remain to be harnessed in this application domain. This challenge belongs to a series started in 2019 under the name "Learning to run a power network" (L2RPN). In this new edition, we introduce new more realistic scenarios proposed by RTE to reach carbon neutrality by 2050, retiring fossil fuel electricity production, increasing proportions of renewable and nuclear energy and introducing batteries. Furthermore, we provide a baseline using a state-of-the-art reinforcement learning algorithm to stimulate future participants.**

***Index Terms*—power network, carbon neutrality, global warming, renewable energy, reinforcement learning**

I. Introduction

Power Systems enable energy transportation from places where it is produced (nuclear or fossil power plants, hydroelectric generators, wind turbines, solar panels, etc.) to places of consumption (e.g. houses, factories, public lighting, etc.). It is a vital component of our society; it has become so common that it is often taken for granted, although it constantly relies on thousands of kilometers of transmission lines and the work of thousands of people. Power systems are currently facing systemic changes, which bring current technology to its edge. Recent successes achieved in AI by Deep Learning techniques [18], including Deep Reinforcement Learning [14], have drawn the attention of the Power Systems community for several reasons: their capacity to learn representations, and their parallelizable architectures.

For the fourth edition of the Learning to Run a Power Network challenge (L2RPN'2022), we look ahead to 2050, in a context of carbon neutrality, by drastically reducing the share of electricity produced by fossil fuels and increasing the share of renewable energies in the power system's energy mix. In this section we briefly review the current context motivating the creation of such a challenge and the problems posed to the AI community, particularly those resulting from the massive use of renewable energy.

A. Energy shift

a) Global warming: In the late 2010s, around 85% of the energy produced came from the combustion of fossil fuels that emit greenhouse gas such as (but not limited to) CO_2. Those emissions have been consistently growing since the start of the industrial era. It is nowadays commonly admitted that the negative impact of modern societies on the environment has become non-negligible since the 1950s. To prevent the irreversible destruction of an ecosystem, which we need for our survival, it has become urgent to drastically reduce, among other things, the emission of greenhouse gas [21].

b) Increasingly uncertain power injection patterns: Political leaders have been pushing toward the development of alternative energy conversion devices that exploit renewable and low-carbon forms of energy, such as solar radiation and wind. Devices that harness those sources of energy have drastically improved over the past two decades, which has enabled their large-scale deployment. A growing amount of research is dedicated to investigating the feasibility of a 100% renewable energy system in the medium term and advocating for massive use of the latter two technologies.

Unfortunately, solar and wind power come with some drawbacks with regard to their integration in power networks [7]:

- Their production highly depends on the weather. This may cause imbalances in the power network.
- Our energy storage capacity is low, which promotes controllable generators, as opposed to intermittent generators such as solar and wind power. It is therefore mandatory to have controllable generators in reserve.

B. Complexity of power network operations

The electric power network can be broken down into main functions: production (power generation), transport (power lines), and consumption (end users). The transport part is usually split into the "transmission system" (long distances, e.g. from a power plant to a city) and the "distribution system" (local scale, e.g. within a city).

RTE is in charge of managing the French transmission system in real-time and ensures that the production equates to the consumption. It anticipates the impacts of potential outages, whether these are planned or accidental. Dispatchers (highly trained engineers) ensure the system's security by performing several actions, including [6]:

- managing power overflows (which can endanger trees, roads, infrastructures or passers-by) and preventing cascading failures (leading to blackouts of the whole system), by changing interconnection patterns of transmission lines, to redirect power flows);
- asking producers or consumers to change what they inject into the power network (for example, remunerating a producer at a specific localization to avoid a local overload);
- in the future, maybe modifying the amount of power produced or absorbed by storage units, such as batteries;
- when required, limiting the amount of energy injected by renewable generators (such as wind or solar) in case of overproduction or local issues for example.

In all cases, dispatchers have to rely on their thorough understanding of the system. Current optimization-based methods are struggling with the complexity of both problems, and some satisfying heuristics exist or are in the process of being experimented. The hope is that AI could assist dispatchers in making better decisions to efficiently control the power network and keep all equipment in security.

II. Previous challenges

The "Learning to run a power network" (L2RPN) challenge [16], [17] is a series of competitions that model the sequential decision-making environments of real-time power network operations, as illustrated in Fig. 1. The participants' algorithms must control a simulated power network, in a reinforcement learning framework.

a) Power networks and data: The physical simulation is based on Python's module Grid2Op which we detail in section IV-A. Power networks of various sizes and topologies are used across competition rounds.

