Imperial College London

Department of Earth Science and Engineering

MSc in Applied Computational Science and Engineering

Independent Research Project

Reinforcement Learning Control of Hybrid Renewable Energy System with Hybrid Storage

by

Son Hoang Thanh

marek.hng@gmail.com
GitHub login: sonhoangthanh

Supervisors:

Dr. Matt Piggot
Industrial Supervisor Zoe Goss

January 2, 2022

GitHub repository:
https://github.com/sonhoangthanh/msc-personal-project

Abstract

This study aims to the use deep reinforcement learning for optimal power control of hybrid renewable energy system with hybrid storage. The model environment for theoretical hybrid plant of 25MW solar-PV plant, 80MW wind farm, 500MWh battery and hydrogen storage and 20MW diesel generator located at the Isle of Man was offline-trained using deep Q-learning method to maximize the predicted revenue return. The results were evaluated against the baseline rule-based and random-action controllers. Early results show a successful stabilization of the training process using modified double Q-learning algorithm, and significantly better performance of the RL-controller in comparison to the baselines. Limitations of the method are outlined, with proposal of future extensions and work presented at the end.
Keywords: hybrid renewable, hybrid storage, power control, reinforcement learning, deep Q-learning

1 Introduction

Global economic growth and continued development of every world economy can only be met by satisfying increasing large energy demands. Current trends in our reliance on the use of fossil fuels can not be sustained, as these resources are finite, and due to their ecological and environmental impact they cause. Renewable energy sources are said to become major factor in achieving long-lasting, low-carbon energy supply in the quest to mitigate the effects of climate change [4]. Among them, wind and solar photo-voltaic (PV) are said to be the most promising, with their abundance, lesser reliance on geographical location and conditions, which contributes to its favoured development and rapid improvements in their technology in the past decade. Furthermore, the introduction of renewable sources may provide an opportunity to provide high quality energy to isolated locations and micro-grids [6]. Nevertheless, future integration of renewable energy into existing power grids and supply chains would heavily depend on overcoming a great amount of challenges. Renewable energy sources are mostly, inherently unpredictable, intermittent, and vary significantly in time and between seasons for a given location. Those aspects lead to significant difficulties in frequency, quality and appropriate response control, which often results in curtailment, lower profitability and efficiency [17, 1].

The hybridization, that is the combination of several renewable energy sources which may complement each other, such as wind and solar, can help resolve these issues, especially when combined with a suitable energy storage system (ESS) and traditional combustion generator, by providing higher efficiencies, better performance and greater stability. [12]. However, such hybrid systems showcase high-complexity, as they require a smooth operation and coordination of multiple differently-characterized subsystems, and signify the need for robust power management applied over short- and long-term periods. This report explores the use of deep reinforcement learning as the potential solution to optimal energy management and power control of hybrid renewable energy plants with hybrid storage to generate maximum profitability for the power producers.

The overview of the literature for reinforcement learning in renewable and hybrid systems and the basic theoretical background is outlined in sections 2 and 3. The study problem description is explained in section 4 and methodology is presented in section 5. Testing and verification strategy is depicted in section 6 with code metadata and implementation showcased in section 7. Finally, results are reviewed in section 8 and discussed in section 9.

2 Literature Review

Reinforcement Learning (RL) is attracting considerable attention due to its human-like capabilities in control and decision making [11], in a model-free way [14]. In summary, RL algorithms assumes the training of the model/agent by incorporating decisions based on a reward for a given action on specific environment or state (Fig. 1). The agent is trained to maximize the reward specific value function over pre-defied time period. RL-related methods can be divided into 3 groups: value-based methods, policy-gradient based (also sometimes called action-based) methods and actor-critic methods [14]. Additionally, RL can be combined with artificial neural networks to provide deep reinforcement learning methods. Some RL techniques were proven to work well when integrating energy storage solution with hybrid electric vehicles [10], and furthermore, showcased 10-20% improvements compared to other alternative methods using rule-based, fuzzy-logic, heuristic models etc [14, 20, 13, 2, 15]. In particular deep reinforcement learning methods were demonstrated to have good results compared to traditional methods [21, 20, 13].

There have been various optimization metrics used by different RL studies, with large work done with Maximal Power Point Tracking to extract maximal power from wind and solar [15], evaluating loss of power supply [13, 2], or maximizing the expected revenue [21]. In contrast to homogeneous RE power generation, not as much research has been carried out in the realm of optimization and power management of hybrid systems, in particular solar and wind combined with hybrid storage solutions. Some studies have only evaluated the use of RL models for renewable systems with ESS from single renewable source [21, 2], or have only considered only single type of ESS e.g. battery storage only. There is not much literature which deals with RL control management of HRES with hybrid ESS, neither is there much literature which combines economic profitability models from power generation with optimal operation and maintenance [16].

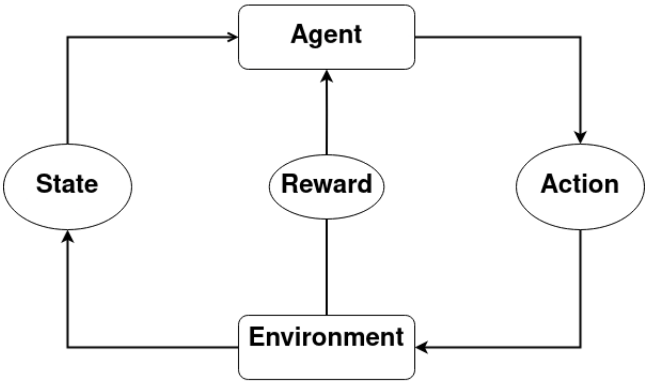

Figure 1: Diagram of Reinforcement Learning.

