

## Details on Experiments

### Network Architecture

In our framework, the actor-critic module  $M_i$  consisted of two separate neural networks: an actor network and a critic network. As illustrated in Fig. S1, the actor network consisted of three convolutional (CNN) layers followed by two fully connected (FC) layers. The final linear layer of the actor network outputted a probability distribution over the action space. We utilized the MiniHack and Atari game image with the dimension of  $1 \times 3 \times 84 \times 84$  as input of the lower layer  $L_i$ , after the first CNN layer with kernel size 8 and stride 4, the second CNN layer with kernel size 4 and stride 2, the last CNN layer with kernel size 3 and stride 1, and the first FC layer to obtain the feature vector with the dimension of 512. Then, the feature vector was input to the top layer (policy head), which consists of an FC layer, and obtained the probability distribution over the action space. The critic network shared the same architectural design as the actor network, except for the final linear layer, which produced a scalar value representing the estimated value. In our work, when expanding the network, the CNN layers and the first FC layer of the actor-critic module were used as the lower layer and initialized using the existing modules, while the last FC layer was used as the policy head and randomly initialized.

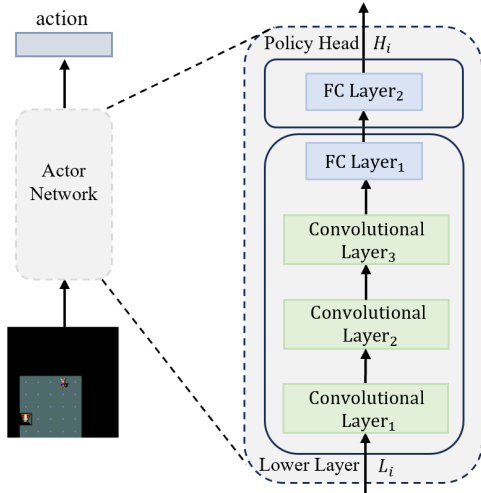


Figure S1: The CNN architecture-based actor network for the policy of the MiniHack and Atari tasks.

### Hyperparameters

In our experiments, all reinforcement learning agents were trained using an IMPALA-based training framework in both the MiniHack and Atari environments for 2 epochs. In MiniHack, each task was trained for  $1e6$  steps per epoch, with evaluations conducted every  $1e5$  steps to monitor reward performance. In the Atari environment, each task was trained for  $1e7$  steps per epoch, and evaluations were performed every  $1e6$  steps. The training hyperparameters for

our proposed method in the MiniHack and Atari environments were summarized in Table S1.

Table S1: The hyperparameters of our proposed method in the task sequences of MiniHack and Atari environments.

Hyperparameters	Ours
Num. actors	64
Learner threads	2
Batch size	32
Unroll length	25
Grad clip	40
Entropy cost	0.001
Discount factor	0.99
Learning rate	$4.8e-6$
Replay buffer size	$2e5$
Policy cloning weight	0.01
Value cloning weight	0.005
Similarity threshold	0.28

### Network Expansion During Training

The network expansion of our proposed DSNet during the training process in the MiniHack and Atari environments is shown in Fig. S2. In Fig. S2a, after training on 10 MiniHack tasks, DSNet included four actor-critic modules. During training, task similarity guided the selection of either reusing existing modules or expanding new ones. Reusing modules for similar tasks reduced network size and resource consumption, while extending the network when there is task conflict can mitigate the catastrophic forgetting. In Fig. S2b, after training on 6 Atari tasks, DSNet included six actor-critic modules, likely due to the low task correlation in the Atari environment.

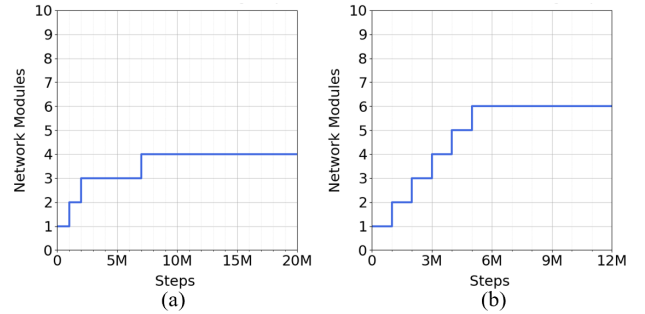


Figure S2: The network expansion during training in the (a) MiniHack and (b) Atari environments.

### The Final Reward of MiniHack and Atari Tasks

The final rewards obtained on each task by all baseline methods and our proposed DSNet training framework in MiniHack and Atari environments are reported in Tables S2 and S3. Table S2 shows that the proposed method DSNet

Table S2: The final rewards of different individual tasks in the MiniHack environment.

