Supplementary Information

Machine Learning-based prediction and optimization of plasma-catalytic dry reforming of methane in a dielectric barrier discharge reactor

Jiayin Li[1], Jing Xu[2], Evgeny Rebrov[3, 4] and Annemie Bogaerts[1,*]

[1] Research Group PLASMANT, Department of Chemistry, University of Antwerp, Antwerp, 2610, Belgium

[2] School of Electronic Information and Communications, Huazhong University of Science and Technology, Wuhan, Hubei, 430074, People's Republic of China

[3] School of Engineering, University of Warwick, Coventry, CV4 7AL, the United Kingdom

[4] Department of Chemical Engineering and Chemistry, Eindhoven University of Technology, P.O. Box 513, 5600 MB Eindhoven, the Netherlands

[*]Corresponding author: jiayin.li@uantwerpen.be (J. Li), annemie.bogaerts@uantwerpen.be (A. Bogaerts).

Contents

1. List of all abbreviations in the paper

Table S1. List of all abbreviations

Original description	Abbreviations
Dry reforming of methane	DRM
Microwave discharge	MW
Dielectric barrier discharge	DBD
Artificial intelligence	AI
Machine learning	ML
Supervised learning	SL
Reinforcement learning	RL
Artificial neural network	ANN
Mean square error	MSE
Coefficient of determination	R^2
Reinforcement learning controllers	RLC
Energy cost	EC
Backpropagation	BP
Pearson's Correlation Coefficient	PCC
Proximal Policy Optimization	PPO
Actor-Critic	AC
Energy yield	EY
Atmospheric pressure glow discharge	APGD
Temporal difference	TD

2. Experimental setup

Fig. S1 shows the schematic overview of the experimental setup, which was detailly described in Ref. [1]. The catalyst preparation and reaction performance diagnostics were described in detail in Ref. [2].

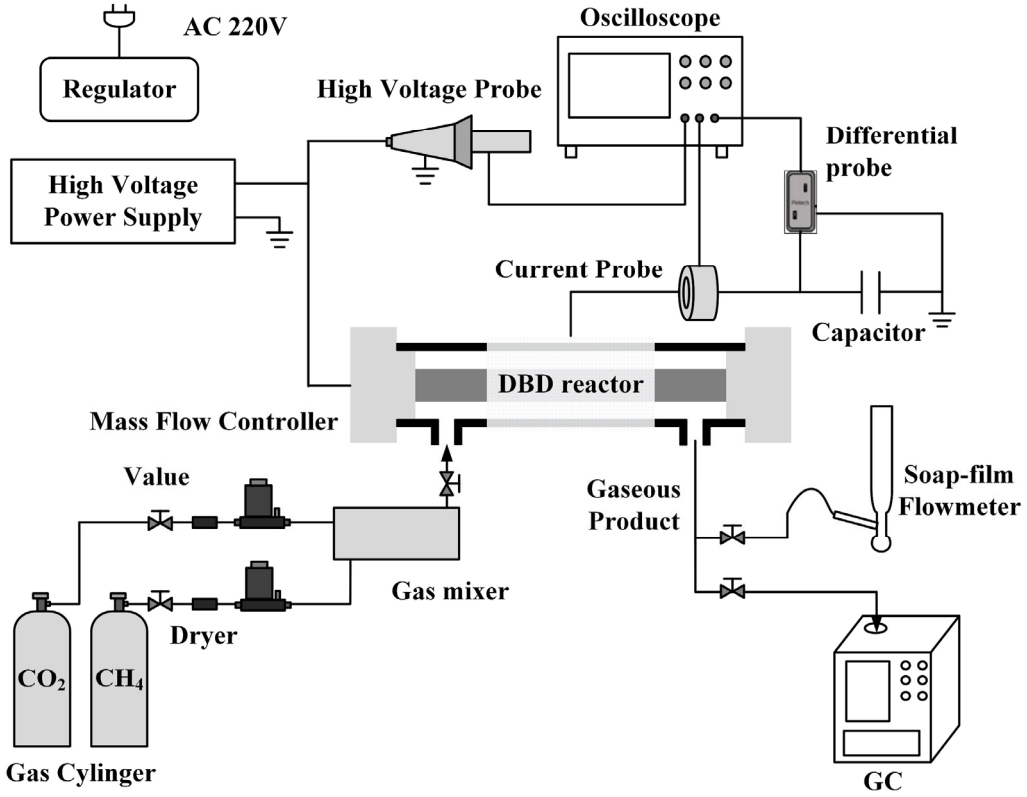

Fig. S1. Schematic overview of the experimental setup [2].

3. The database for the ML model development

Table S2. Operating parameters for the plasma-catalytic DRM process

No.	Ni loading (wt%)	Discharge Power (W)	CO_2/CH_4 molar ratio	Total flow rate (mL/min)
1	5	20	1	75
2	7.5	20	1	75
3	10	20	1	75
4	12.5	20	1	75
5	15	20	1	75
6	5	20	1.25	75
7	7.5	20	1.25	75
8	10	20	1.25	75
9	12.5	20	1.25	75
10	15	20	1.25	75
11	5	20	1.5	75
12	7.5	20	1.5	75
13	10	20	1.5	75