b) Results and main outcomes: In the 2019 edition, the provided power network, a slightly adapted version of the IEEE 14-bus network, was composed of 20 power lines, 11 loads and 5 generators. The winner team of this edition [12] used the Double Deep Q-Learning algorithm [27] along with imitation learning to initialize the policy. In 2020, the power network was much more complex. It was composed of 59 power lines, 37 loads and 22 generators. There were two competitions that year. The winner team of the first one [28] was a team of experienced researchers in reinforcement learning, that have won similar competitions in NeurIPS 2018 and 2019: "Learning to Run" and "Learning to Move" [31]. First, they perform an action space reduction to 1000 elements using simple expert systems and initializes a policy parameterized by a feed-forward neural network with millions of parameters. Then, they train a policy using evolutionary black-box optimization [13]. This functioning, using evolution strategies, can be opposed to most standard RL strategies, such as Deep Q Network (DQN) [19], [20], which rely on gradient descent. The winner team of the second one [30] used a policy neural network to select the Top-K actions and applied an optimization algorithm to choose the best one.

Overall, this latest competition has shown encouraging results about how artificial intelligence methods can be successfully applied to the problem of operating power networks. The Grid2Op platform, as a simulator lowering the barriers to entry in the specific domain of electricity network management, permitted a research team with no knowledge in this domain to produce a satisfying model. Solutions could be extended to more complex cases, especially as the power systems are evolving to meet decarbonization, which leads to increasingly difficult decision problems for operators.

III. Why a new competition

The French electricity network management company RTE has recently published the results of an extensive study outlining various scenarios for tomorrow's French power management [10]. Due to the ecological concerns, all scenarios mostly rely on nuclear and renewable energies. The Paris region, Ile-de-France, being particularly concerned, proposed two milestones [9]:

- **By 2030**: Reduce by half the dependence on fossil fuels and nuclear power in the Ile-de-France region. This would be achieved by both reducing the energy consumption by 20% and by multiplying by 2 the energy production from renewable sources.
- **By 2050**: Moving towards a 100% renewable energy and zero carbon region. This would be achieved by both reducing the energy consumption by 40% and by multiplying by 4 the energy production from renewable sources.

In this context, it is necessary to re-factor the problem tackled by past L2RPN challenges [16], mainly by updating the simulator and the data to represent zero carbon scenarios [10]. As explained in section I-A, such scenarios are even harder to control, therefore advancing Artificial Intelligence methods could be particularly useful.

IV. Competition design

In order to organize the 2022 edition of Learning to Run a Power Network, we need an environment that will simulate the behavior of a power system during a defined time (e.g. one week) that we will call scenario. To simulate such a scenario, the environment needs data and more specifically time series describing the electricity injections in the power

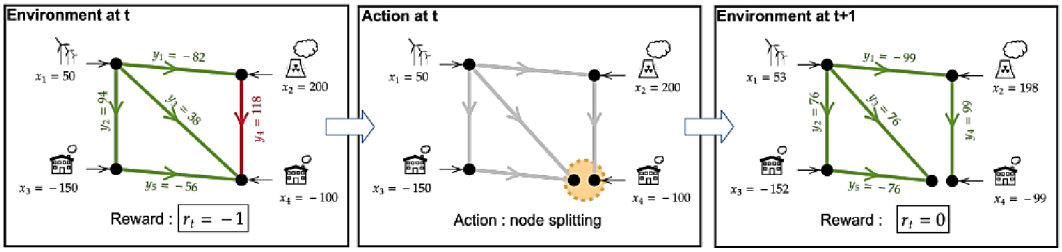

Fig. 1. Power system operation: The task of dispatchers is to monitor the power network and make eventual changes to ensure safe network operation with no line overflow. If in the environment at time t (left) a line is overflowing (indicated in red), a corrective action may be taken (center), such as a "node splitting", resulting in restored "power network safety" in the environment at time t+1 (right). Borrowed from [16].

network (referred to as chronics). In this section, we describe the specificities of the simulation environment used as well as the chronics. In addition, we also look at the details of the organization of the competition on an online platform and the metric used to rank the participants.

A. Simulation environment and chronics

a) Grid2Op: To run a L2RPN competition, we need a library capable of simulating a power system in a reinforcement learning framework. RTE has therefore developed Grid2Op [5], a Python module that casts the operational decision process into a Markov Decision Process (S, A, P_a, R_a) [2]. Grid2Op will therefore discretize the time of a scenario into a list of states corresponding to a time step of 5 minutes. For example, a one day scenario will be discretized in $24 * 60/5 = 288$ time steps. Then, for a state $s_t \in S$ and an action $a_t \in A$, Grid2Op will calculate s_{t+1} i.e. the power flow (the amount of electricity flowing on each power line) at time $t + 1$. For this, it will need the chronics at time t. We detail the generation of these data in the next paragraph. In addition, Grid2Op uses the Gym interface developed by OpenAI [4] to interact with an agent. Also, a set of startup notebooks is available to facilitate its handling. Thanks to these two points and as previous editions have confirmed, future participants don't need to have strong knowledge in the field of power systems to create efficient agents. This corresponds perfectly to our desire to create a competition open to all and oriented towards reinforcement learning.