Tasks	EWC	P&C	CLEAR	SANE	FT	Ours
1	0.90±0.00	0.88±0.00	0.82±0.06	0.82±0.25	-0.10±0.00	0.96±0.06
2	0.23±0.11	0.22±0.16	0.51±0.27	0.01±0.55	-0.75±0.43	0.26±0.11
3	0.05±0.31	0.85±0.15	0.63±0.17	0.89±0.19	-0.03±0.13	1.00±0.00
4	-0.47±0.04	-0.29±0.04	-0.60±0.07	-0.91±0.08	-0.75±0.43	-0.54±0.17
5	0.93±0.060	0.81±0.12	0.89±0.11	0.96±0.06	-0.03±0.06	0.93±0.12
6	-0.32±0.03	-0.040±0.01	0.36±0.27	0.01±0.49	-0.35±0.00	0.87±0.15
7	0.86±0.06	0.78±0.00	0.78±0.22	1.0±0.00	0.01±0.11	0.93±0.13
8	0.62±0.06	-0.01±0.00	0.70±0.20	0.73±0.06	-0.17±0.06	0.43±0.06
9	0.69±0.10	0.81±0.06	0.64±0.07	0.85±0.13	-0.03±0.06	1.0±0.00
10	0.60±0.10	0.03±0.06	0.66±0.15	0.73±0.12	-0.18±0.04	0.46±0.31
Average	0.45±0.04	0.40±0.02	0.54±0.08	0.50±0.07	-0.24±0.07	<b>0.63±0.09</b>

Table S3: The final rewards of different individual tasks in the Atari environment.

Tasks	EWC	P&C	CLEAR	SANE	FT	Ours
1	77.00±110.64	193.83±45.87	412.67±152.34	359.00±185.33	160.00±140.00	1073.17±459.33
2	1867.67±759.75	1691.00±1214.49	7431.67±421.29	3956.33±947.06	409.67±384.07	9632.00±1836.05
3	384.93±266.74	452.67±38.33	685.60±133.22	1036.07±885.45	309.47±299.24	5013.27±1549.63
4	0	1249.33±191.76	4278.33±2574.07	7534.33±7400.35	0	16660.16±3744.36
5	940.00±124.90	850.00±186.81	1143.33±652.10	640.00±271.85	1326.67±549.30	52703.33±2025.35
6	316.67±130.24	329.33±194.63	1838.00±164.76	2394.33±390.49	1330.67±202.24	2535.33±232.42
Average	597.71±122.30	794.19±408.31	2631.60±429.89	2653.34±1182.03	589.41±62.85	<b>14557.878±895.17</b>

achieved competitive performance, with an average performance of 0.63, exceeding the second-best by 0.09. In addition, on 10 tasks, our method achieved the best performance in 1-Room-Random-5x5-v0, 3-Room-Dark-5x5-v0, 6-CorridorBattle-v0, and 9-Room-Ultimate-5x5-v0, and the second best in 2-Corridor-R2-v0, 5-Room-Monster-5x5-v0, and 7-Room-Trap-5x5-v0. Moreover, our proposed DSNet training framework obtained the best final reward on all Atari tasks as shown in Table S3, indicating that our task-aware expansion strategy and dual-mode initialization method effectively expand the module and alleviate the catastrophic forgetting.

## Environments

The experiment involves two task sequences from MiniHack and Atari, where agents are trained sequentially to achieve continual reinforcement learning.

**MiniHack Environment** We selected 10 tasks from the MiniHack environment to conduct continual learning experiments. The task sequence includes: (1) Room-Random-5x5-v0, (2) Corridor-R2-v0, (3) Room-Dark-5x5-v0, (4) Corridor-R3-v0, (5) Room-Monster-5x5-v0, (6) CorridorBattle-v0, (7) Room-Trap-5x5-v0, (8) HideNSeek-v0, (9) Room-Ultimate-5x5-v0, and (10) HideNSeek-Lava-v0. Fig. S3 presents the randomly initialized observations for each task, and we provide detailed descriptions of the task sequence below.

**1-Room-Random-5x5-v0:** The agent is required to explore a randomly generated room to reach the goal. In each episode, the layout, as well as the initial positions of the

agent and the goal, are randomly initialized.

**2-Corridor-R2-v0:** The agent is required to reach the exit by navigating through two connected corridors, with the positions of the agent and the exit randomized in each episode.

**3-Room-Dark-5x5-v0:** The agent is required to find the goal hidden in a dark room, with both the agent’s starting position and the goal location randomized in each episode.

**4-Corridor-R3-v0:** The agent is required to reach the exit by navigating through three connected corridors, with randomized agent and exit positions in each episode.

**5-Room-Monster-5x5-v0:** The agent is required to reach the goal while avoiding or defeating a monster in the room. The positions of the agent, monster, and goal are randomized in each episode.

**6-CorridorBattle-v0:** The agent is required to fight monsters in the corridor and navigate through it to reach the exit. The positions of the agent, enemies, and exit are randomized in each episode.

**7-Room-Trap-5x5-v0:** The agent is required to reach the goal while avoiding hidden traps scattered in the room. The positions of the agent and the goal are randomized in each episode.