3 Background Theory

3.1 Markov Decision Process and Bellman's Theory

In reinforcement learning, the agent takes actions and receives an immediate reward in the dynamically evolving environment as show in Figure 1. The aim of the reinforcement learning is to control the behaviour of such dynamical system by taking the most appropriate actions to achieve a certain goal. An RL problem can be modelled as a Markov Decision Process (MDP), in both discrete and continuous state and action spaces, with a set of states $s_t \in S$ and set of a actions $a_t \in A$ for each state at a given timestep. The objective of the reinforcement learning is to obtain a optimal policy $\pi^*(a_t|s_t)$ which maximizes the total expected reward [9]. To reach that objective, methods of dynamical programming and estimation of value functions can be used. A value function is a map from state space S to domain of values V, where the value can be interpreted as a prediction of the expected, cumulative, discounted future rewards for a given state s_t. For an optimal policy $\pi^*(s_t, a_t)$ under MPD, the value function follows the Bellman's relation

$$V^{\pi^*}(s_t) = r_{t+1} + \gamma V^{\pi^*}(s_{t+1}) \tag{1}$$

, where $V^{\pi^*}(s_t)$ and $V^{\pi^*}(s_{t+1})$ is the value function at state s_t and s_{t+1}, the r_{t+1} is the reward received by transitioning from state s_t to s_{t+1} and γ is the discount factor which determines the weighting between the current reward and the future ones. Given the knowledge of the all the state values, an optimal policy can be extracted using look-up table method, value iteration and temporal difference learning.

3.2 Q-Learning

Nevertheless, calculating state values becomes quickly unfeasible with increasing number of states and actions. As a result, the Q-value approach was proposed, which evaluated the state-action function called Q-function [19]. For an optimal policy in MDP, the Q-function satisfies similar condition to the value function

$$Q^{\pi^*}(s_t, a_t) = r_{t+1} + \gamma Q^{\pi^*}(s_{t+1}, a_{t+1}). \tag{2}$$

The Q-value can be interpreted as the expected discounted reward for executing action a_t at state s_t and following the policy π thereafter. This allows the agent to learn to act optimally by experiencing the consequences of their actions, without estimating a value function over the whole domain. Such a method has been shown to iteratively converge towards the optimal Q-value [19]. However, this approach still requires discretised state and action space, which may be undesirable as many problem in the real world operate in the continuous domains. The well known solution is to use artificial neural neural network as Q-function approximator, and adjusting its weights using stochastic gradient descent would result in near optimal policy. A popular example would be the DQN algorithm first proposed in 2015 [11] with the update rule following

$$\begin{aligned} Q_t(s_t, a_t) &\leftarrow Q_t(s_t, a_t) \\ +\alpha &\left[r_t + \gamma \max_{a_t} Q_t(s_{t+1}, a_{t+1}) - Q_t(s_t, a_t) \right] \end{aligned} \tag{3}$$

, where the $\max_{a_t} Q_t(s_{t+1}, a_{t+1})$ is the maximum Q-value for state s_{t+1} given the action a_t and α is the learning rate factor.

4 Problem Statement

The project focuses on developing a deep reinforcement learning (DRL) model for power control of hybrid renewable energy system with hybrid storage for the isolated micro-grid. The RL model agent would take input simulated power generation data from renewable energy components, the demanded load information, as well as the energy states of both battery and hydrogen storage, and select the most optimal action to achieve the highest cumulative revenue. The performance of the model is then analysed and compared to the example rule-based and randomised controllers to evaluate the possible benefits and drawback of this method.

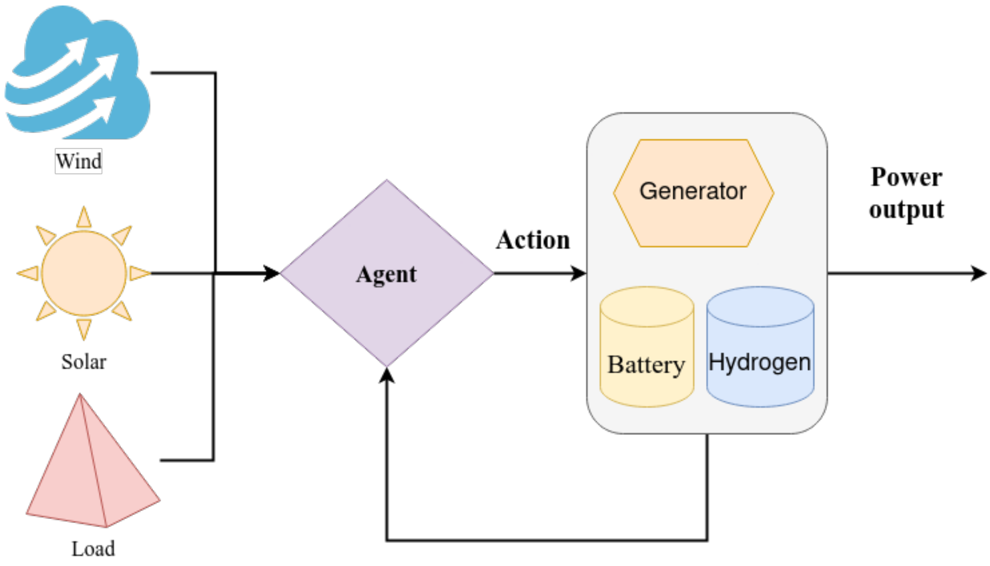

Figure 2: Diagram of the Hybrid Renewable System with Hybrid storage.

5 Methodology

5.1 Environment

The simplified environment of the hybrid renewable energy system (HRES) consists of solar power generator, wind power generator, demand load, 2 energy storage system (ESS) modules including battery and hydrogen storage, and a diesel power generator. The RL agent controls the actions of charge/discharge scheduling of each individual ESS module and the power output of the diesel generator. The diagram of the hybrid renewable system can be seen in the Figure 2.

