Data Augmented Rule-based Expert System to Control a Hybrid Storage System

Ricardo J. Bessa
Centre for Power and Energy Systems (CPES)
INESC TEC
Porto, Portugal
ricardo.j.bessa@inesctec.pt

Francisco Lobo, Francisco Fernandes, Bernardo Silva
CPES & Department of Electrical and Computer Engineering
INESC TEC & Faculty of Engineering of the University of Porto
Porto, Portugal
{francisco.s.lobo; francisco.s.fernandes; bernardo.silva}@inesctec.pt

***Abstract*—Hybrid storage systems that combine high energy density and high power density technologies can enhance the flexibility and stability of microgrids and local energy communities under high renewable energy shares. This work introduces a novel approach integrating rule-based (RB) methods with evolutionary strategies (ES)-based reinforcement learning. Unlike conventional RB methods, this approach involves encoding rules in a domain-specific language and leveraging ES to evolve the symbolic model via data-driven interactions between the control agent and the environment. The results of a case study with Li-ion and redox flow batteries show that the method effectively extracted rules that minimize the energy exchanged between the community and the grid.**

***Index Terms*—Hybrid storage, control, symbolic, evolutionary strategies, reinforcement learning**

I. INTRODUCTION

Energy storage systems (ESS) will be fundamental in future power systems with high renewable energy shares, extending their significance to the microgrid or community level. Pumped hydro storage stands out as the ESS with the highest technological maturity, but its applicability is constrained by specific site conditions. As the energy storage landscape evolves, various solutions, including electrochemical (batteries, supercapacitors), electrical (superconducting magnets), mechanical (compressed air), chemical (hydrogen, ammonia), or thermal technologies, are gaining prominence. Each technology has its characteristics, which can be combined to better adapt to its final use. High energy density ESSs (HEDE) are mainly used to supply loads with slow dynamic response, while high power density ESSs (HPDE) can cope with high and fast power fluctuation. Hybrid energy storage systems (HESS) often combine HPDE and HEDE technologies [1].

In the coming years, HESS will provide services such as power quality improvement, price arbitrage, peak shaving, reserve, and lifetime extension. Control methodologies must be implemented for these use cases to optimize the power allocation among the technologies during charging and discharging. The control strategies can be divided into three groups [1]: centralized, decentralized, and distributed.

Examples of centralized approaches are the rule-based (RB) methods, generally adopted by industry due to their lower computational complexity, high interpretability to human experts, and more seamless for real-time applications [2, 3]; the fuzzy logic control approaches that generate reference power, which is decomposed into average and transient power [4]; filtration-based control, which consists on the use of a low-pass filter to split the power between the high energy (low frequency) device and high power device (high frequency) [5].

The decentralized approach is essentially based on consensus optimization or multi-agent approaches where information is exchanged between neighboring agents to achieve the global control goal via a sparse communication network [6]. Fully decentralized approaches are essentially based on the droop control concept [7]. Nevertheless, controllers based on artificial neural networks (ANNs) and reinforcement learning (RL) are becoming an alternative to complex droop-based control loops, especially when combined with domain knowledge in power system control theory [8].

Lin et al. extensively discussed the impact of communication delays on different control strategies with a competitive advantage of decentralized methods [7]. However, RB methods are also unaffected by communication delays since they are based on switching modes, but, as shown in [4], they lead to non-optimal solutions. To overcome this limitation, the main idea behind the present paper is to augment these RB expert systems by learning and improving from data and learning new tasks (objectives) without human expert intervention.

This work builds upon the foundation of evolutionary strategies (ES)-based RL as introduced by OpenAI [9], a methodology applied to train an ANN for playing different games. However, instead of having the ANN as the policy, this approach involves coding the rules in a domain-specific language, forming a symbolic model. ES is then used to refine the symbolic model via data collected from the interactions between the control agent and the environment. To our knowledge, this is the first work to propose an RB method capable of learning from data while preserving interpretability for human experts. The following case study is used to assess the methodology's effectiveness: Li-ion and redox flow batteries

The research leading to this work is being carried out as a part of the ENFIELD (*European Lighthouse to Manifest Trustworthy and Green AI*) project, European Union's Horizon Research and Innovation Programme, Grant Agreement No. 101120657. Views and opinions expressed are, however, those of the author(s) only and do not necessarily reflect those of the European Union. Neither the European Union nor the granting authority can be held responsible for them.

installed in a residential energy community with centralized photovoltaic (PV) generation, which is intended to work towards net-zero energy exchange with the main power grid.