14	12.5	20	1.5	75
15	15	20	1.5	75
16	5	20	0.75	75
17	7.5	20	0.75	75
18	10	20	0.75	75
19	12.5	20	0.75	75
20	15	20	0.75	75
21	5	20	0.5	75
22	7.5	20	0.5	75
23	10	20	0.5	75
24	12.5	20	0.5	75
25	15	20	0.5	75
26	10	20	1	75
27	7.5	30	1.25	100
28	7.5	30	0.75	100
29	12.5	30	0.75	100
30	12.5	30	1.25	100
31	7.5	30	0.75	50
32	7.5	30	1.25	50
33	12.5	30	0.75	50
34	10	40	1.5	75
35	10	40	0.5	75
36	10	40	1	75
37	10	40	1	75
38	15	40	1	75
39	5	40	0.5	25
40	7.5	40	0.5	25
41	10	40	0.5	25
42	12.5	40	0.5	25
43	15	40	0.5	25
44	5	40	0.75	25
45	7.5	40	0.75	25
46	10	40	0.75	25
47	12.5	40	0.75	25
48	15	40	0.75	25
49	5	40	1.5	25
50	7.5	40	1.5	25
51	10	40	1.5	25
52	12.5	40	1.5	25
53	15	40	1.5	25
54	5	40	1.25	25
55	7.5	40	1.25	25
56	10	40	1.25	25

57	12.5	40	1.25	25
58	15	40	1.25	25
59	5	40	1	25
60	7.5	40	1	25
61	10	40	1	25
62	12.5	40	1	25
63	15	40	1	25
64	7.5	40	0.5	125
65	7.5	40	0.75	125
66	7.5	40	1	125
67	7.5	40	1.25	125
68	7.5	40	1.5	125
69	15	40	0.5	125
70	15	40	0.75	125
71	15	40	1	125
72	15	40	1.25	125
73	15	40	1.5	125
74	5	40	0.5	125
75	5	40	0.75	125
76	5	40	1	125
77	5	40	1.25	125
78	5	40	1.5	125
79	12.5	40	0.5	125
80	12.5	40	0.75	125
81	12.5	40	1	125
82	12.5	40	1.25	125
83	12.5	40	1.5	125
84	10	40	0.5	125
85	10	40	0.75	125
86	10	40	1	125
87	10	40	1.25	125
88	10	40	1.5	125
89	15	40	1	75
90	7.5	50	1.25	50
91	12.5	50	0.75	100
92	7.5	50	0.75	100
93	7.5	50	0.75	50
94	7.5	50	1.25	100
95	12.5	50	1.25	100
96	12.5	50	1.25	50
97	7.5	60	1	75
98	7.5	60	1.5	75
99	10	60	1	75

100	10	60	0.5	75

Table S3. Experimental results for the plasma-catalytic DRM process

No.	CO yield (%)	H_2 yield (%)	CO_2 conversion (%)	CH_4 conversion (%)	Total conversion (%)	Energy cost (eV/molec)
1	8.4	6.5	13.3	20.1	16.7	23.87604731
2	9.5	7.2	14.8	21.4	18.1	22.02928122
3	9.9	7.4	15.2	21	18.1	22.02928122
4	9.3	7	14.5	19	16.75	23.80477552
5	7.9	6.1	12.8	15.4	14.1	28.2787227
6	9.8	7.9	11.7	25	17.552	22.71706871
7	11	8.6	13.3	25.7	18.756	21.25879665
8	11.3	8.7	13.7	24.9	18.628	21.40487385
9	10.7	8.3	13.1	22.4	17.192	23.1927635
10	9.3	7.4	11.3	18.4	14.424	27.64351012
11	11	9.1	9.7	29.5	17.62	22.62939785
12	12.1	9.7	11.3	29.8	18.7	21.32245936
13	12.3	9.9	11.8	28.5	18.48	21.57629816
14	11.7	9.4	11.2	25.6	16.96	23.510023
15	10.3	8.5	9.4	21	14.04	28.39957194
16	6.5	4.9	14.4	15	14.742	27.04721137
17	7.7	5.6	15.9	16.6	16.299	24.4634634
18	8.1	5.8	16.3	16.7	16.528	24.12451537
19	7.6	5.5	15.6	15.2	15.372	25.9387191
20	6.2	4.6	13.8	12	12.774	31.21418428
21	4.3	3.1	15.1	9.4	11.281	35.34526993
22	5.6	3.9	16.6	11.6	13.25	30.09282944
23	6	4.1	16.9	12.1	13.684	29.13840909
24	5.5	3.8	16.2	11	12.716	31.35655788
25	4.1	3	14.3	8.4	10.347	38.53580652
26	9.8	7.3	15.5	21	18.25	21.84821863
27	9.8	7.4	11.4	22.5	16.284	27.54674765
28	6.7	4.7	14	14.2	14.114	31.78200643
29	6.6	4.6	13.7	14.2	13.985	32.07516902
30	9.5	7.2	11.2	20.6	15.336	29.24955913
31	10.9	9.3	23.5	26.2	25.039	35.82980461
32	14.7	12.5	19.7	37.1	27.356	32.79508984
33	10.7	9	23	24.3	23.741	37.78874005
34	15.4	12.8	15.8	37.8	24.6	32.41707236
35	8.4	6.9	23.3	19.4	20.687	38.54884614
36	12.6	10.2	20.5	29.3	24.9	32.02650522
37	12.6	10.2	20.5	29.3	24.9	32.02650522
38	10.6	8.9	17.8	24.8	21.3	37.43943568