Unlike in traditional RL, where the agent has the ability to collect new state transitions as it interacts with the environment, the approach used in this project presents a data-driven, offline variation of RL, where the agent tries to extract the optimal policy from the dataset and is given the ability to interact with two of the state parameters, that is the energy states of the ESS modules.

5.2 Data Source and Pre-Processing

The chosen site of the model training was decided to be the Isle of Man located in neighbourhood of United Kingdom. The input data used in the reinforcement learning consisted of simulated solar and wind power generation datasets from 2019 with hourly resolution. Due to lack of available data for electricity demand at Isle of Man, the demanded load was taken to be the the demand data for the neighbouring United Kingdom scaled to the population of the Isle of Man with 5 minute resolution, which was then also transformed into hourly dataset. The training data consists of the extended 2018 power generation and load datasets, augmented with random noise. The evaluation dataset consists of unaugumented 2018 power generation and 2019 demanded load data. For details, please refer to the documentation.

The optimal sizings of each HRES component were obtained with inference from popular renewable energy optimization software called HOMER. The components of HRES chosen for this model are: solar power generation of 25MW capacity, wind power generation of 80MW capacity, battery and hydrogen storage capacity of 500MWh and nominal power output of 20MW and diesel power generator with the output power of 20MW. The model assumes that an inverter used to transform AC to DC power and vice versa have the efficiency of 1 and thus is dismissed. The charge and discharge efficiency of the battery and hydrogen storage is assumed to be 0.95 and 0.7 respectively. Other forms of power losses in the system are assumed to be negligible.

5.3 Reinforcement Learning Model

The RL model consists of two parts, the training module and the application module. In the training module, the agent learns on the training dataset, whereas the application module loads the pre-trained model and evaluates its performance on the evaluation set. The learning process can be viewed in the Figure 4. Training consists of action selection using stochastic ϵ-greedy policy, where ϵ denotes the probability ratio between the exploration versus exploitation of the current policy. The closer the ϵ is to 1, the more exploratory the agent becomes. If agent were to choose exploration, it randomly samples from the pool of 18 available actions. If agent is greedy, it calculates the the Q values of each action under current state and performs the action with the largest Q-value.

Based on the action selected, the total supply power and reward are then calculated and pushed into the memory replay buffer and the process repeats. After a predefined amount of steps, the learning ensues with mini-batch sampling from the memory, calculation of the temporal difference error and performing the gradient descent. Finally both the policy and target networks are updated. The training carries on until the end of the user-specified number of episodes. Afterwards, the model can be evaluated using the application module, where the stochastic ϵ-greedy agent is replaced with the greedy neural-network agent. The agent with well-trained parameters would then make the decisions by taking the action corresponding to the highest Q-value based on its networks forward propagation of the input state. Lastly, the performance of the RL-agent can be assessed in comparison to other types of controllers.

5.3.1 Observation Space

At any given time, the RL environement consists of current solar and wind simulated power, the demanded load power, as well as the energy states of battery and hydrogen storage modules. The electric energy stored in both storage modules E_t is constrained to:

$$0 \leq E_t \leq E_{capacity} \tag{4}$$

and the energy stored in storage changes with time according to

$$\begin{aligned} E_{t+1} = E_t + (&U^{charge} P^{charge} \nu^{charge} \\ &+ U^{discharge} P^{discharge} \nu^{discharge}) \Delta t \end{aligned} \tag{5}$$

, where P^{charge}, $P^{discharge}$ are charge and discharge power of ESS modules, ν^{charge} and $\nu^{discharge}$ are charge and discharge efficiencies of the storage modules, and U^{charge} and $U^{discharge}$ are binary actions of the controller such that:

$$\begin{gathered} U^{charge} + U^{discharge} = 1 \\ \iff \begin{cases} U^{charge} = 1, U^{discharge} = 0 \\ U^{charge} = 0, U^{discharge} = 1 \end{cases} \end{gathered} \tag{6}$$

Each state s_t at a given timestep is in the form of

$$s_t = \begin{bmatrix} P_{solar,t} \\ P_{wind,t} \\ L_t \\ E_{B,t} \\ E_{H,t} \end{bmatrix} \tag{7}$$

, where P_{solar} and P_{wind} are the simulated solar-PV and wind output power, the L_t is the demanded load power, and $E_{B,t}$ and $E_{H,t}$ are energy states of the battery and hydrogen storage.

5.3.2 Action Space

The action space consists of charge / discharge of both ESS modules with defined power output, and usage of power generator. The actions were encoded into the form of:

$$a_t = \begin{bmatrix} a_{B,t} \\ a_{H,t} \\ a_{G,t} \end{bmatrix}$$

, where $a_{B,t}$ and $a_{H,t}$ are battery storage and hydrogen storage actions encoded in form of output power P_B an P_H, which were discretised into sets of

$$P_B = [P_0, ..., P_M], P_H = [P_0, ..., P_N] \tag{8}$$

The charge/discharge power was set to 20 MW, therefore the available actions for both battery and hydrogen modules were $[-20MW, 0, 20MW]$, where negative value and positive values indicated charging and discharging respectively. The $a_{G,t}$ specifies the power state of the generator, with 0 meaning off and 1 on. Thus the action space of this environment consists of 18 actions in total, for example $[20MW, 20MW, 1]$ describes the action of charging of both ESS modules and the usage of diesel generator.