This paper is divided into Section II, which describes the case study and ESS models; Section III presents a benchmark RB control method; Section IV describes the augmented expert system approach. The numerical results and conclusions are presented in Sections V and VI correspondingly.

II. System Description

The HESS diagram of the test setup is depicted in Fig. 1, which considers a HEDE (Li-ion battery) and HPDE (redox flow) storage units, each with its converter. These are connected to a common DC bus, where the PV panels are also connected. On the same bus, a DC/AC converter allows the connection to the residential community, which is connected to the main grid. The DC/DC converters of the HESS are centrally controlled by an Energy Management System (EMS), which takes as inputs features of the ESS, their state of charge (SOC), and the input current (i_{load}) at the given moment.

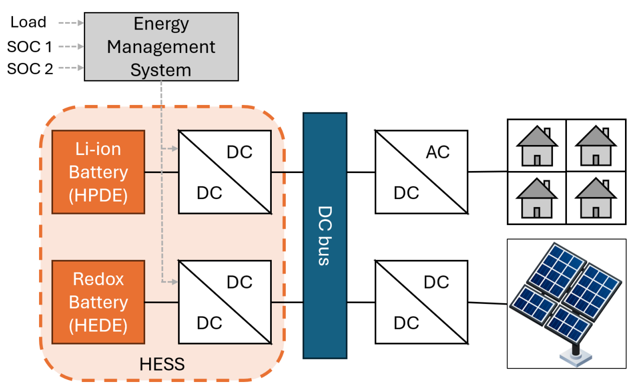

Fig. 1. Diagram of the proposed HESS structure.

A redox flow battery is a rechargeable ESS in which an electrolyte flows through an electrochemical cell from one or more tanks, where electrical and chemical energy conversion occurs. The energy scalability (or modularity) of redox flow batteries is easily achieved by increasing the volume of electrolytes in the tanks. However, the power scalability is difficult to implement since it consists of adding more permutation membranes. In this setup, a redox flow battery is used as the HEDE. It is based on the VisFlow 8 kW indoor module, with a capacity of 100 kWh. Assuming an operation at a nominal voltage of 60 V, it is calculated that the module has a capacity of 1666 Ah and a max current bidirectional of 133 A.

Li-ion is a well-established technology with widespread implementation in various functions. The connection of Li-ion batteries in series increases the capacity and voltage, while the connection in parallel allows an increase in the capacity and maximum charge/discharge current. For this setup, 20 battery modules were connected in parallel to model the HPDE. The modules used were PPCB-6030 60V 30Ah, and considering operation at 60 V, the total capacity is 600 Ah, and the maximum charge/discharge current was 400 A.

A 1-year dataset containing the hourly load profile of an aggregate of residential buildings in New Jersey, USA, was used to simulate the load connected to the DC bus [10]. The max power observed was 105 kW, a min of 13.6 kW, and an average of 41.79 kW. From the PV GIS (Photovoltaic Geographical Information System) platform [11], the production profile of a PV power plant with an installed capacity of 224 kWp and 14% efficiency was obtained for 2016 in New Jersey. This plant was oversized to produce electricity to supply the residential load and provide enough energy to accumulate in the HESS for use in the night hours.

The redox flow and the Li-ion batteries were modeled using a Coulomb Counting SOC estimator. This is a simple model based on the integration of the load current (I_{batt}) to obtain the SOC at any given moment, Eq. (1). SOC_0 represents the SOC at the beginning of the simulation, while C_{rated} is the capacity of the battery in Ah.

$$SOC(t) = SOC_0 + \frac{1}{C_{rated}} \int_0^t I_{batt} dt \tag{1}$$

Although many other variables can influence the behavior of the batteries, this model presents fast computational time, making it suitable for discovering the control method. It was considered that both storage devices had protections that would activate to prevent the SOC from going above 100% or under their maximum depth of discharge (DOD) of 20%.

A class was created in Python to model the ESS as objects that have as input the initial SOC, its capacity in Ah, and the saturation in A. It also contains a function to update the SOC based on the current in A, the time interval, and the operation mode, according to Eq. (1).