b) Chronics: As described in the previous paragraph, to make Grid2Op work, we need to generate time series describing the electricity injections into the power network. These time series are referred to as chronics. An injection is the amount of electricity that is injected into the power network by generators, loads and batteries. The generators inject a positive amount of electricity while the loads inject a negative amount. Batteries can inject either a negative or positive amount of electricity depending on whether they are storing or delivering electricity. The sums of the injections must be equal to 0 at all times for the power network to work (taking into account the loss of electricity in heat). To generate these chronics, we need data concerning the architecture of the power network, the weather, the consumptions and the generators (e.g. their types and maximum production). These data, especially about the consumptions and the weather, come from RTE studies. Concerning the power network, we are using an even more complex power network than in previous years. It is composed of 186 power lines, 91 loads and 62 generators. In addition, we have added 7 batteries that can be used by agents to store and deliver electricity. Then, a library created by RTE named Chronix2Grid [15], uses these data to generate chronics. Fig 3 is a example of such chronics. From the chronics, we deduce the energy mix of our power system:

$$em_{gt} = \frac{\sum_{g \in G_{gt}} \sum_{t=0}^{T} \epsilon_{g,t}}{\sum_{g \in G} \sum_{t=0}^{T} \epsilon_{g,t}} , \quad (1)$$

where gt is a generator type that belongs to the set $\{nuclear, solar, wind, thermal, hydro\}$, em_{gt} is the percentage of electricity produced by generators of type gt, G_{gt} is the set of the generators of type gt, G is the set of all the generators, T is the maximum time step of the scenario, and $\epsilon_{g,t}$ is the injection of generator g at time step t.

To generate this edition's chronics, we privileged renewable generators and penalized the use of fossil fuel generators (referred to as *thermal*) in Chronix2Grid. In addition, we have given it the possibility to curtail the excess of renewable energy, if needed, when creating the chronics. This has allowed us to significantly increase the power of renewable generators. As a result, we were able to generate chronics with an almost carbon-free energy mix as shown in Fig. 2. With less than 3% of electricity generated by fossil fuels, this energy mix is very satisfactory for our competition, so we have generated 32 years of scenarios that are available to participants to train their agents. Moreover, they can generate more scenarios with the same specifications through Grid2Op.

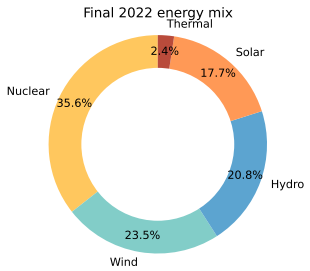

Fig. 2. L2RPN 2022 energy mix over a year.

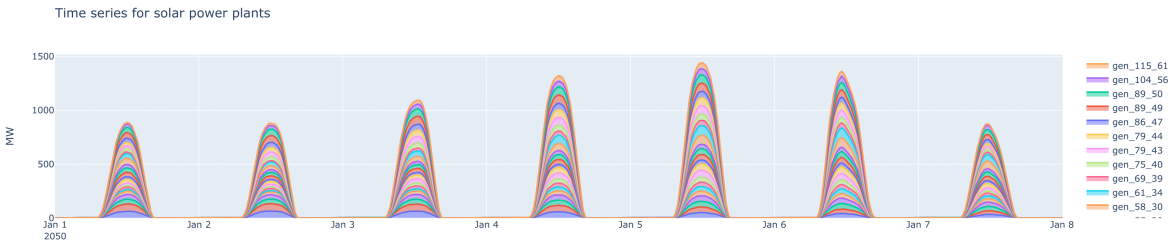

Fig. 3. Example of time series representing the energy produced by each solar power plant at each time step.

B. Hosting on Codalab

To facilitate participation, we implemented the L2RPN'2022 competition on Codalab as follows:

- **Competition with code submission**: RL agents capable of controlling the power network will be blind tested on the platform with new scenarios not known to the participants. These scenarios were carefully chosen to be representative of the different problems encountered by power network operators.
- **Starting kit**: We provide a set of tools and tutorials to help participants getting started, including power network visualization and diagnosis tools (Grid2Viz), and a white paper describing the problem and baseline methods. A sample submission with the code of a baseline agent, following a designated API, is provided. Sample scenarios are supplied. They are chosen with the same criteria as those used to test the agents on Codalab, but they are not the same. The starting kit is available here. The execution of a task consists in repeating the following RL-style steps, until time is out or a blackout occurs:
  - suggestion = agent.act(observation)
  - observation = environment.step(suggestion)
- **Protocol**: Participants will need to train their agent on their own machine or cloud server, and submit trained agents. We will use a three-phase competition protocol: (0) warm-up phase: participants try the starting kit, can ask for modifications of the computational resources and the available packages; (1) development phase: packages and computational resources are frozen; participants get feedback on their submissions on a leaderboard, and (2) final phase: a single final submission is evaluated on new unseen scenarios. This evaluation is the only one counting for the final ranking.
- **Timeline**: The project was accepted as an official IJCNN/WCCI'22 in January 2022. We opened the warm-up phase on June 15, and the development phase July 5. The final test phase will start September 15, and the results will be revealed September 30.