39	12.8	12	33.7	29.8	31.087	76.95756877
40	13.9	12.6	34.9	31.6	32.689	73.18608523
41	14.2	12.7	35.1	31.7	32.822	72.8895235
42	13.6	12.2	34.1	30.3	31.554	75.8185948
43	12.1	11.3	32	27.2	28.784	83.11492288
44	15.4	14.1	31.8	37.2	34.878	68.59280751
45	16.5	14.7	33.1	38.5	36.178	66.12803196
46	16.8	14.7	33.2	38.2	36.05	66.36282775
47	16.2	14.3	32.3	36.3	34.58	69.18391961
48	14.7	13.2	30.2	32.8	31.682	75.51227638
49	21.3	19.3	23.5	57.3	37.02	64.62398542
50	22.3	19.8	24.8	57.2	37.76	63.3575196
51	22.5	19.8	25.1	55.5	37.26	64.20772786
52	21.8	19.2	24.2	52.2	35.4	67.58135424
53	20.2	18.1	22.3	47.3	32.3	74.06749041
54	19.7	17.7	26.7	50.9	37.348	64.05644051
55	20.8	18.3	28	51.3	38.252	62.54261059
56	20.9	18.3	28.2	50.1	37.836	63.23025532
57	20.3	17.7	27.3	47.3	36.1	66.27091247
58	18.7	16.6	25.4	42.8	33.056	72.37354611
59	17.8	16	29.5	44.2	36.85	64.92211507
60	18.8	16.6	30.8	45.1	37.95	63.04031463
61	19	16.6	31	44.3	37.65	63.54262789
62	18.4	16.1	30	41.9	35.95	66.54742532
63	16.9	15	28	38	33	72.49636182
64	2.6	1.5	11.6	7.9	9.121	52.45872032
65	4.6	3.1	10.9	12.1	11.584	41.30490228
66	6.2	4.4	9.9	15.9	12.9	37.09116186
67	7.5	5.6	8.4	19.4	13.24	36.13866979
68	8.5	6.5	6.4	22.6	12.88	37.14875684
69	1.3	0.8	9.5	9	9.165	52.20687267
70	3.2	2.2	9	11.8	10.596	45.15628426
71	4.8	3.5	8	14.3	11.15	42.91264467
72	6	4.5	6.6	16.5	10.956	43.67250712
73	6.8	5.4	4.7	18.3	10.14	47.18698107
74	1.3	0.7	10.1	4.3	6.214	76.99967622
75	3.3	2.3	9.4	8.9	9.115	52.49325157
76	5	3.6	8.3	13.2	10.75	44.50939424
77	6.3	4.8	6.8	17.2	11.376	42.06012553
78	7.3	5.8	4.8	20.9	11.24	42.56903808
79	2.6	1.6	11.3	10.2	10.563	45.29735757
80	4.5	3.1	10.7	13.5	12.296	38.91314151
81	6.1	4.3	9.7	16.5	13.1	36.52488458

82	7.3	5.4	8.3	19.1	13.052	36.6592084
83	8.2	6.3	6.4	21.3	12.36	38.71164952
84	3	1.8	12	9.8	10.526	45.45658256
85	5	3.3	11.4	13.6	12.654	37.81223234
86	6.6	4.6	10.4	17	13.7	34.9252546
87	7.9	5.7	8.9	20	13.784	34.71241933
88	8.8	6.7	7	22.8	13.32	35.92162072
89	11	9	18.1	25.4	21.75	36.66482667
90	20.5	17.4	26.5	48.4	36.136	41.37805686
91	8.6	6.8	18.6	23	21.108	35.41873846
92	8.8	7	19	21.9	20.653	36.19903798
93	16.5	14	31.6	36.4	34.336	43.5472234
94	12.2	9.8	15.2	31.1	22.196	33.68258836
95	11.9	9.5	14.9	30.4	21.72	34.4207519
96	20.1	16.9	25.9	45.8	34.656	43.14512531
97	17.5	14.4	26.7	40.3	33.5	35.70716329
98	20.7	17.2	20.8	50.7	32.76	36.51373535
99	17.8	14.4	27	41	34	35.18205794
100	13.2	11	31	30.1	30.397	39.35223772

4. Data processing

To normalize the input and output data, the linear function method was used to transform the sample data into the interval [0,1]:

$$X_{norm} = \frac{X - X_{\min}}{X_{\max} - X_{\min}} \tag{1}$$

where X_{norm} denotes the normalized input parameters and predicted output; X denotes the original input parameters and actual output; $X_{\max}$ and $X_{\min}$ denotes the maximum and minimum value of the data, respectively. The inverse normalization formula for the predicted output is:

$$Y = Y_{\min} + Y_{norm}(Y_{\max} - Y_{\min}) \tag{2}$$

where Y_{norm} denotes the normalized predicted value; Y denotes the inverse normalized predicted value; and $Y_{\max}$ and $Y_{\min}$ denote the maximum and minimum values of the data, respectively.

5. Hyperparameters optimization

Hyperparameters are parameters that are set before the actual learning process begins. For an artificial neural network (ANN) model, the number of hidden layers and neurons determine the capacity of the network. More layers and neurons can increase the model's ability to learn complex patterns but may also increase the risk of overfitting. Therefore, it was optimized by the grid search method to find the optimal combination within the specified ranges. For the reinforcement learning (RL) model, the

hyperparameter selection was optimized by a random search method: 1) Define a search space for each hyperparameter; 2) Generate multiple configurations by randomly sampling from the defined search space; 3) Train the RL model with each configuration, and then evaluate the performance of each configuration using a validation metric (e.g., accumulative reward); 4) Choose the configuration that performs best based on the evaluation metric, and consider running additional simulations with slight variations around the best configuration to refine the results.