III. Benchmark Rule-Based Control Method

The benchmark RB control is based on the one described in [2, 4]. This RB is used by the EMS for controlling the distribution of load between a HEDE and HPDE. These rules are in the form of *if-else* statements that consider 8 operation modes that are described in the flowchart of Fig. 2. The rules were made with the goal of minimizing the load on the redox flow battery i_{rd} to increase its lifespan, by applying the excess load to the Li-ion battery (i_{sc}) when the demand is high. The parameters K_t and K_r are calculated with the following equations:

$$K_t = \frac{x}{x+y} \tag{2}$$

$$K_r = \frac{1-x}{2-x-y} \tag{3}$$

$$x = \alpha_{SOC} \frac{SOC_{rd} - SOC_{rd,low}}{SOC_{rd,low}} \tag{4}$$

$$y = \frac{SOC_{bat} - SOC_{bat,low}}{SOC_{bat,low}} \tag{5}$$

$$\alpha_{SOC} = \frac{\frac{SOC_{bat,high} - SOC_{bat,low}}{SOC_{bat,low}}}{\frac{SOC_{rd,high} - SOC_{rd,low}}{SOC_{rd,low}}} \tag{6}$$

where *low* and *high* are used to represent the minimum and maximum allowable SOC of each device; rd represents the Redox-flow battery, and bat the Li-ion battery; x, y and α_{SOC} are intermediate variables; K_t and K_r are used to divide the load among the devices when they are both charging and discharging.

IV. Augmented Expert System

A. ES: Core Learning Algorithm

The ES is the core learning algorithm to build a symbolic model with easily interpreted rules to optimize the HESS, which should be embedded in the EMS. ES is a population-based meta-heuristic that can be applied to achieve an optimal solution to an optimization problem [12], i.e., objective function and constraints. Inspired by natural selection, a population of individuals (solutions) is mutated for the next generation, and an evaluation and selection process allows the best-adapted individuals to be selected for the next generation. This process occurs in a cycle until a certain stop condition is reached. Ultimately, the best-performing individual is selected as the final solution to the problem.

The innovative modification in this work involves a departure from the conventional approach of seeking a global optimal solution for an optimization problem. However, instead, ES is employed to iteratively conduct symbolic learning within a predefined template (see IV-B). This iterative process aims to optimize a reward function (see IV-E), estimated through interactions between the augmented expert system and the environment.

B. Symbolic Model

1) HESS Operating Modes: Five operation modes of the HESS, inspired by [2], are considered and provide the applied current for each ESS, i_{bat} for Li-ion battery and i_{rd} for redox flow battery. Namely:

- **Mode 1**: Load divided following parameter K, representing the percentage of load for each ESS; $i_{bat} = i_{load} * K$ and $i_{rd} = i_{load} * (K - 1)$;
- **Mode 2**: All the load is directed to the Li-ion battery; $i_{bat} = i_{load}$ and $i_{rd} = 0$;
- **Mode 3**: All the load is directed to the redox flow battery; $i_{bat} = 0$ and $i_{rd} = i_{load}$;
- **Mode 4**: The load is directed to the Li-ion battery, and also current (i_{min}) is transferred from the redox flow battery to charge the Li-ion battery. Depending on the signals, it can be the opposite; $i_{bat} = i_{load} + i_{min}$ and $i_{rd} = -i_{min}$;
- **Mode 5**: It is the opposite of Mode 4, with transfer from the Li-ion battery to charge the redox flow battery (or the opposite, depending on the signal); $i_{bat} = -i_{min}$ and $i_{rd} = i_{load} + i_{min}$;

2) Templates: The presented templates transform the ES individuals structure into a Python language string that encapsulates the rules dictating the load distribution, specifically indicating the selected mode. Four distinct templates were examined, each exhibiting an increasing level of complexity. Notwithstanding their variations, all templates are constructed with one or more conditional statements, adhering to the following domain-specific language:

```
if {[Parameter] [Operator] [Threshold]}:
    mode = [Mode]
else:
    mode = [Mode]
```

where the elements inside squared brackets will evolve to optimize a reward function that was constructed to prioritize individuals with more suitable rules. The template to be selected is also a parameter of the population's members, and therefore it will also result from the learning process.