C. Metric

To rank the participants, we need a **score function** which assigns a real number to each agent evaluating its performance. To that end, we created a score function that is the average of these three cost functions over the test scenarios:

- **Cost of energy losses**: Calculated by multiplying the amount of electricity lost due to the Joule effect by the current price of the MWh.
- **Cost of operation**: Sum of the costs of the agent's actions. Operations involving changes in the production of electricity have a cost that depends on the energy market. The use of batteries has a fixed cost per MWh.
- **Cost of blackout**: If the agent did not manage the power network until the end of the scenario, this cost is calculated by multiplying the amount of electricity left to supply by the current price of MWh.

Note that, as expected, the cost of a blackout is much higher than the two other costs, which means that an agent who succeeds in a scenario will always have a better score than an agent who has not succeeded, even though its actions are less costly.
Moreover, our score function is normalized so that it is to be maximized and is between the bounds $[-100, 100]$. A score of 0 corresponds to an agent that does nothing at each time step. Having a positive score is already pretty good.

D. Setting recap

We summarize the setting of the optimization problem to be solved, at every step:

- **Observation space**. Complete state of the power network: all information over power nodes (electricity produced and consumed), flows of each power lines, and more.
- **Action space**. Four types of actions allowed:

  1. Line status (line connection/disconnection).
  2. Topology changes (node splitting). (2)
  3. Power production changes/curtailment (of generators).
  4. Storage changes (storage or delivery from batteries).

  For the 2022 edition, the action space still contains over 70,000 discrete actions (topology changes) and 69-dimensional continuous action space (production changes).
- **Reward**. The participants are free to design their own reward function. However, the leaderboard metric is defined in Section IV-C.
- **Game over condition**. A game over is triggered if total demand is not met anymore (taken into account in the metric as "cost of blackout").

V. BASELINE

In addition to providing a more complex power network with a carbon neutral energy mix, for the 2022 edition of L2RPN we also provide a baseline using reinforcement learning (RL). It has a relatively simple architecture but performs quite well on the validation scenarios. The goal is on one hand, to give an example of a simple agent using reinforcement learning and, on the other hand, to stimulate the competition. Furthermore we also wanted to give a working example to the participants to leverage the new types of actions at their disposal: curtailment and action on storage units. This is why our baseline agent only uses these new actions.

A. Prior art

The efficiency of reinforcement learning (RL) to solve complex sequential decision problems has been demonstrated many times. For example, Go [25], Chess [24] and even complex video games like Dota 2 [3] have recently made great strides, thanks to RL algorithms. The use of RL in power system operation has also been illustrated [8], [29], and the winners of previous Learning to Run a Power Network competitions [12], [28], [30] have used RL approaches.

B. Architecture

The architecture of our baseline agent is illustrated in Fig. 4. It takes all data concerning power lines, generators, and storage units from the power network state as "observation", and first checks whether it improves the system state by performing an "obvious action" with "expert rules" (specifically: line reconnections or do-nothing). If not fruitful, the agent uses a "trained policy" to choose another action, involving a parameterized neural network, trained with an Actor-Critic algorithm [11]. This architecture is more efficient than just using a trained policy network because simple expert rules maintain the power network well in most situations.

C. Proximal Policy Optimization

The RL part of the agent focuses on continuous actions from Eq.2, which are: 3. curtailment and 4. storage units, described in more detail Sec. IV-D. We chose the Proximal Policy Optimization (PPO) algorithm [23], which has had success in previous editions of our competition, and is generally known to be very efficient in complex environments with a continuous action space. It is used e.g. in MuJoCo [26] or Roboschool [23].

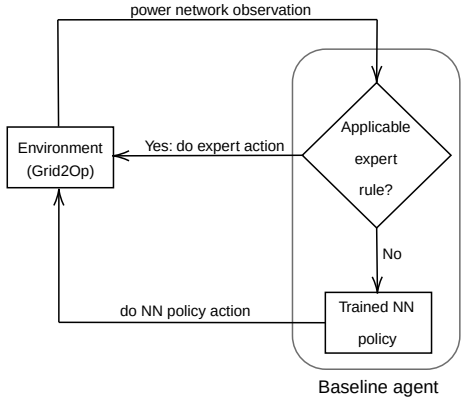

Fig. 4. Overview of our baseline architecture

PPO is a policy gradient and Actor-Critic algorithm: it uses an approximate value of the cumulative sum of the rewards (say V_t) to "criticize" and update the policy. Generally, this type of algorithm has two neural networks, one which returns V_t, called *critic network*, and one which delivers the policy, called *policy network*.