5.3.3 Artificial Neural Network

To account for very large amount of possible input states, an artificial neural network (ANN) is used as a non-linear function approximator. The diagram of the ANN can be seen in the Figure 3. The size of input layer of the neural networks are the size of the observation space of the HRES and the size of the output layer has the size of the action space. The neural network also consists of 2 hidden layers with the size of 18 and 90 neurons. The neural network is used by the agent to calculate each action's expected Q-value. For each input, the action taken was the maximum Q-value obtained using the forward propagation of the ANN.

5.3.4 Reward Distribution

Reward shaping is at the center of reinforcement learning, as it defines the signals sent to the agent from the environment from which it attempts to learn the most optimal policy. The goal in this project was to define the reward system which encourages positive behaviour but does not overly restrict the actions of the agent, which may hinder generalization. The rewards should be shaped in the way such that the agent is able to distinguish the correct combination of the actions from the rest and perform the actions which lead to maximum cumulative revenue over an infinite time horizon.

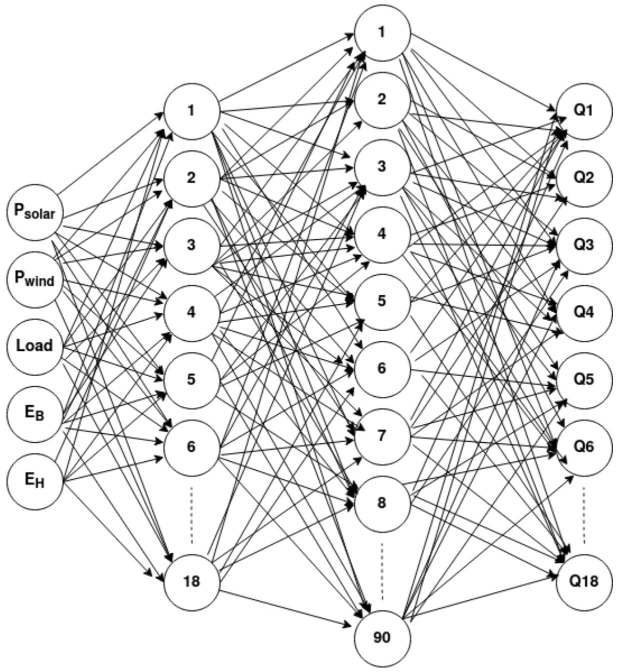

Figure 3: Deep Q-Network with 5 inputs, 2 hidden layers of length 18 and 90 neurons and output layer of Q-values corresponding to 18 actions. Activation function is ReLU.

To achieve this, financial revenue model was developed to provide the predicted revenue reward for a given state and action performed by the agent. At each time step, the expected revenue returned follows:

$$\begin{aligned} R_t = \mu_{electricity} P_{total,t} - \mu_{deficit} D_t \\ -\mu_{surplus} S_t - nC_{ESS} - \mu_G P_{G,t} - C_{LPS} \end{aligned} \tag{9}$$

where R_t is the revenue returned at each time step, $P_{total,t}$ is the total supplied power, D_t is the deficit power, S_t is the surplus power, n is the number of ESS modules used and $P_{G,t}$ is the diesel power generation. The $\mu_{electricity}$, $\mu_{deficit}$, $\mu_{surplus}$ and μ_G are the price of the demanded electricity, the power deficit penalty, power surplus penalty and the cost of diesel generator, all measured in British Pounds per kWh. The agent is also penalized for not meeting the demanded load (loss of power supply) with a fix cost of C_{LPS} and is charged a fixed maintenance cost C_{ESS} per ESS module in use. The total supplied power is restricted to the load such that

$$0 \leq P_{total,t} \leq L_t \tag{10}$$

where the L_t is the demanded load power at timestep t, and $P_{deficit} \neq 0 \implies P_{surplus} = 0$ and vice versa. The prices, costs and penalties used in this project are presented in Table 1.

Parameter	Value
$\mu_{electricity}$	0.169 £/kWh
$\mu_{deficit}$	0.2 £/kWh
$\mu_{surplus}$	0.1 £/kWh
μ_G	0.2 £/kWh
C_{ESS}	100 £
C_{LPS}	1000 £

Table 1: The prices and costs used in the financial model to estimate the returned revenue for a given

5.3.5 Double-Q-Learning

Traditional Q-learning and DQN model training is prone to overestimation of action values, which creates the upward bias [18]. This is due to the use of the *argmax* function, and also due to it being used not only to choose the value, but also in its evaluation. To overcome this problem, the use of two similarly structured neural networks to make the current state estimate of Q-value and the optimal target estimate of the Q-value is used to decouple the action selection from the evaluation. This has shown to increase the training stability and decreasing the bias, resulting in better performance [18]. The optimization loss value would be the temporal difference error between the current estimate of the Q-value for the current state according to estimation policy network and the estimate of target Q-value of the next state given the current policy calculated using the target network. The update rule is in the form of

$$\begin{aligned} TD_{error} = r_t + \gamma \max_{a_t} Q_t^{target}(s_{t+1}, a_{t+1}) - \\ Q_t^{estimate}(s_t, a_t) \end{aligned} \tag{11}$$

, where the $Q_t^{estimate}(s_t, a_t)$ and $Q_t^{target}(s_{t+1}, a_{t+1}$ are the Q values of the policy and target networks respectively.