Template 1: This template considers only one conditional statement, with 2 mode options selected. Example:

```
if {SOC_bat < 0.5}:
    mode = 2
else:
    mode = 3
```

Template 2: An additional condition is added to the rule. Example:

```
if {SOC_bat < 0.3}:
    mode = 2
else:
    if {SOC_rd < 0.8}:
        mode = 3
    else:
        mode = 5
```

Template 3: A third condition is added to Template 2. Example:

```
if {SOC_bat < 0.33}:
    mode = 2
else:
    if {SOC_rd < 0.65}:
        mode = 3
    else:
        if {i_load < i_min}:
            mode = 4
        else:
            mode = 5
```

Template 4: Based on having two independent expressions of Template 3, with a previous fixed condition dividing the cases of i_{load} being positive or negative. Example:

```
if i_load > 0:
    if {SOC_bat < 0.78}:
        mode = 2
    else:
        if {SOC_rd < 0.36}:
            mode = 3
        else:
            if {i_load < i_min}:
                mode = 4
            else:
                mode = 5
else:
    if {i_load < i_min}:
        mode = 1
    else:
```

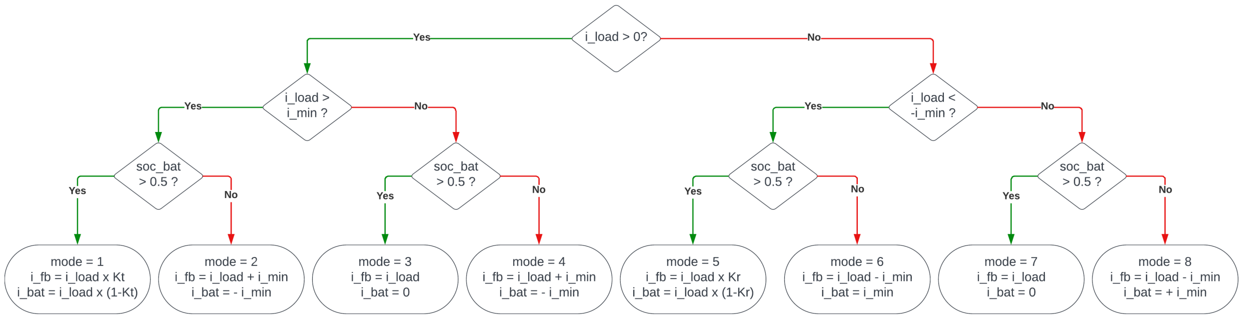

Fig. 2. Flowchart of the rule-based control strategy [2, 4].

```
if {SOC_rd < 0.8}:
    mode = 2
else:
    if {SOC_bat < 0.66}:
        mode = 5
    else:
        mode = 4
```

3) Individuals: The individuals are possible solutions to the learning problem that contain the elements to be placed on the template, thereby forming the rules. Each individual is composed of 35 positions containing the elements that can be mutated, namely:

- 8 positions related to scalar variables divided in 4 SOC thresholds (2 for Li-ion and 2 for redox flow, in which one is for positive load and the other for negative load); 2 load current thresholds (positive and negative load); and 2 parameters K to define the percentage of current attributed to each ESS (again, one for positive and the other for negative loads). All these variables are normalized to include values between 0 and 1 only. In the case of variable I_{min}, this normalization is referenced by the value of the current saturation of the HPDE.
- The following 26 positions define which variables go to each position of the used template. They define the 8 modes, the 12 parameters (SOC or i_{load} that are compared, and 6 comparison operators ($>$ or $<$) that are used.
- The final position in each individual is an integer determining the template to employ.

It is important to highlight that the decisive factor in determining the template lies in the last element of the individual. Consequently, this element defines which scalars and variables are effectively integrated into the rules.

C. ES Initialization

The program starts with the definition of 3 hyper-parameters: the number of iterations *n_it*, the population *POP*, and the mutation rate σ. Then, the second step is to define the lists with the possible discrete variables to take the 26 middle positions of the individual, which are:

```
comp_par = [['soc_bat', 'soc_bat_comp'],
    ['soc_rd', 'soc_rd_comp'],
    ['i_load', 'i_min']]
```

```
comp_par_n = [['soc_bat', 'soc_bat_comp_n'],
    ['soc_rd', 'soc_rd_comp_n'],
    ['i_load', 'i_min_n']]
log_op = ['<', '>']
modes = list(range(1, 6))
templates = list(range(1,5))
```

where *comp_par* contains 3 sub-lists with the possible parameters being compared and their threshold, and *comp_par_n* is the same but for negative loads. List *log_op* contains the considered logical operators. As for *modes* and *templates*, they contain all numbers between 1 and 5 (number of operation modes) and between 1 and 4 (number of templates), respectively.