In the vanilla policy gradient algorithm, the V_t value is used to compute an estimator of the quality of the action chosen by the policy at time t:

$$\widehat{A}_t = \sum_{k=0}^{T-t} \gamma^k r_{t+k} - V_t \tag{3}$$

where T is the number of time steps of the episode, $0 \leq \gamma \leq 1$ is the discount parameter, and r_t is the reward obtained at time t. $\widehat{A}_t > 0$ means that the *critic network* predicts that the action chosen by the policy at time t is good so the algorithm will update the policy in order to increase the probability of doing this action and vice-versa if $\widehat{A}_t < 0$. The *critic network* is used only during training and it is trained using the MSE loss function to approximate the cumulative sum of rewards. However, at the beginning, because of the random initialization of the weights of the *critic network*, the value of $\widehat{A}_t$ is random. Thus, many times, the algorithm will update the policy in the wrong direction, which explains the instability and the slow convergence of the vanilla policy gradient algorithm.

PPO tries to solve this problem by proposing an objective that optimizes the policy, while penalizing too large updates. The objective is to find the θ parameters that maximizes:

$$\mathcal{L} = \mathbb{E}_t \left[min \left(\widehat{A}_t q_t(\theta), \widehat{A}_t clip \left(q_t(\theta), 1-\epsilon, 1+\epsilon \right) \right) \right] \tag{4}$$

Where ϵ is a hyperparameter (0.2 in the original PPO paper) and $q_t(\theta)$ is the ratio of the probability of doing the action a_t at state s_t between the new policy parameters θ and the previous θ_{old}:

$$q_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)} \tag{5}$$

Where $\pi_\theta(a_t|s_t)$ is the probability of doing the action a_t at state s_t using the distribution parameterized by θ. When $q_t(\theta) > 1$ it means that the probability of doing the action a_t at state s_t became more probable with the new distribution parameters. In the other hand, $q_t(\theta) < 1$ means that the probability of doing the action a_t at state s_t became less probable.

Thanks to this term and the min and $clip$ functions, PPO limits the chance of making destructively large policy updates to prevent the agent from going in a direction that looks good but turns out to be a bad one. This behavior is very well explained by Fig. 1 of the original paper [23]. PPO is therefore more stable and trains faster than vanilla policy gradient algorithm.

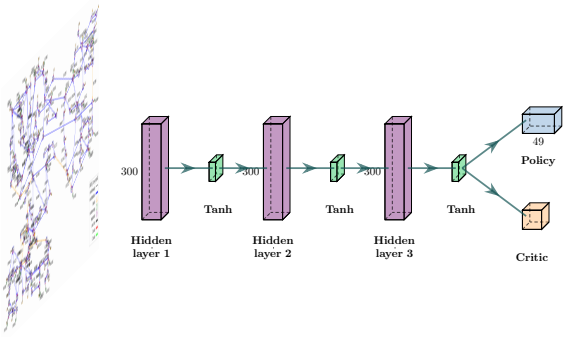

Fig. 5. Architecture of our baseline agent Actor-Critic network

D. Experimental setting

a) Actions: Four types of actions are possible in principle (Equation2), but our baseline method excludes actions of type 2 (node splitting). The rule-based part of the agent performs type 1 actions (line reconnections); the RL part of our baseline agent only performs two types of actions: 3. curtailment actions and 4. storage actions. The former can modify the production of renewable generators and the latter can store or deliver electricity to the batteries. Both actions are continuous.

b) Actor-Critic neural network: Our neural network is a Multi-Layer-Perceptron. Its architecture is illustrated by Fig. 5. It is composed of 3 hidden layers of 300 neurons each which are shared by the Critic network and the Actor network. The input shape is 1225, the output shape of the Critic network is 1 and the output shape of the policy network is 49, which is consistent with the possible actions detailed above since there are 42 renewable generators and 7 batteries. We use the $tanh$ activation function between the hidden layers.

c) Reward: The reward used to train our PPO agent is defined as: if the game is over, it returns the ratio between number of time steps survived by the agent and number of time steps of the scenario; otherwise, it returns 0.

d) Expert rules and associated hyperparameters (HP): The expert rules of our baseline agent shown in Fig. 4 are:

- **Reconnect all possible power lines.** Some lines may have been disconnected because of a maintenance for example. We assume that the more lines are connected, the less likely it is that the power network will be overloaded.
- **Do nothing else (if possible); HP *safe_max_rho*.** If the power network is not close to being overloaded by doing nothing, then perform the *do nothing* action. Let l be the most loaded power line and ρ_l the value of its load (ratio between amount of electricity passing through l and its maximum capacity). If $\rho_l <$ *safe_max_rho*, *do nothing*, else use the PPO policy.
- **Limit action impact; HP *limit_cs_margin*.** The HP *limit_cs_margin* limits the impact of an action on the power network. This last rule is motivated by empirical observations that RL agents have trouble producing smooth actions and change multiple generator states in a single action. This often leads to a premature "game over" by violating some constraints of the environment.