6. Artificial neural network model

An ANN is a common SL model [3,4]. Indeed, it can be viewed as a sophisticated function [5]. The goal of ANN training process is to find optimum weight and bias values to reduce the discrepancy between predicted and actual values. The loss function at the output serves as an adjustment signal to constantly optimize the weights in the direction of the input and will reduce to an acceptable level or within a specified number of training epochs (where epoch is defined as one complete cycle of training data being processed through the algorithm). Fig. S2 presented the example of this process. The MSE of the network decreases continuously during the training and converges at about 60 epochs. Once the loss function is computed, the partial derivatives of the weights are determined by using the chain rule, then we use gradient descent method to update the network parameters.

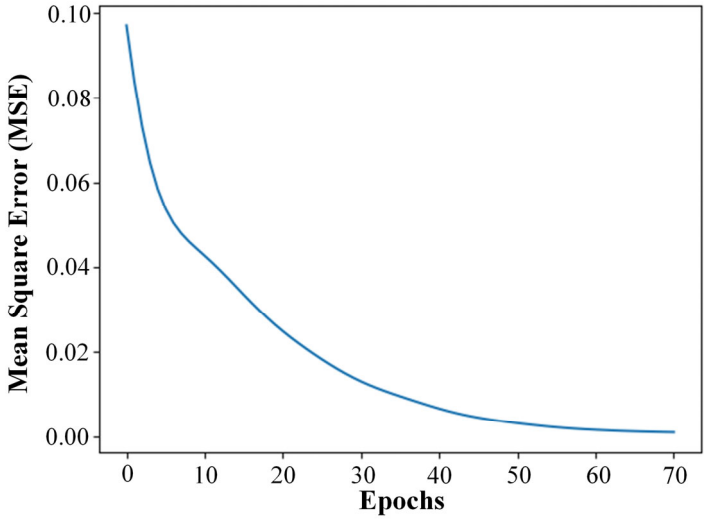

Fig. S2. MSE of the best fitness value in each epoch for ANN

7. Relative significance of different parameters

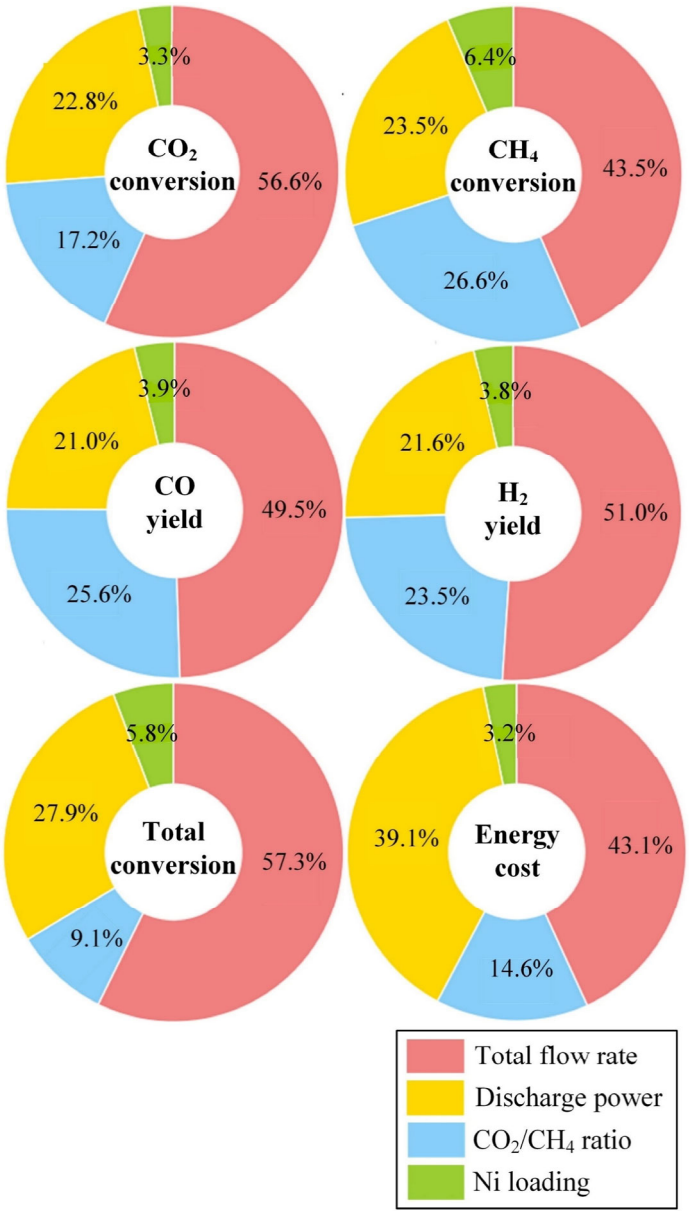

Fig. S3. Importance analysis between reaction performance and four different operating parameters.

8. Reinforcement learning model

8.1 Basic concepts

RL aims to formalize the decision-process based on experience through interaction with the world. The decision-maker or learner is called as *agent* while the *environment* encompasses everything the agent interacts with. At any time step t, the agent observes the environment, represented as state s_t, and it must choose an action $a(t)$, and it also receives some rewards, $r(t)$. Through this action, it reaches new state (s_{t+1}). The objective is to develop a policy π ($a|s$), in order to maximize the expected cumulative future reward. Thus, the reward signal determines which actions are good or bad, guiding the agent's subsequent actions. Fig. S4 shows such a closed-loop operation. In the next, we will present fundamental RL concepts and explain how to train the RL

controller (RLC) for plasma-catalytic DRM process discussed in this paper.