5.3.6 Memory Buffer and Batch Learning

The experience replay memory is used in the training of the Q-networks. It stored a set of recent state transitions, actions used and the immediate rewards that the agent experienced which could then be used for future learning. The experiences are then randomly sampled as mini-batches for batch-learning. The usage of replay buffer has been shown to accelerate learning due to temporal decorrelation of the experiences. [11]

5.3.7 Regularization and Soft Update

As offline training of DRL model has similarities to that of the supervised-learning, great care has to be taken to decreasing variance towards the training data. To help

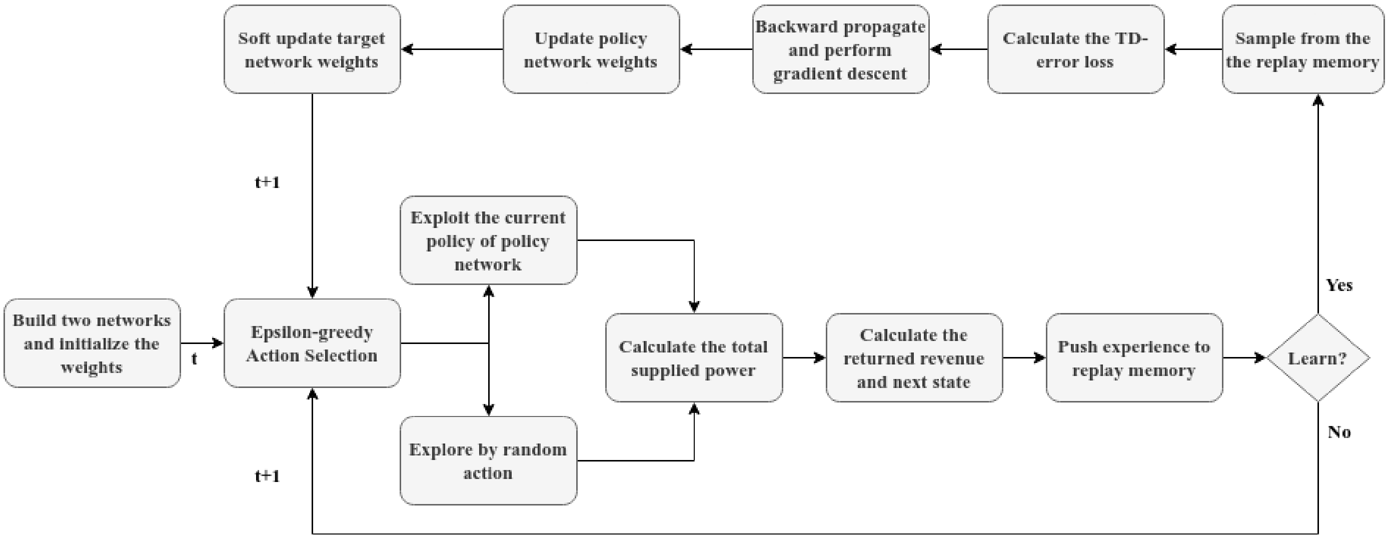
Soft update target network weights
Update policy network weights
Backward propagate and perform gradient descent
Calculate the TD-error loss
Sample from the the replay memory
t+1
Exploit the current policy of policy network
Yes
Build two networks and initialize the weights
t
Epsilon-greedy Action Selection
Calculate the total supplied power
Calculate the returned revenue and next state
Push experience to replay memory
Learn?
Explore by random action
No
t+1

Figure 4: Flowchart of DRL training process.

reduce over-fitting, the use of L-2 regularisation has been made. This regularisation add the the loss function the additional weight decay factor term $\lambda \sum_{i=1}^{N} \theta_i^2$, where θ_i are the weights of the network.

Additionally, the the soft update of the target network in relation to policy network has been employed, in contrast to the traditional DQN, where the parameters of the target network are periodically synchronized with the policy network. It has the form of:

$$\theta_{target} = \tau\theta_{policy} + (1-\tau)\theta_{target} \tag{12}$$

, where τ is the soft-update parameterization factor between policy and target networks parameters. This approach was proven to reduce the bias in the early stages of the training and was shown to perform well in the noisy environments [5].

5.4 Hyperparameter Tuning

To achieve an optimal policy during training, the hyperparameters of the model needed to be tuned. There were several hyperparameters to consider: learning rate, mini-batch size, memory size, discount rate (gamma), learning frequency, epsilon decay rate (epsilon decay), soft update rate (tau). Trial and error of various combinations of hyperparameters was used to determine the general trend. To limit the parameter search space, only learning rate, soft update rate and learning frequency were tuned with other held constant. The most optimal configuration was then used to train the DRL model. For details on tuning results, refer to the documentation in the gihub repository.

6 Testing and Verification

The components of the model, such as data loading and processing, environment components, and agent action selection and learning were unit tested to ensure correct operation and behaviour. The neural network layout was determined using iterative method, through manual pruning of initially three-layered configuration.

The model performance was evaluated on different load and power generation data. The performance of the RL controller was compared to two **baselines**: the random-action and rule-based controllers. For random controller, the action taken at anytime was randomly sampled from the action space. On the other hand, rule-based controller follows a set of rules in order to determine the chosen action. The rule-based controller uses only the nominal power outputs of the ESS modules as the predicted value to make its predictions. It has the form of:

- Usage of both storage modules is dependent on the observed surplus and deficit of the hybrid power generation in comparison to the load. If there is power surplus from hybrid renewable source, the action is to charge and if there is a deficit, the action is to discharge.
- The charge/discharge of the battery storage is prioritised.
- If the charge/discharge of a battery module can not account for the total surplus/deficit, the hydrogen storage is utilised as well.
- If both modules can not sustain the load during power deficit, the diesel generator is used as the extra source of power.

7 Code Metadata and Implementation

This project was build under Linux environment with Visual Studio Code using Python programming language. The deep learning framework used to create and implement deep reinforcement learning was Pytorch. The choice of Pytorch was made due to large collection of methods and procedures available for creation of custom neural networks and abundance of documentation and examples. Additionally, numpy and pandas libraries were used for data and array processing, and matplotlib was used for data visualization. For iterative learning, itertools and random library were used to obtain efficient iterables and random number generation. Pytest was used for unit testing of the reinforcement learning model components.