e) Neural Network software and hyperparamaters: To implement our PPO agent we used the Python Stable Baselines 3 library [22].[1] The training hyperparameters are described in Table I[2].

f) Training data: Preliminary experiments that we conducted revealed that training on all scenarios (available from the public dataset of the competition) resulted in worse results than "cherry picking" scenarios. To alleviate this problem, we limited training to scenarios taken from the most difficult week of the year (one week in February, when power consumption is high). More systematic experiments to optimize the training curriculum are under way.

g) Validation & Test data: The results presented in this paper use the validation set of the "development phase" to select hyperparameters (of the expert rules and the neural network) and the test set of the "test phase" of our competition on Codalab[3] to report results.

E. Results

We trained 14 agents for 10 million iterations. In the following figures, we compare several agents with the *Do Nothing* agent (does nothing at each time step) and the *Expert only* agent (uses only our experts rules defined in Sec. V-D).

To improve the score of our agent (Fig. 4), we tuned (using validation data) the two hyperparameters described in Sec. V-D: *safe_max_rho* and *limit_cs_margin*. They limit the impact of the action of our RL agent on the power network, using expert rules, to avoid erratic actions triggering early "game over".

To find the best combination of these hyperparameters, we set $limit\_cs\_margin = 60$ (a medium value chosen empirically) and look for the best corresponding value of *safe_max_rho*. As illustrated in Fig. 6, this value is 0.99. We then set $safe\_max\_rho = 0.99$ in order to look for the best corresponding value of *limit_cs_margin*. As illustrated in

[1] version 1.5.0. The code to reproduce these results and Fig. 6, 7 and 8 is available here.

[2] oracle: for training, we rely on the environment to limit the impact of the action in case the action is too heavy. It is not possible at inference time as the environment cannot be modified by our agent.

[3] https://codalab.lisn.upsaclay.fr/competitions/5410

TABLE I
BASELINE AGENT TRAINING HYPERPARAMETERS

Model type	Multi-Layer-Perceptron
Input shape	1225
Output shape	(1, 49)
Shared hidden layers	3 of 300 neurons each
Loss function	PPO loss for policy and MSE for critic
Batch size	16
Environment steps	16
Gamma	0.999
Epochs loss optimization	10
Optimizer	ADAM
Learning rate	$3e^{-6}$
safe_max_rho	0.2
limit_cs_margin	oracle

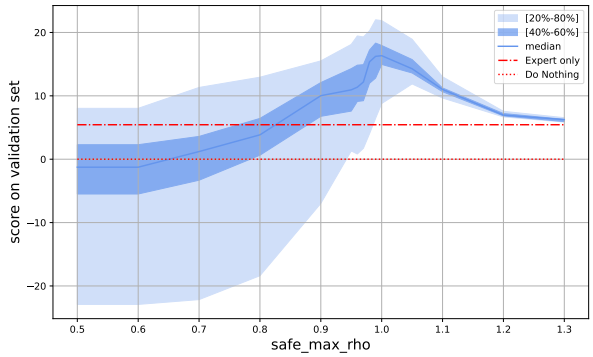

Fig. 6. Baseline score (as defined in Sec. IV-C) averaged over all validation scenarios as a function of *safe_max_rho* at evaluation time. $limit\_cs\_margin = 60$.

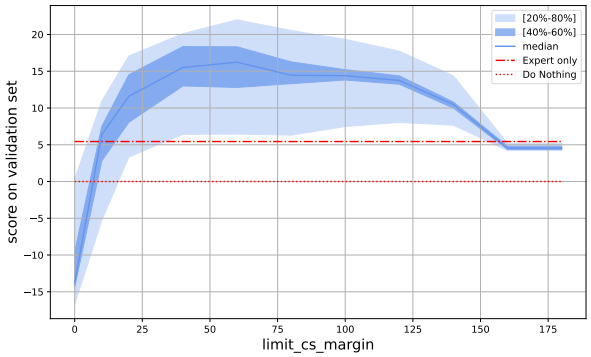

Fig. 7. Baseline score (as defined in Sec. IV-C) averaged over all validation scenarios as a function of *limit_cs_margin* at evaluation time. $safe\_max\_rho = 0.99$.