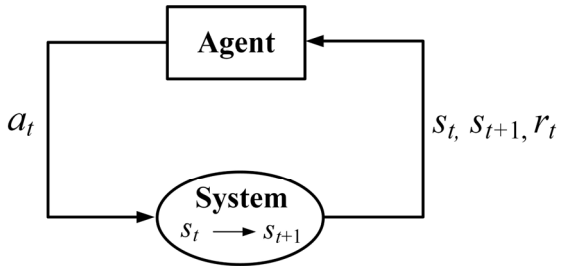

Fig. S4. Schematic principle of RL.

8.2 States, actions, and rewards

The states and actions are defined by the four operating parameters described in the main paper. It should be noted that the agent's actions must remain within the defined limits when considering the model within investigated range. The reward function is employed to guide optimization toward the best possible outcome. In our case, we aim to maximize the reaction performance (gas conversion and product yield) while minimizing the EC. Therefore, the reward function is determined by the value difference between current time step and previous time step during the iteration. Specifically, the higher the reaction performance and lower the energy cost, the higher is the reward.

8.3 Actor-critic framework

The Actor-critic (AC) algorithm is an extension of the idea of gradient bandit methods [6]. The *actor* is responsible for choosing actions, represented by the policy and the *critic* is used to evaluate the quality of actions made by the *actor*. The working process of AC framework is presented in Fig. S7:

(1) The actor selects an action a_t by using its policy $\pi_\theta(a_t|s_t)$

(2) According to the current state s_t, leading to a new state s_{t+1} and a reward r_t;

(3) Based on the reward, the critic compares the value of the new state $V(s_{t+1})$ with the previous state $V(s_t)$ by evaluating the temporal difference (TD) error:

$$\delta_t = r_t + \gamma V_\phi(s_{t+1}) - V_\phi(s_t) \tag{3}$$

where γ denotes the discount factor (0,1).

(4) The critic's value function is updated by gradient descent to minimize TD error:

$$\theta_c \longleftarrow \theta_c + \alpha_c \delta_t \nabla_{\theta_c} V(s_t) \tag{4}$$

where α_c and θ_c are the learning rate and parameters for the critic network, respectively.

(5) Based on the feedback from the critic, the actor's policy is updated by policy gradient [7]:

$$\nabla_\theta J(\theta) = \nabla_\theta \log\pi(a_t|s_t;\ \theta) \cdot \delta_t \tag{5}$$

The actor updates the policy according to the critic's evaluation, to favor actions that are more likely to yield higher rewards:

$$\theta \longleftarrow \theta + \alpha_a \nabla_\theta J(\theta) \tag{6}$$

where α_a and θ are the learning rate and parameters for the actor network, respectively.

The process is repeated for step (1) to (5) in the episode, continuously updating the actor and critic until convergence or until the end of the episode.

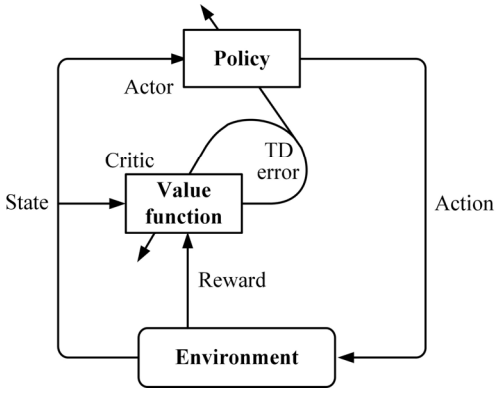

Fig. S5: The actor–critic architecture.

8.4 Proximal policy optimization algorithm

Proximal policy optimization (PPO) is a specific algorithm built on the AC framework, designed to improve the stability and performance by constraining policy updates through clipping. The PPO algorithm updates its critic network similarly to the AC algorithm, but it features two types of policies in its actor network, which are called the target policy and the current policy, respectively. The current policy is used to generate the batch of trajectories (sequences of states, actions, and rewards), and the target policy updates the gradient according to these data and updates the current strategy at the end of each iteration cycle. The probability ratio $r_t(\theta)$ measures the divergence between the new policy $\pi_\theta(a_t|s_t)$ and the old one $\pi_{\theta\_old}(a_t|s_t)$ for a given action by importance sampling：

$$r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta\_old}(a_t|s_t)} \tag{7}$$

The TD error is then multiplied by the probability ratio to update the target policy, thus transforming the AC algorithm from an On policy to Off policy. The PPO's objective function is designed to maximize the expected reward, while keeping the target policy close to the current policy. Therefore, the loss function is clipped in the PPO algorithm as follows:

$$L^{Clip}(\theta) = Clip(r_t(\theta), 1-\varepsilon, 1+\varepsilon) \cdot \delta_t \tag{8}$$

where ε denotes a hyperparameter that determines the clipping range, restricting $r_t(\theta)$ to the interval $[1-\varepsilon,1+\varepsilon]$.