The python scripts were placed in separate directories and the repository was model in form of a python package, which allowed for modularity and ease of importing of different functions across the whole project. The structure of the classes and methods developed for this project, which rely on extensive arguments and keywords without hard-coded parameters, allows for great customization and exploration of the hyperparameters and options, and furthermore, leaves room for future extension and added functionality.

For accelerated learning, the use of CUDA enabled GPU was supported through Pytorch. The acceleration could be achieved with a local machine using the provided training and evaluation python scripts or on the cloud using attached Jupyter Notebook. The results of this project were obtained with help of GPU-accelerated environment on Google Colab platform. Final results were trained with T4 GPU and were placed in the results directory. On average, the model training of 50 episodes spanned about 8 minutes. The generated figures were placed in fig directory. The unit test were placed in the tests directory.

For further details on how to download data files, different options of the environment and the agent, as well as the usage of training, evaluation and tuning methods, please refer to the README file and see the attached Jupyter Notebook.

8 Results

Hyperparameter	Value
Activation function	ReLU
Optimizer	Adam
Batch size	50
Memory size	5000
Learning rate	1e-3
Soft update rate	1e-3
Learning frequency	20
Regularization factor	1e-4

Table 2: List of final hyperparameters used for the model training.

The list of hyperparameters used in model training can be seen in the Table 2. The RL agent was trained with Adam optimizer due to its proven performance in stochastic function estimators such as neural networks [7], especially showcased when used in reinforcement learning problems. The agent was trained for 80 episodes with the change in total rewards over episodes shown in Figure 5. As can be observed, the total rewards initially very sharply increases from firstly negative values until around revenue return of 50 million at the 20^{th} episode, but then stagnates and seemingly stabilizes. After around 40 episodes, the total reward only marginally increases with more training. Further learning beyond 80 episodes resulted in insignificant change in total reward. This points to the possibility that the controller may have found the optimal policy for the training environment or have very closely approached it.

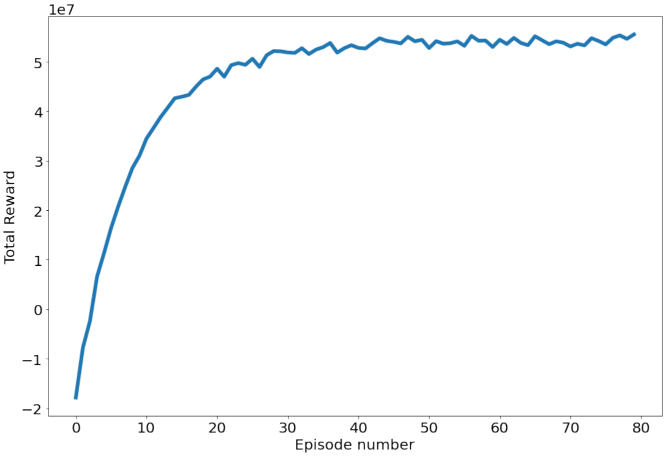

Figure 5: Evolution of total rewards per episode during 80 episode model training.

The trained model was then applied on the evaluation set with total power output versus load shown in Figure 6. We can observe that the total output of the system varies significantly, with minimum of around 10MW up to around 90MW. In Figure 6, we can additionally see that although the system seems chaotic, at the majority of points in time the controller provides supply power near or above the the demanded load power. We can see that the RL-controller attempts to minimize the loss of power supply and output the supply of power close to the load, but this notion is not absolute. In contrast, it often outputs much more power than the required load needs, especially in the summer when the load is lower. This behaviour can be attributed to the formulation of the reward function used for model training and to greater

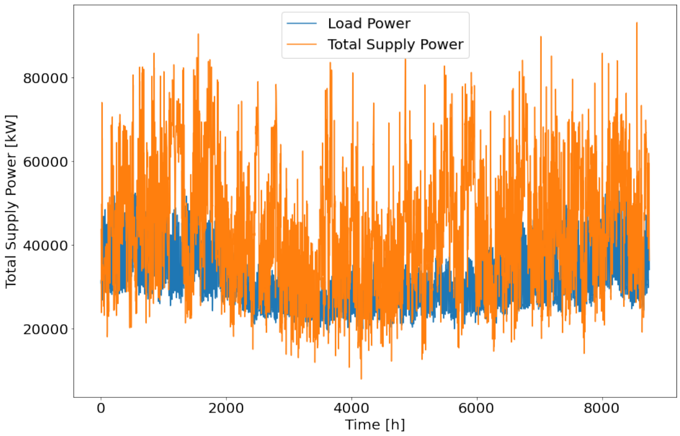

Figure 6: Total supplied power vs load for RL model application on the evaluation dataset.

supply of solar renewable power during summertime. The reward function greatly penalizes the loss of power supply with both static penalty and the cost proportional to the amount in deficit, whereas the surplus penalty is significantly smaller. Therefore the RL-agent could be more inclined to output more energy that the load needs as opposed to less to reduce the amount penalized and maximize the future rewards.

Performance of the RL-agent was subsequently compared to the two baselines, with the temporal reward distribution depicted in Figure 7 and both cumulative rewards and average rewards summarised in Table 3. In the Figure 7, we can see that both RL and rule-base agents consistently return positive revenue value during operation, with both receiving a maximum of around 7500 £ . The RL model displays less overall variance in the total revenue returned with only sparse negative-valued outliers. Contrarily, the random-action controller performs visibly worse, with majority of operation returning revenue between 5000£ and -5000£. Moreover, there seems to be a good proportion of times when the random action controller returns very large negative revenue value of up to around -20000£. All three controllers demonstrate a visible decrese in revenue returned between 3000 and 6000 hour points, which implies that all of them have difficulty in maximizing the revenue during summertime. This is probably due to high output of the hybrid renewable energy, especially solar-PV power. As a result, all three types of agents are not able to fully off-load the excess power and are appropriately penalised.