Fig. 7, this value is 60. With this method, the best combination of these hyperparameters is $safe\_max\_rho = 0.99$ and $limit\_cs\_margin = 60$. A large *safe_max_rho* results in using the "RL part of our agent" only in critical states of the power network, thus avoids doing potentially destructive actions, when doing nothing is enough. In addition, 60 is an intermediate value of *limit_cs_margin*, which is a good compromise between limiting and preserving the action. With these parameters, our best agent has a score of 22.46 on the validation scenarios and 26.80 on the scenarios of the test phase of our competition. This agent is thus much better than the *Do Nothing* or *Expert only* agents (higher scores are better).

To take advantage of the stochasticity of the score of our agent, depending on its initialization and evaluation scenarios, we created a mixture of experts to choose the best action, taking advantage of the strengths of various RL agents. Grid2Op allows us to estimate of the reward obtained if we do the action a at state s_t. Our mixture of expert algorithm therefore implements a "look ahead" policy that simulates the actions of the available RL agents to choose the action that brings the best approximation of the reward. To evaluate this strategy, we have used 14 instances of our baseline trained previously. Fig. 8 illustrates the performance of our mixture of experts algorithm on the validation set, compared with other agents. This algorithm obtains a score of 23.58 on the validation set and 24.47 on the scenarios of the test phase of our competition. These scores are quite similar to those of our baseline agent. However, in some scenarios, our mixture of agents outperforms any instance of our baseline. This strategy looks promising and many enhancements are possible.

VI. Conclusion

In this paper, we presented the design of the fourth edition of "Learning to Run a Power Network challenge", focusing on "energies of the future and carbon neutrality". This competition targets the real-world problem of ensuring the safety of power networks, using a lot of renewable energies and several batteries, with a focus on real-time operations. We provided a baseline agent to the participants, combining heuristic rules and a trained RL agent, to lower the barrier of entry and stimulate participation. This baseline performs quite well, but could still be improved. Indeed, we show in this paper a first promising avenue: creating random mixtures of RL agents. This could be further enhanced by specializing the RL agents on subsets of scenarios, e.g., around given times of the year. Other improvements could be made by exploiting actions of type 2: Node splitting. Finally, more sophisticated but slower optimization algorithms could be used off-platform to initialize various specialized policies. On our side, we are conducting more experiments to understand why agents benefit from training on well-chosen scenarios, and whether performing a kind of "curriculum learning", with progressively increasing difficulty in scenarios, may help. However, our role as organizers is to bootstrap the competition with a reasonably good agent, but leave room for improvement. Hence, we did not provide our latest and greatest agent. The final results of the challenge and post-challenge analyses will be included in this paper at revision time. The challenge platform will remain open beyond the termination of the challenge as an ever lasting benchmark, and we hope to continue organizing challenges in this series with the feed-back of participants and the power network community.

Aknowledgements: We are grateful to Alessandro Leite, Farid Najar, and Sébastien Treguer for stimulating discussions. This project is co-organized by RTE France and Université Paris-Saclay, with support of Région Ile-de-France, TAILOR EU Horizon 2020 grant 952215, ANR Chair of Artificial Intelligence HUMANIA ANR-19-CHIA-0022, and ChaLearn.

References

[1] Artelys, Armines, and Energies Demain. A 100% renewable electricity mix? analyses and optimisations. 2015.

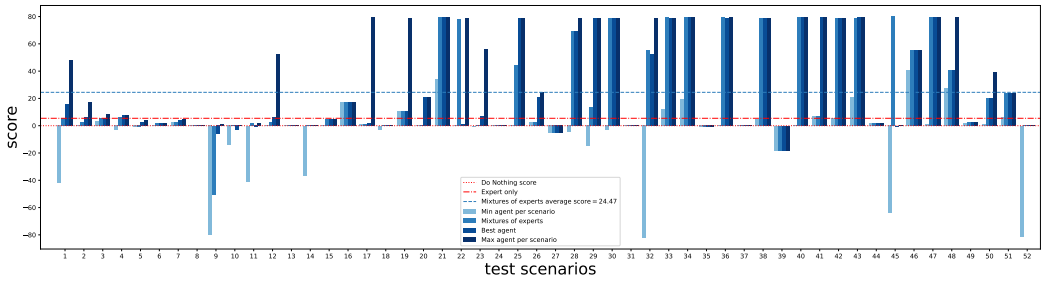

Fig. 8. Score of our "mixture of expert agents" algorithm over all validation scenarios compared with the best and worst agent in the set at each scenario and our best baseline agent.

[2] Richard Bellman. A markovian decision process. *Indiana Univ. Math. J.*, 6:679–684, 1957.

[3] Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemyslaw Debiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Christopher Hesse, Rafal Józefowicz, Scott Gray, Catherine Olsson, Jakub Pachocki, Michael Petrov, Henrique Pondé de Oliveira Pinto, Jonathan Raiman, Tim Salimans, Jeremy Schlatter, Jonas Schneider, Szymon Sidor, Ilya Sutskever, Jie Tang, Filip Wolski, and Susan Zhang. Dota 2 with large scale deep reinforcement learning. *CoRR*, abs/1912.06680, 2019.