8.5 Network structure for RL agent training

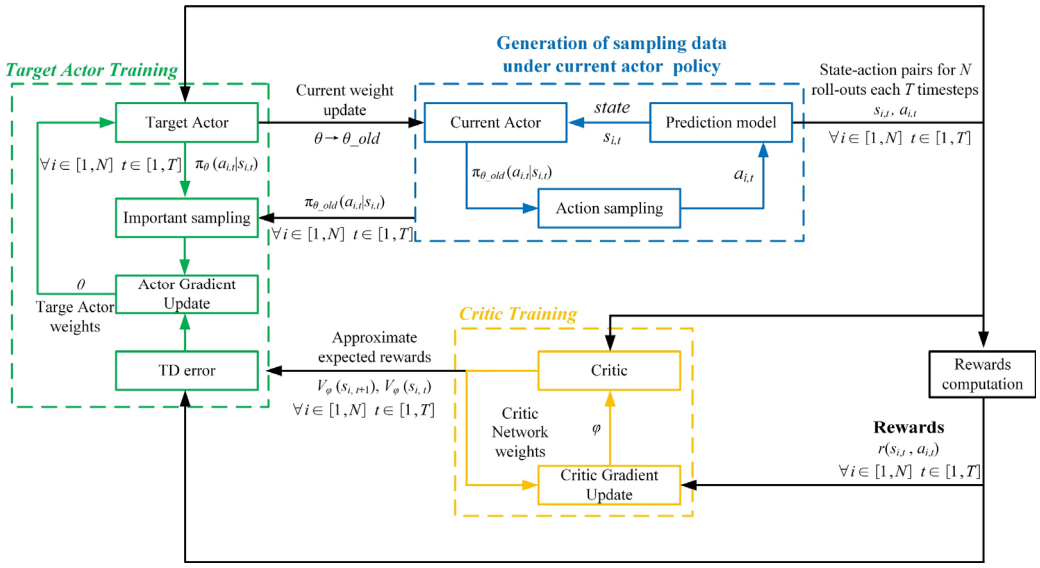

Fig. S6. Block diagram representation of the network structure.

The block diagram of the PPO algorithm is shown in Fig. S8. Each iteration cycle needs to generate a batch of training data (N = 200 roll-outs and T = 300 time steps) based on the current actor policy to update current weight. At the start of each roll-out, we randomly select a new setpoint from the uniform distribution, $\chi_{sp} \sim u$ [7,8]. In each actor training step, we perform 10 times updates of the critic target values, with the critic network being updated 1 gradient steps per target update. Based on the optimised crtic network, the target actor network is updated with 1 gradient steps.

Table S4. Detailed parameters of the RL controller

Parameter	Actor network	Critic network
Number of input layers	4	4
Number of hidden layer	16	16
Number of output layers	4	1
Activation function	tanh	softplus
Learning rate α	1e-3	1e-2
Discount γ	0.98	
Scaling factor	0.95	
Clipped factor ε	0.2	

The parameters of the RLC are shown in Table S4. The input layer of the network contains one hidden layer with 16 neurons. The output layer of the actor network contains 4 nodes, represented by the probability distribution of the four action parameters. The activation function for the mean parameter is a tanh function. The sampled action values are controlled by clipping in the range of [0,1]. The softplus activation function is used for the critic network.