A population of *POP* random solutions is created to start the ES. For each, 8 scalars are selected with a random function of uniform distribution between 0 and 1. Then, 4 elements are randomly sampled from *modes*, 3 from *comp_par* and from *comp_par_n*. 6 logical operators are randomly chosen from *log_op* and 1 is chosen from the list *templates*. For this case study, a population of 50 was considered.

After the initialization, the ES will enter its loop phase, which lasts until the stopping criterion is met. In this case, a limit of 1000 (*n_it*) iterations was established. The general pseudocode is presented in Algorithm 1.

Algorithm 1 ES Algorithm

Require: $n\_it, POP, \sigma, input\_load$
  $B \leftarrow$ **initialize**()
  for it **in range**(n_it) **do**
    $B \leftarrow$ **duplicate**(B, POP)
    $B \leftarrow$ **mutate**(B, POP, σ)
    $C \leftarrow$ **evaluate**($B, POP, input\_load$)
    $B, C \leftarrow$ **select**(B, POP, C)
  end for

D. Duplicate and Mutate

The first step in the cycle is duplicating the population to form the descendants. The descendants will then be subject to a mutation process to produce new solutions to the problem. This mutation process is done following the procedure in Algorithm 2. The scalar values are updated by adding small increments based on the product of the σ parameter by a randomly generated number with a Gaussian distribution of mean 0 and standard deviation 1. The remaining elements are updated as in the initialization (section IV-C). During the last 100 iterations, only the scalar values are mutated to reduce variability and increase the precision of the solutions found at that point.

Algorithm 2 Mutation Algorithm

Require: B, n_it, it, POP, σ
  for i **in range**(POP, POP $\times$ 2) **do**
    for j **in range**(8) **do**
      $new\_value \leftarrow B[i][j] + (\sigma \times \text{random.gauss}(0, 1))$
      $B[i][j] \leftarrow \max(0, \min(new\_value, 1))$
    end for
    if $it < (n\_it - 100)$ **then**
      $B \leftarrow$ **update_parameters**()
      $B \leftarrow$ **update_modes**()
      $B \leftarrow$ **update_template**()
    end if
  end for
  return B

E. Evaluate and Select

At this point, the simulations using each of the individuals are conducted. The individuals are translated into the rules used by the EMS to divide the load between the two ESS.

For each of the ESS, two lists are obtained. The first shows the current set-point as calculated by the EMS (i_cal_bat and i_cal_rd), and the second list is the current that effectively consumed/injected from the ESS when considering the protections for DOD and for saturation (i_lim_bat and i_lim_rd).

To evaluate the performance of the RB control, a reward function ($R = |R_1 + R_2|$) that exploits the difference between the current set-point calculated by the augmented expert system ($i\_cal(t)$) and the current that is effectively consumed/injected from the ESS ($i\_lim(t)$) was adopted, Eq. (7-8).

$$R_1 = \sum_{t}^{t_{max}} (|i\_cal\_bat(t)| - |i\_lim\_bat(t)|) \tag{7}$$

$$R_2 = \sum_{t}^{t_{max}} (|i\_cal\_rd(t)| - |i\_lim\_rd(t)|) \tag{8}$$

The 50% of the individuals in the population with higher R are selected to become the progenitors of the next generation.

When all iterations run, the individual with the best performance in reward is selected as the final solution for the problem.