**8-HideNSeek-v0:** The agent is required to find and reach the hidden target while avoiding detection. The positions of the agent and the goal are randomized in each episode.

**9-Room-Ultimate-5x5-v0:** The agent is required to reach the goal while navigating through a room filled with monsters and traps. The positions of the agent, monsters, traps, and the goal are randomized in each episode.

**10-HideNSeek-Lava-v0:** The agent is required to find and reach the hidden target while avoiding dangerous lava

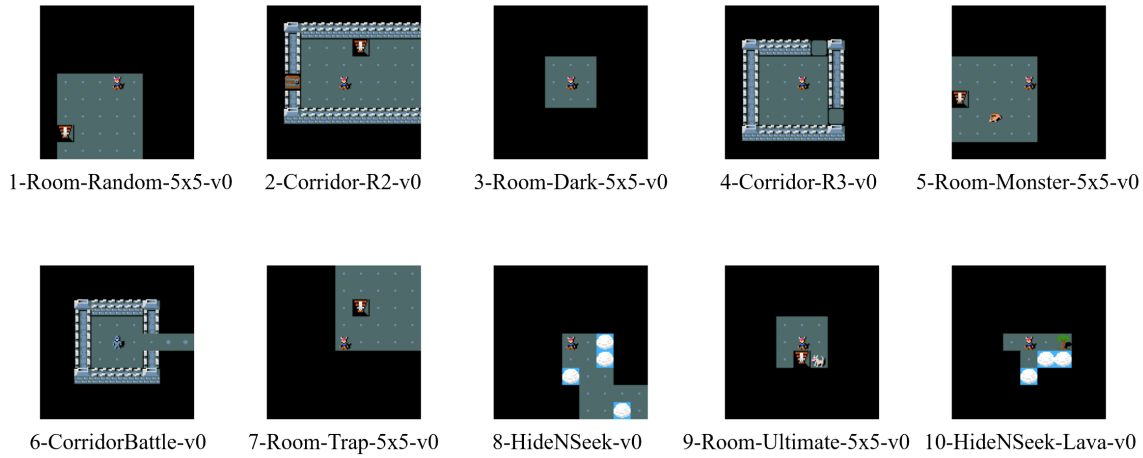


Figure S3: Examples of initial observations for each task in the MiniHack environment.

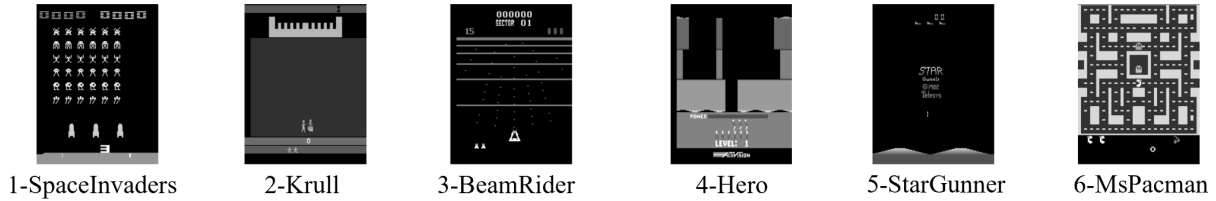


Figure S4: Examples of initial observations for each task in the Atari environment.

hazards. The positions of the agent, target, and lava are randomized in each episode.

**Atari Environment** We selected 6 tasks from the Atari environment to conduct continual learning experiments. The task sequence includes: (1) SpaceInvaders, (2) Krull, (3) BeamRider, (4) Hero, (5) StarGunner, and (6) MsPacMan. Fig. S4 presents the randomly initialized observations for each task, and we provide detailed descriptions of the task sequence below.

**1-SpaceInvaders:** The agent is required to move horizontally to shoot descending aliens. The goal is to eliminate as many aliens as possible while avoiding enemy fire. The positions of the aliens and the agent are randomized in each episode.

**2-Krull:** The agent is required to navigate a landscape to defeat enemies and rescue a captive. The positions of enemies and obstacles are randomized each episode. The agent must avoid hazards while attacking foes to progress.

**3-BeamRider:** The agent is required to shoot down waves of enemy ships while avoiding their attacks. Enemy positions and attack patterns are randomized each episode. The goal is to survive and maximize the score.

**4-Hero:** The agent is required to navigate through a castle to rescue a princess. The environment contains enemies and traps with randomized positions in each episode. The agent must avoid dangers and defeat foes to reach the goal.

**5-StarGunner:** The agent is required to shoot down en-

emy ships while avoiding incoming attacks. Enemy spawn locations and attack patterns are randomized each episode. The goal is to survive and eliminate as many enemies as possible.

**6-MsPacMan:** The agent is required to navigate a maze to eat all pellets while avoiding ghosts. The positions and movements of ghosts are randomized each episode. The goal is to clear the maze without being caught.