9. Comparative analysis of the predicted and experimental results

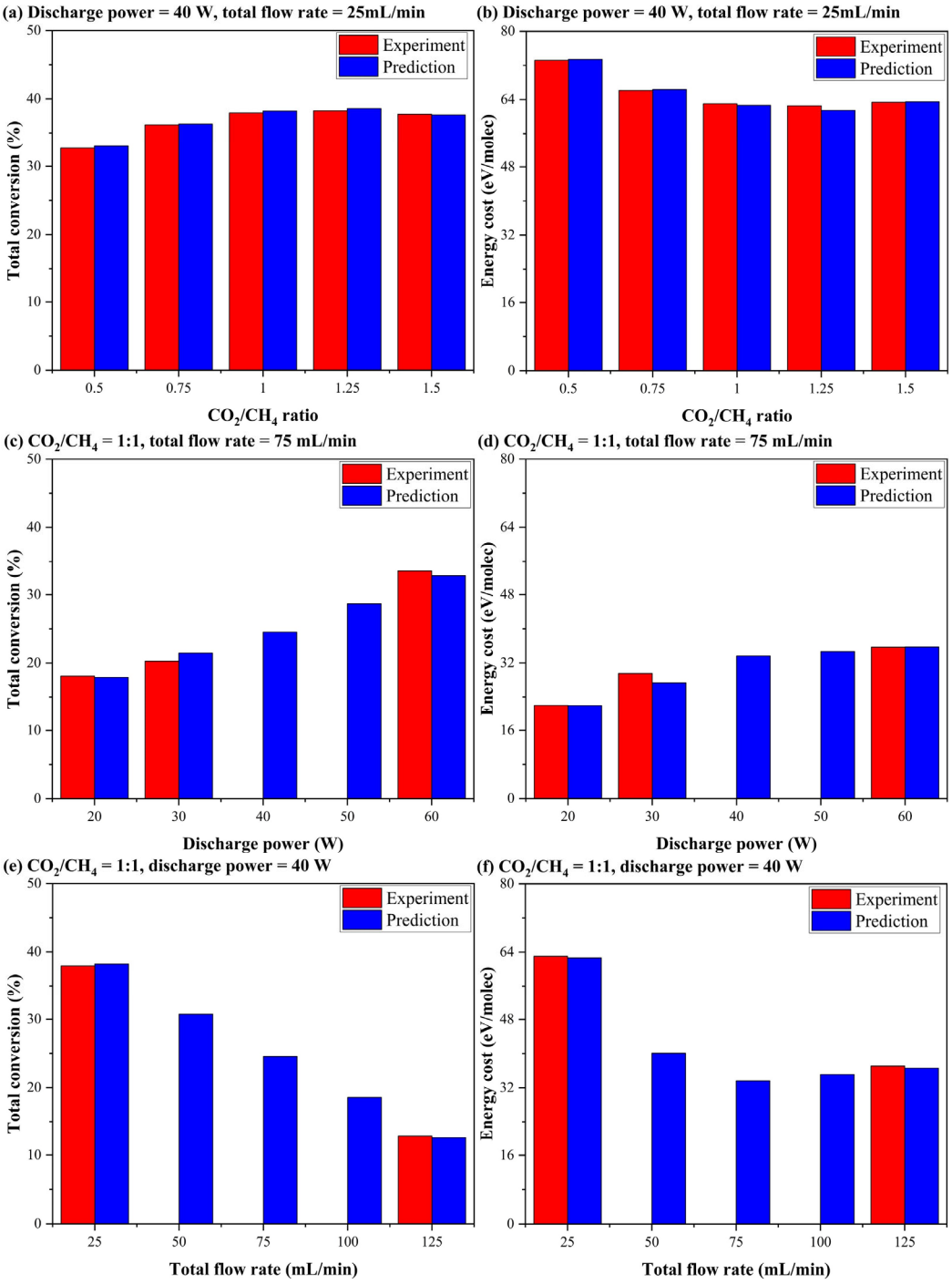

Fig. S7. Comparison of predicted values with available experimental data, using 7.5 wt% Ni/Al_2O_3 for total conversion (a, c, e) and energy cost (b, d, f).

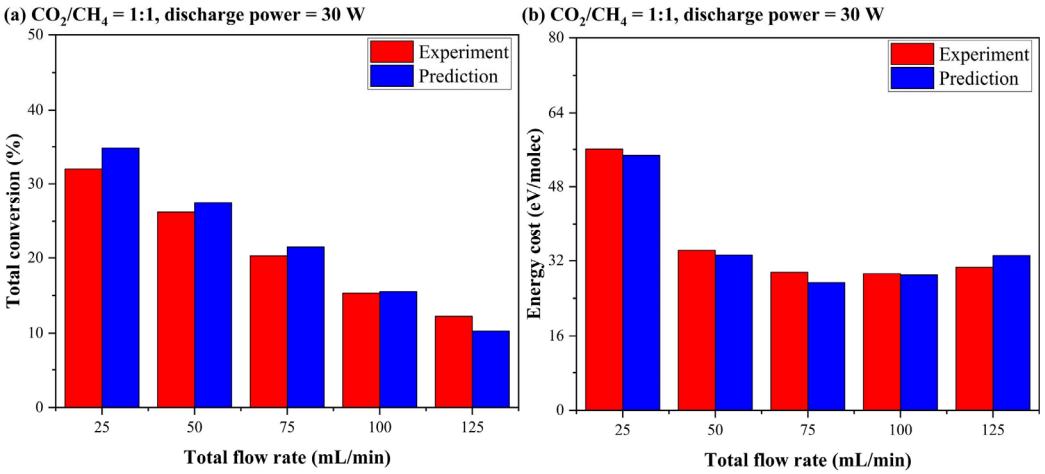

Fig. S8. Comparison of predicted values and unseen experimental data, using 7.5 wt% Ni/Al_2O_3 for model generalization evaluation: (a) Total conversion; (b) Energy cost.

10. RL agents training results

The maximum training iterations for the agent is 300. Before training, the two RLC will generate 1 random number uniformly distributed within their limited interval, and each random number will be trained for 200 rounds. Therefore, in each iteration cycle, 200 data are used to train the critic network. The training results of CO_2 conversion and total conversion within investigated range are shown in Figure S9. The average return of both agents in each iteration converges to the maximum value 23 after about 25 iterations.

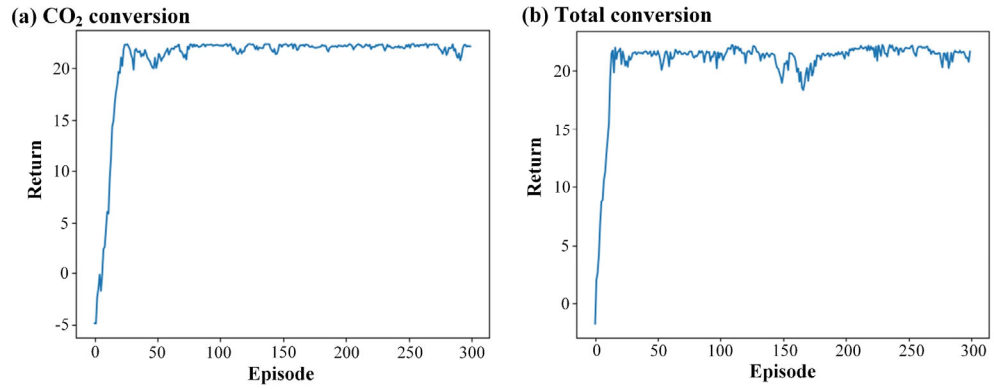

Fig. S9. Trainning curve of CO_2 conversion and total conversion RL agent.