V. Numerical Results

For this section, the data generated with the system from Section II was partitioned into the following: from every four consecutive weeks, the last week was designated to the testing dataset, and the preceding weeks were assigned to the training dataset. This approach yielded 39 weeks dedicated to training and 13 weeks to testing, effectively covering the seasonality effect. The augmented RB expert system from Section IV was applied to the training set in parallel for 12 different random seeds, requiring a computational time of 22 hours. All seeds converged to solutions using template 4, the template with more conditions that better partition the search space. The best RB model found by the ES-based algorithm (i.e., with the lower reward per week) both for the training and testing was:

```
1  K=0.543 #scalar used in mode 1
2
3  if i_load > 0:
4          if soc_rd < 0.242:
5          mode = 2
6      else:
7          if i_load < 140.8:
8              mode = 3
9          else:
10             if soc_bat < 0.05:
11                 mode = 4
12             else:
13                 mode = 1
14 else:
15     if soc_rd < 0.061:
16         mode = 1
17     else:
18         if soc_bat > 0:
19             mode = 4
20         else:
21             if i_load > -133.2:
22                 mode = 5
23             else:
24                 mode = 3
```

Upon examining the learned rule, it is noticeable that the first condition for negative load (line 15) is impossible for negative load due to the set DOD of 20%. Likewise, the second condition (line 18) is always true since the Li-ion battery SOC is always above the DOD of 20%. Consequently, these rules could be simplified so that whenever the load is negative, the mode is 4. This is how the algorithm was found to automatically create a simpler template, eliminating the need for extraneous modes to achieve convergence.

Fig. 3 depicts the reward as a function of the number of iterations throughout the training process, starting from higher values until converging to a final solution. Notably, the influence of exclusively evolving the scalar parameters in the template becomes evident at iteration 900, effectively narrowing the search space and expediting the discovery of improved solutions.

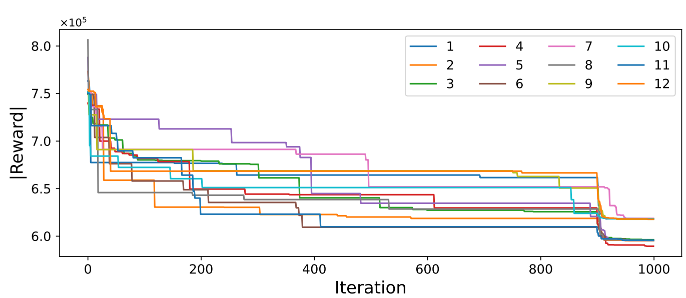

Fig. 3. Reward vs iterations during the training phase for each seed.

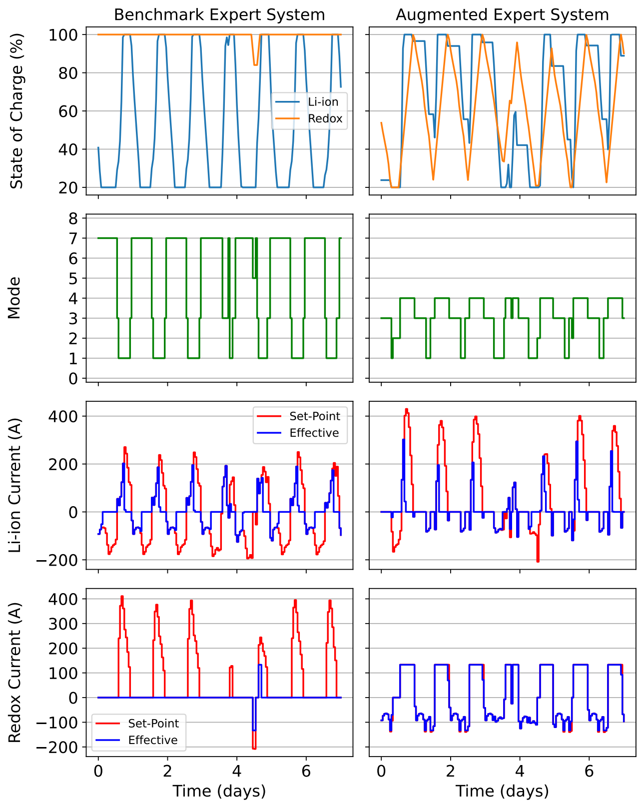

Fig. 4. Evolution of SOC, mode, and currents for the two methods.

Table I presents the total reward achieved in both the training and test sets for the augmented expert system and the benchmark RB method [2]. The reward was reduced approximately 3 fold for the training and 2 fold for the testing compared to the benchmark. This implies that the derived rules with the proposed approach exhibit better adaptability to the specific objective of the case study. It is essential to emphasize that the existing RB system was successfully evolved through the proposed methodology to learn a new objective or task based on the available data.