Controller	Total Reward[£]	Average Reward[£/h]
RL	2.92e7	3329.146
Rule-based	2.16e7	2465.991
Random	-1.16e7	-1326.508

Table 3: Total and average rewards comparison for RL, rule-based and random controllers for evaluation set.

The above ranking in terms of yearly performance is additionally supplemented by the results shown in Table 3. The control of the HRES presented in this project using RL-agent obtained the highest yearly cumulative revenue reward of around 29 million pounds with average revenue of 3329.146 £ per hour, followed by the rule-based controller which received around 21 million pounds with average hourly revenue return of 2465.991£. Operation which followed a random policy returned the negative loss of revenue of around -11 million £ and average loss of -1326.508£ per hour.

All the above seemingly suggests that RL approach to control of the system presented in this project is superior to the baseline controllers. The offline training of such a controller provides a relatively cheap, in terms of computation, solution to optimal control of hybrid power plant outlined in this report.

9 Discussion

As mentioned in the introduction, the area of reinforcement learning is a very active subject of research and its applications in control of in hybrid renewable energy systems, especially with multiple storage solutions, are still yet unexplored. The findings of this study are the extension to the studies carried out on single-sourced storage systems, and showcase great potential of RL in managing the control of hybrid plants with cooperative, multiple-storage methods. However, the proposed implementation is still far away from commercial viability and adoption, due to its low power resolution, accuracy and simplicity of the model, especially in the representation of its learning environment, accuracy in modelling of storage modules, as well as the choice of tuned hyper-parameters. Many assumptions and over-simplifications were made to account for time-constraints of this project, and ultimately arised due to lack of experience on the part of the author with research in RL domain . Substantial amount of time was taken to understand the reinforcement learning methodology, its basic algorithms, and developing the bug-free and efficient learning environment and agent, thus leaving much less time which could be used on exploring more complex algorithms and implementations.

Nonetheless, future work could be carried out to fine-tune the method proposed in this report or to develop a higher-fidelity and more accurate model. Firstly, the HRES environment would have to be revised to correctly model each of its component and its interaction with the agent. Such an endeavor would substantially increase the complexity of the model and could transition it into continuous action space. This would likely demand an implementation of higher complexity deep reinforcement learning algorithm, many of which are

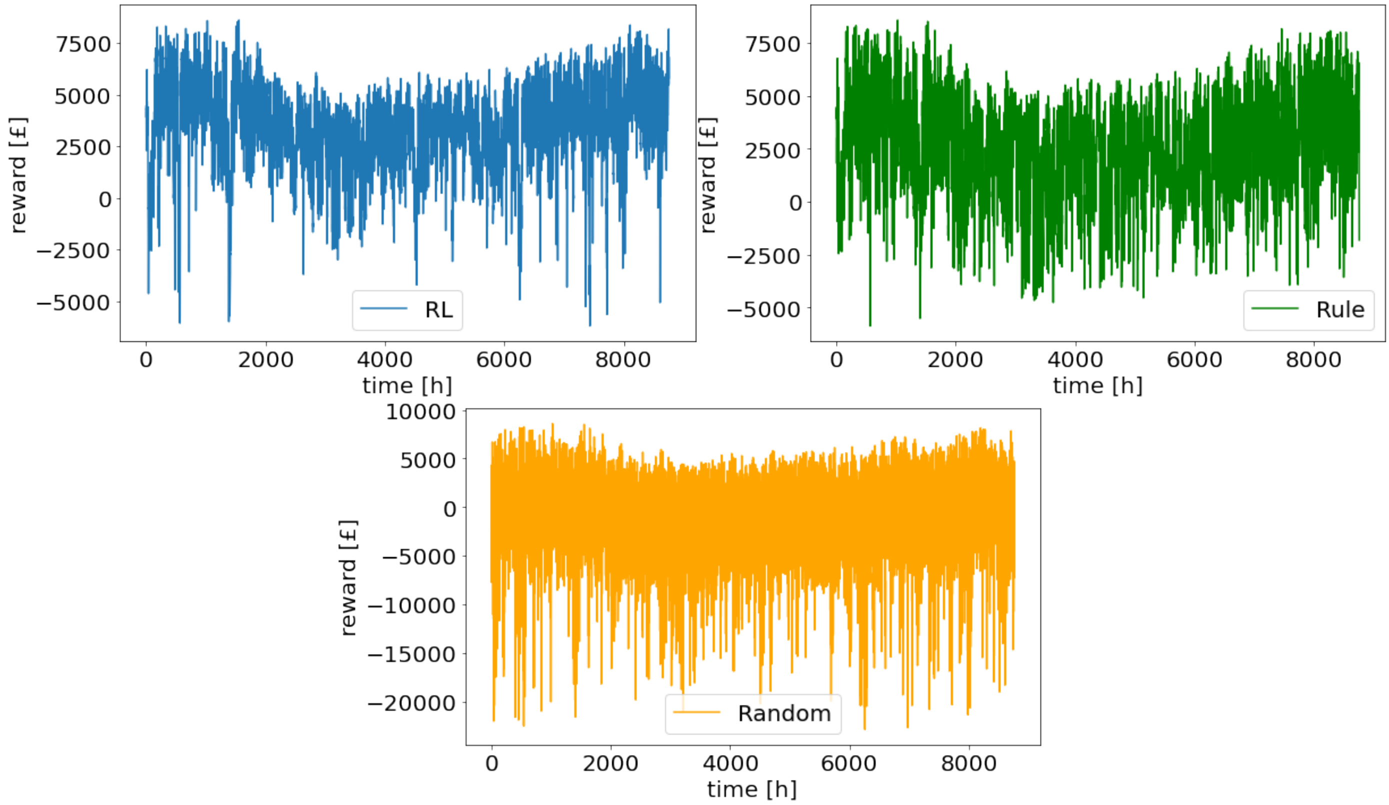

Figure 7: Rewards over time for (top-left) RL-agent, (top-right) the rule-based and (bottom centre) random-base controllers obtained from application on the evaluation dataset.

also under research for other renewable energy systems, such as Rainbow Algorithm [21], DDPG [3] and multi-step TD_{error} learning [8]. Finally the development of more complex and extensive benchmarks and baselines could aid in the analysis of such future implementations, and could help in uncovering the greater potential of reinforcement learning approaches in renewable energy power management and control.

