the k-mean algorithm. The iterative formula for updating the cluster center is shown below [43].

$$a_k = \frac{\sum_{i=1}^{n} z_{ik} x_{ij}}{\sum_{i=1}^{n} z_{ik}} \tag{1}$$

$$z_{ik} = \begin{cases} 1 & \text{if } \|x_i - a_k\|^2 = \min_{l \leqslant k \leqslant c} \|x_i - a_k\| \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

where $\| x_i - a_k \|$ is the Euclidean distance between data x_i and the cluster center a_k.

3.2 Reinforcement learning

When faced with a problem, different actions are adopted to influence the outcome by perceiving the environment and maximizing the benefits of learning by interaction, which is also referred to as reinforcement learning. Current reinforcement learning algorithms utilize neural networks to extract high-dimensional characteristics from observed data and represent their strategies or value functions as function approximators.

The agent receives state s_t at each time step t, and selects an action a_t from a collection of actions $\mathcal{A}$ according to the policy model π, where π is a map of states to actions a_t the next state s_{t+1} and scalar reward r_t as return. The whole process runs continuously to maximize rewards $R_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k}$, in which γ is the discount factor $(0 < \gamma < 1)$ and r_t is the reward at step t. In value-based model-free reinforcement learning methods, the action-value function is represented by a neural network and other function approximators.

3.2.1 A2C and A3C

A2C (Advantage Actor-critic) combines policy-based and value-based approaches, which maintain a policy $\pi(a_t \mid s_t; \theta)$ and an estimate of the value function $V(s_t; \theta)$. Policies $\pi(a_t \mid s_t; \theta)$ and $V(s_t; \theta)$ are mainly defined as the policy network and critic network, respectively. θ can be optimized by the gradient ascent method as follows [44].

$$\nabla \bar{R}_\theta \approx \frac{1}{N} \sum_{n=1}^{N} \sum_{t=1}^{T_n} \left(r_t^n + V_\pi\left(s_{t+1}^n\right) - V_\pi\left(s_t^n\right)\right) \nabla \log p_\theta\left(a_t^n \mid s_t^n\right) \tag{3}$$

A2C can learn from only one work with the environment, which differs from A3C (Asynchronous Advantage Actor-critic). A3C operates asynchronously with a global network and multiple branch networks learning from multiple works. Each branch network can be understood as a replica of the global network. Asynchronous training updates multiple branch network parameters to uninterruptedly update the global network parameters. The updates performed can be seen below [45].

$$\nabla_{\theta'} \log \pi\left(a_t \mid s_t; \theta'\right) A\left(s_t, a_t; \theta, \theta_v\right) + \beta \nabla_{\theta'} H\left(\pi\left(s_t; \theta'\right)\right) \tag{4}$$

$\beta \nabla_{\theta'} H\left(\pi\left(s_t; \theta'\right)\right)$ is the entropy, which can increase exploration of the environment and prevent premature convergence leading to suboptimal results. $A\left(s_t, a_t; \theta, \theta_v\right)$ is the advantage function, which is given as follows [45].

$$A\left(s_t, a_t; \theta, \theta_v\right) = \sum_{i=0}^{k-1} \gamma^i r_{t+i} + \gamma^k V\left(s_{t+k}; \theta_v\right) - V\left(s_t; \theta_v\right) \tag{5}$$

The parameter θ_v of the value network is then updated by the gradient descent of

$$\nabla_{\theta_v} \left[\left(\sum_{i=0}^{k-1} \gamma^i r_{t+i} + \gamma^k V\left(s_{t+k}; \theta_v\right) - V\left(s_t; \theta_v\right) \right) \right] \tag{6}$$

3.2.2 Parallel Advantage Actor-Critic (PAAC)

A general framework for deep reinforcement learning is proposed by Clemente [42], and multiple participants can be trained synchronously on one machine.

Gradients in PAAC are calculated using mini batches of experiences. Multiple environment instances are trained in parallel and may explore different states at any given time. The advantage is reducing the relevance of the state encountered and contributing to stable training [45].

The updated formulas for policy gradients ∇_θ^π and value gradients ∇_θ^V are shown below [42]

$$\begin{aligned} \nabla_\theta^\pi \approx \frac{1}{n_e \cdot t_{\max}} \sum_{e=1}^{n_e} \sum_{t=1}^{t_{\max}} \Big(Q^{(t_{\max} - t + 1)}\left(s_{e,t}, a_{e,t}; \theta, \theta_v\right) - V\left(s_{e,t}; \theta_v\right) \Big) \\ \nabla_\theta \log \pi\left(a_{e,t} \mid s_{e,t}; \theta\right) + \beta \nabla_\theta H\left(\pi\left(s_{e,t}; \theta\right)\right) \end{aligned} \tag{7}$$

$$\nabla_{\theta_v}^V \approx \nabla_{\theta_v} \frac{1}{n_e \cdot t_{\max}} \sum_{e=1}^{n_e} \sum_{t=1}^{t_{\max}} \Big(Q^{(t_{\max} - t + 1)}\left(s_{e,t}, a_{e,t}; \theta, \theta_v\right) - V\left(s_{e,t}; \theta_v\right) \Big) \tag{8}$$

3.3 Parallel Advantage Actor-Critic with K-means(PAAC-K)

The PAAC-K model consists of a parallel advantage actor-critic, K-means, reward functions, and stepwise training