TABLE I
TOTAL REWARD FOR TRAINING AND TESTING SETS

	Train	Test
Benchmark Expert System [2]	45868	27066
Augmented Expert System	15115	14819

The SOC of the devices, the modes, the current set-point and the effective current obtained with each ruleset for the 4^{th} week of August 2016 are depicted in Fig. 4. The curve labelled "Set-Point" corresponds to the current values as determined by the rules, while the curve labelled "Effective" (blue) corresponds to the current values that were effectively supplied after considering saturation and availability of stored energy. It can be noted how the new rules attempt to better distribute the load between both ESS compared to the original rules, which mainly rely on the Li-ion battery to supply the load, with the redox flow battery supplying the Li-ion battery in some situations.

VI. CONCLUSIONS

The effectiveness of the proposed methodology in developing a symbolic model for RB control in a HESS within a community, aiming to minimize transactions with the main grid, has been successfully demonstrated. The ES as a base for reinforcement learning allowed for optimizing the rules to the available data and the reward function, showing good performance for the training and testing set. This methodology serves as a promising initial step for extracting highly interpretable rules. It can be a powerful decision-making system that could be applied across diverse domains where comprehending and explaining the reasoning behind algorithmic decisions is paramount.

Future work consists of extending this framework to automatically design decentralized control schemes and handle multiple HESS with different capacities. The control of the DC bus voltage could also be considered in future work.

REFERENCES

[1] X. Lin and R. Zamora, "Controls of hybrid energy storage systems in microgrids: Critical review, case study and future trends," *J. of Ener. Stor.*, vol. 47, p. 103884, 2022.

[2] K. Gokce and A. Ozdemir, "A rule based power split strategy for battery/ultracapacitor energy storage systems in hybrid electric vehicles," *Int. J. of Electrochemical Sci.*, vol. 11, pp. 1228–1246, 2016.

[3] S. Teleke, M. E. Baran, S. Bhattacharya, and A. Q. Huang, "Rule-based control of battery energy storage for dispatching intermittent renewable sources," *IEEE Trans. on Sust. Ener.*, vol. 1, pp. 117–124, Oct. 2010.

[4] M. H. Ali, D. Slaifstein, F. M. Ibanez, C. Zugschwert, and M. Pugach, "Power management strategies for vanadium redox flow battery and supercapacitors in hybrid energy storage systems," in *IEEE PES ISGT Europe 2022*, (Novi Sad, Serbia), 2022.

[5] S. K. Abadi and A. Bidram, "A distributed rule-based power management strategy in a photovoltaic/hybrid energy storage based on an active compensation filtering technique," *IET Ren. Power Gen.*, vol. 15, pp. 3688–3703, Nov. 2021.

[6] R. Olfati-Saber, J. A. Fax, and R. M. Murray, "Consensus and cooperation in networked multi-agent systems," *Proc. of the IEEE*, vol. 95, pp. 215–233, Jan. 2007.

[7] X. Lin, R. Zamora, and C. A. Baguley, "A coordinated droop controls and power management scheme for hybrid energy storage systems in DC microgrids," in *AUPEC 2021*, (Perth, Australia), 2021.

[8] J. Duan, Z. Yi, D. Shi, C. Lin, X. Lu, and Z. Wang, "Reinforcement-learning-based optimal control of hybrid energy storage systems in hybrid AC-DC microgrids," *IEEE Trans. on Ind. Infor.*, vol. 15, pp. 5355–5364, Sept. 2019.

[9] T. Salimans, J. Ho, X. Chen, and I. Sutskever, "Evolution strategies as a scalable alternative to reinforcement learning," *ArXiv*, vol. abs/1703.03864, 2017.

[10] F. Angizeh, A. Ghofrani, and M. A. Jafari, "Dataset on hourly load profiles for a set of 24 facilities from industrial, commercial, and residential end-use sectors," in *Mendeley Data, V1*, Aug. 2020.

[11] T. Huld, I. P. Pascua, and A. G. Amillo, "PVGIS 5: Internet tools for the assessment of solar resource and photovoltaic solar systems," Tech. Rep. JRC107813, European Commission, 2017.

[12] H.-G. Beyer and H.-P. Schwefel, "Evolution strategies a comprehensive introduction," *Natural Comp.*, vol. 1, pp. 3–52, Mar. 2002.