11. A visualization figure of the input distribution for the SL model

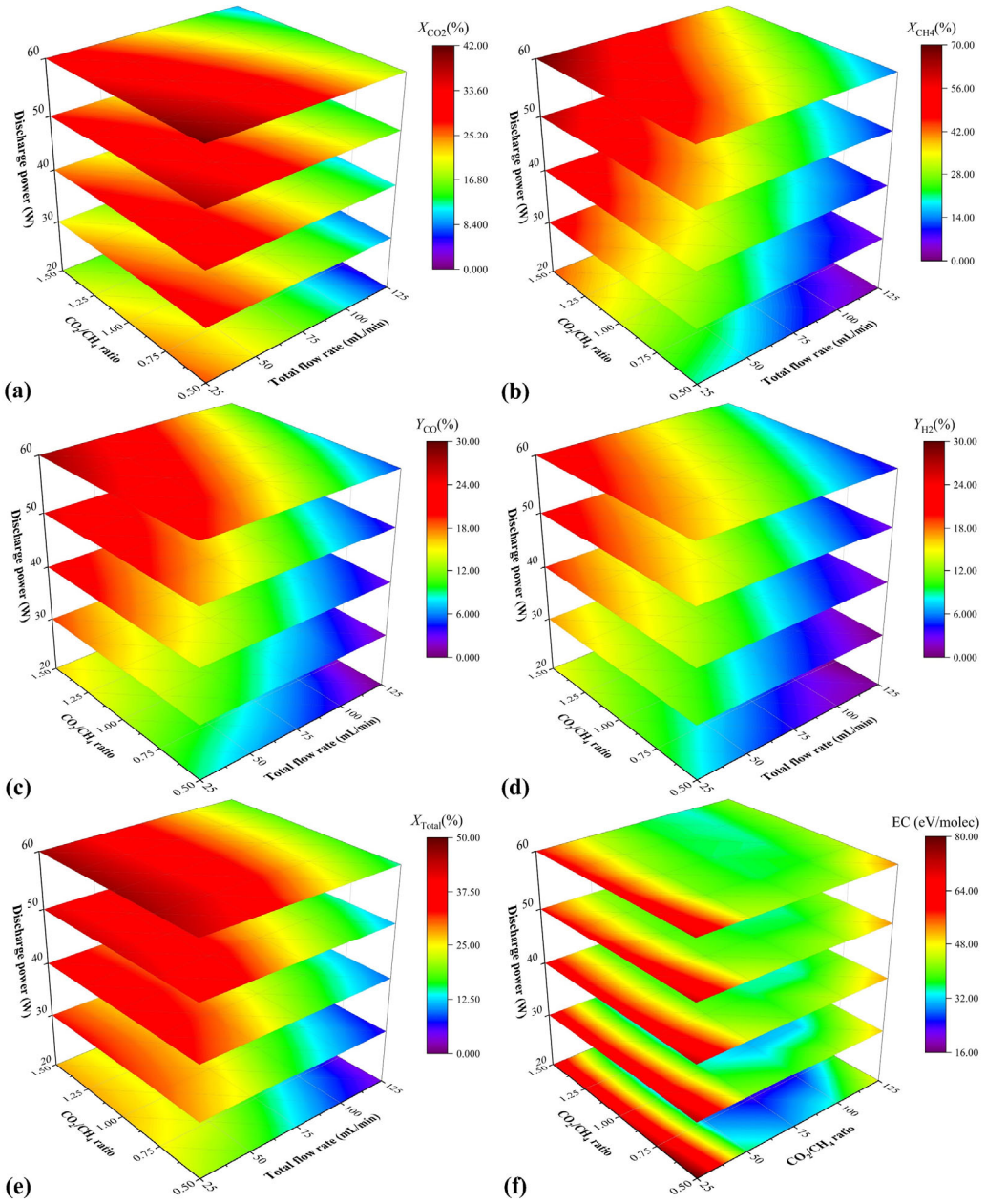

Fig. S10. Prediction results of input distribution including discharge power, CO_2/CH_4 ratio, and total flow rate on the performance of plasma-catalytic DRM process using 7.5 wt% Ni/Al_2O_3. (a) CO_2 conversion; (b) CH_4 conversion; (c) CO yield; (d) H_2 yield; (e) Total conversion; (f) EC.

References

[1] D. Mei, B. Ashford, Y.-L. He, X. Tu, Plasma-catalytic reforming of biogas over supported Ni catalysts in a dielectric barrier discharge reactor: Effect of catalyst supports, Plasma Process. Polym. 14 (2017) 1600076. https://doi.org/10.1002/ppap.201600076.

[2] Y. Cai, D. Mei, Y. Chen, A. Bogaerts, X. Tu, Machine learning-driven optimization of plasma-catalytic dry reforming of methane, J. Energy Chem. 96 (2024) 153–163.

https://doi.org/10.1016/j.jechem.2024.04.022.
[3] M.A.N. Dewapriya, R.K.N.D. Rajapakse, W.P.S. Dias, Characterizing fracture stress of defective graphene samples using shallow and deep artificial neural networks, Carbon 163 (2020) 425–440. https://doi.org/10.1016/j.carbon.2020.03.038.
[4] Y. Wang, C. Ling, H. Yin, W. Liu, Z. Tang, Z. Li, Thermophysical properties of KCl-NaF reciprocal eutectic by artificial neural network prediction and experimental measurements, Solar Energy 204 (2020) 667–672. https://doi.org/10.1016/j.solener.2020.05.029.
[5] Y. Wang, Y. Chen, J. Harding, H. He, A. Bogaerts, X. Tu, Catalyst-free single-step plasma reforming of CH4 and CO2 to higher value oxygenates under ambient conditions, Chem. Eng. J. 450 (2022) 137860. https://doi.org/10.1016/j.cej.2022.137860.
[6] R.S. Sutton, D. McAllester, S. Singh, Y. Mansour, Policy Gradient Methods for Reinforcement Learning with Function Approximation, in: Advances in Neural Information Processing Systems, MIT Press, 1999. https://proceedings.neurips.cc/paper_files/paper/1999/hash/464d828b85b0bed98e80ade0a5c43b0f-Abstract.html.
[7] J. Schulman, P. Moritz, S. Levine, M. Jordan, P. Abbeel, High-Dimensional Continuous Control Using Generalized Advantage Estimation, (2018). https://doi.org/10.48550/arXiv.1506.02438.
[8] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, P. Abbeel, Domain Randomization for Transferring Deep Neural Networks from Simulation to the Real World, (2017). https://doi.org/10.48550/arXiv.1703.06907.