Word count: 4623

References

[1] Muhammad Abdul Basit et al. "Limitations, challenges, and solution approaches in grid-connected renewable energy systems". In: *International Journal of Energy Research* 44.6 (2020), pp. 4132–4162.

[2] Francisca Daniel and Arnold Rix. "Reinforcement Learning-based Control System of a Hybrid Power Supply". In: *2020 International SAUPEC/RobMech/PRASA Conference*. 2020, pp. 1–6. DOI: 10.1109/SAUPEC/RobMech/PRASA48453.2020.9041138.

[3] Hongyang Dong, Jincheng Zhang, and Xiaowei Zhao. "Intelligent wind farm control via deep reinforcement learning and high-fidelity simulations". In: *Applied Energy* 292 (2021), p. 116928.

[4] Ottmar Edenhofer. *Climate change 2014: mitigation of climate change*. Vol. 3. Cambridge University Press, 2015.

[5] Roy Fox, Ari Pakman, and Naftali Tishby. "Taming the noise in reinforcement learning via soft updates". In: *arXiv preprint arXiv:1512.08562* (2015).

[6] Hirak Al-Hammad et al. "Renewable energy in hybrid mini-grids and isolated grids: economic benefits and business cases". In: *Frankfurt School—UNEP Collaborating Centre for Climate and Sustainable Energy Finance* (2015).

[7] Diederik P Kingma and Jimmy Ba. "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980* (2014).

[8] R Leo, R S Milton, and S Sibi. "Reinforcement learning for optimal energy management of a solar microgrid". In: *2014 IEEE Global Humanitarian Technology Conference - South Asia Satellite (GHTC-SAS)*. 2014, pp. 183–188. DOI: 10.1109/GHTC-SAS.2014.6967580.

[9] Sergey Levine et al. "Offline reinforcement learning: Tutorial, review, and perspectives on open problems". In: *arXiv preprint arXiv:2005.01643* (2020).

[10] Teng Liu et al. "Reinforcement learning–based energy management strategy for a hybrid electric tracked vehicle". In: *Energies* 8.7 (2015), pp. 7243–7260.

[11] Volodymyr Mnih et al. "Human-level control through deep reinforcement learning". In: *nature* 518.7540 (2015), pp. 529–533.

[12] M. H. Nehrir et al. "A Review of Hybrid Renewable/Alternative Energy Systems for Electric Power Generation: Configurations, Control, and Applications". In: *IEEE Transactions on Sustainable Energy* 2.4 (2011), pp. 392–403. DOI: 10.1109/TSTE.2011.2157540.

[13] Bassey Etim Nyong-Bassey et al. "Reinforcement learning based adaptive power pinch analysis for energy management of stand-alone hybrid energy storage systems considering uncertainty". In: *Energy* 193 (2020), p. 116622.

[14] A.T.D. Perera and Parameswaran Kamalaruban. "Applications of reinforcement learning in energy systems". In: *Renewable and Sustainable Energy Reviews* 137 (2021), p. 110618. ISSN: 1364-0321. DOI: https://doi.org/10.1016/j.rser.2020.110618. URL: https://www.sciencedirect.com/science/article/pii/S1364032120309023.

[15] Phan and Ying-Chih Lai. "Control Strategy of a Hybrid Renewable Energy System Based on Reinforcement Learning Approach for an Isolated Microgrid". In: *Applied Sciences* 9 (Sept. 2019), p. 4001. DOI: 10.3390/app9194001.

[16] Roberto Rocchetta et al. "A reinforcement learning framework for optimal operation and maintenance of power grids". In: *Applied energy* 241 (2019), pp. 291–301.

[17] Simon R. Sinsel, Rhea L. Riemke, and Volker H. Hoffmann. "Challenges and solution technologies for the integration of variable renewable energy sources—a review". In: *Renewable Energy* 145 (2020), pp. 2271–2285. ISSN: 0960-1481. DOI: https://doi.org/10.1016/j.renene.2019.06.147. URL: https://www.sciencedirect.com/science/article/pii/S0960148119309875.

[18] Hado Van Hasselt, Arthur Guez, and David Silver. "Deep reinforcement learning with double q-learning". In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 30. 1. 2016.

[19] Christopher JCH Watkins and Peter Dayan. "Q-learning". In: *Machine learning* 8.3-4 (1992), pp. 279–292.

[20] Lei Xi et al. "A deep reinforcement learning algorithm for the power order optimization allocation of AGC in interconnected power grids". In: *CSEE Journal of Power and Energy Systems* 6.3 (2020), pp. 712–723. DOI: 10.17775/CSEEJPES.2019.01840.

[21] J.J. Yang et al. "A deep reinforcement learning method for managing wind farm uncertainties through energy storage system control and external reserve purchasing". In: *International Journal of Electrical Power Energy Systems* 119 (2020), p. 105928. ISSN: 0142-0615. DOI: https://doi.org/10.1016/j.ijepes.2020.105928. URL: https://www.sciencedirect.com/science/article/pii/S0142061519312505.