[4] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym, 2016.

[5] B. Donnot. Grid2op- A testbed platform to model sequential decision making in power systems. . https://GitHub.com/rte-france/grid2op, 2020.

[6] Benjamin Donnot. *Deep learning methods for predicting flows in power grids : novel architectures and algorithms*. Theses, Université Paris Saclay (COmUE), February 2019.

[7] Balthazar Donon. *Deep statistical solvers & power systems applications*. Theses, Université Paris-Saclay, March 2022.

[8] D. Ernst, M. Glavic, and L. Wehenkel. Power systems stability control: reinforcement learning framework. *IEEE Transactions on Power Systems*, 19(1):427–435, 2004.

[9] Valérie Pécresse et le conseil régional d'Ile-de France. StratÉgie Énergie-climat de la rÉgion Île-de-france. 2018.

[10] RTE France. Futurs énergétiques 2050 principaux résultats. October 2021.

[11] Vijay Konda and John Tsitsiklis. Actor-critic algorithms. In S. Solla, T. Leen, and K. Müller, editors, *Advances in Neural Information Processing Systems*, volume 12. MIT Press, 1999.

[12] Tu Lan, Jiajun Duan, Bei Zhang, Di Shi, Zhiwei Wang, Ruisheng Diao, and Xiaohu Zhang. Ai-based autonomous line flow control via topology adjustment for maximizing time-series atcs, 2019.

[13] Kyunghyun Lee, Byeong-Uk Lee, Ukcheol Shin, and In So Kweon. An efficient asynchronous method for integrating evolutionary and gradient-based policy search. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[14] Yuxi Li. Deep reinforcement learning: An overview. *CoRR*, abs/1701.07274, 2017.

[15] A. Marot, N. Megel, V. Renault, and M. Jothy. ChroniX2Grid - The Extensive PowerGrid Time-serie Generator. https://github.com/BDonnot/ChroniX2Grid, 2020.

[16] Antoine Marot, Benjamin Donnot, Gabriel Dulac-Arnold, Adrian Kelly, Aïdan O'Sullivan, Jan Viebahn, Mariette Awad, Isabelle Guyon, Patrick Panciatici, and Camilo Romero. Learning to run a power network challenge: a retrospective analysis. *CoRR*, abs/2103.03104, 2021.

[17] Antoine Marot, Benjamin Donnot, Camilo Romero, Luca Veyrin-Forrer, Marvin Lerousseau, Balthazar Donon, and Isabelle Guyon. Learning to run a power network challenge for training topology controllers. *CoRR*, abs/1912.04211, 2019.

[18] Matiur Rahman Minar and Jibon Naher. Recent advances in deep learning: An overview. *CoRR*, abs/1807.08169, 2018.

[19] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning, 2013.

[20] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, Feb 2015.

[21] H.-O. Pörtner, D.C. Roberts, M. Tignor, E.S. Poloczanska, K. Mintenbeck, A. Alegría, M. Craig, S. Langsdorf, S. Löschke, V. Möller, A. Okem, and B. Rama (eds.). *IPCC, 2022: Climate Change 2022: Impacts, Adaptation, and Vulnerability. Contribution of Working Group II to the Sixth Assessment Report of the Intergovernmental Panel on Climate Change*. Cambridge University Press.

[22] Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22(268):1–8, 2021.

[23] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017.

[24] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, Timothy Lillicrap, Karen Simonyan, and Demis Hassabis. Mastering chess and shogi by self-play with a general reinforcement learning algorithm, 2017.

[25] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, Yutian Chen, Timothy Lillicrap, Fan Hui, Laurent Sifre, George van den Driessche, Thore Graepel, and Demis Hassabis. Mastering the game of go without human knowledge. *Nature*, 550:354–, October 2017.

[26] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033, 2012.

[27] Hado van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning, 2015.

[28] Deunsol Yoon, Sunghoon Hong, Byung-Jun Lee, and Kee-Eung Kim. Winning the l2rpn challenge: Power grid management via semi-markov afterstate actor-critic. In *International Conference on Learning Representations*, 2021.

[29] Zidong Zhang, Dongxia Zhang, and Robert C. Qiu. Deep reinforcement learning for power system applications: An overview. *CSEE Journal of Power and Energy Systems*, 6(1):213–225, 2020.

[30] Bo Zhou, Hongsheng Zeng, Yuecheng Liu, Kejiao Li, Fan Wang, and Hao Tian. Action set based policy optimization for safe power grid management, 2021.

[31] Bo Zhou, Hongsheng Zeng, Fan Wang, Yunxiang Li, and Hao Tian. Efficient and robust reinforcement learning with uncertainty-based value expansion. *CoRR*, abs/1912.05328, 2019.