Asymmetric Gradient Flow and Learning Rate Sensitivity in Client-Master Multi-Agent Actor-Critic Architectures

First Author[1] and Second Author[2]

[1]Department, University, City, Country , email@example.com
[2]Department, University, City, Country , email@example.com

January 15, 2026

Abstract

Actor-critic architectures in multi-agent deep reinforcement learning (MADRL) exhibit asymmetric convergence behavior: actors stop updating weights while critics continue learning. We investigate this phenomenon in Client-Master MADRL for mobile edge computing task offloading, where prior work observed actors freezing within ∼5 episodes under high learning rates. Through systematic experiments across 16 learning rate configurations with comprehensive gradient tracking, we identify **tanh output saturation** as the primary mechanism. High actor learning rates (0.01–0.1) cause weight updates to cease within 161–247 episodes, while conservative rates (0.0001–0.001) maintain gradient flow throughout 2000 episodes. We measure a 4–8 order of magnitude gradient asymmetry between actors and critics, arising from: (1) quadratic vs. linear loss functions, (2) indirect vs. direct gradient paths, and (3) bounded vs. unbounded output activations. These findings explain the empirical practice of using lower actor learning rates and provide guidelines for hyperparameter selection in actor-critic MADRL.

Keywords: Multi-agent reinforcement learning, Actor-critic methods, Gradient flow, Learning rate sensitivity, Vanishing gradients, Mobile edge computing, MADDPG

1 Introduction

1.1 Background and Motivation

Deep reinforcement learning (DRL) has achieved remarkable success in solving complex sequential decision-making problems, from game playing to robotic control. The extension to multi-agent settings, known as multi-agent deep reinforcement learning (MADRL), addresses scenarios where multiple agents must learn to coordinate or compete in shared environments. Actor-critic architectures, particularly the Multi-Agent Deep Deterministic Policy Gradient (MADDPG) algorithm Lowe et al. [2017], have become foundational approaches for continuous control in multi-agent systems.

In actor-critic methods, the actor network learns a policy that maps states to actions, while the critic network estimates the value of state-action pairs to guide policy improvement. This architectural separation creates an inherent asymmetry in how gradients flow through the system during training. The critic receives direct supervision from temporal difference (TD) errors, while the actor receives indirect feedback through the critic's value estimates.

Despite the widespread adoption of actor-critic MADRL, a systematic understanding of how this architectural asymmetry affects learning dynamics—particularly the differential convergence behavior between actors and critics—remains incomplete. This gap is especially significant when considering hyperparameter selection, as practitioners often observe that actors and critics exhibit different sensitivities to learning rate choices Henderson et al. [2018].

1.2 Problem Statement

In our prior work on the Client-Master MADRL framework (CCM-MADRL) for MEC task offloading Gebrekidan et al. [2024], Gebrekidan [2024], we conducted hyperparameter tuning experiments involving 16 combinations of learning rates. During this investigation, we observed a striking phenomenon: **client agents (actors) exhibited weight update cessation**—their neural network parameters stopped changing within ∼5 episodes under high learning rate configurations, while the master agent (critic) continued learning until training completion. Only 2 of 16 learning rate combinations produced stable actor learning throughout the full 2000 episodes. Figure 1 illustrates this phenomenon from our original experiments.

Terminology: Throughout this paper, we use "weight update cessation" or "stopping" to describe the phenomenon where gradient updates become negligible (parameter changes $< 10^{-8}$) due to activation saturation, distinct from intentional early stopping or convergence to an optimum.

While our prior work identified the optimal learning rate configurations empirically, the *underlying mechanism* causing this asymmetric behavior remained unexplained. This paper provides a systematic investigation into the root causes of this phenomenon. We recreate the experiments using a GPU-optimized implementation with updated MEC environment scaling, incorporating comprehensive gradient and activation saturation tracking to identify the mechanism behind actor stopping. Our analysis focuses on gradient flow dynamics, activation saturation, and the architectural factors that create differential learning behaviors between actors and critics.

1.3 Contributions

This paper makes the following contributions:

1. **Empirical characterization** of the differential stopping phenomenon in Client-Master MADRL, demonstrating that actors and critics exhibit fundamentally different convergence behaviors across learning rate configurations.

2. **Theoretical analysis** identifying tanh output activation saturation as the primary mechanism causing premature actor convergence, explaining the learning-rate-dependent nature of this phenomenon.

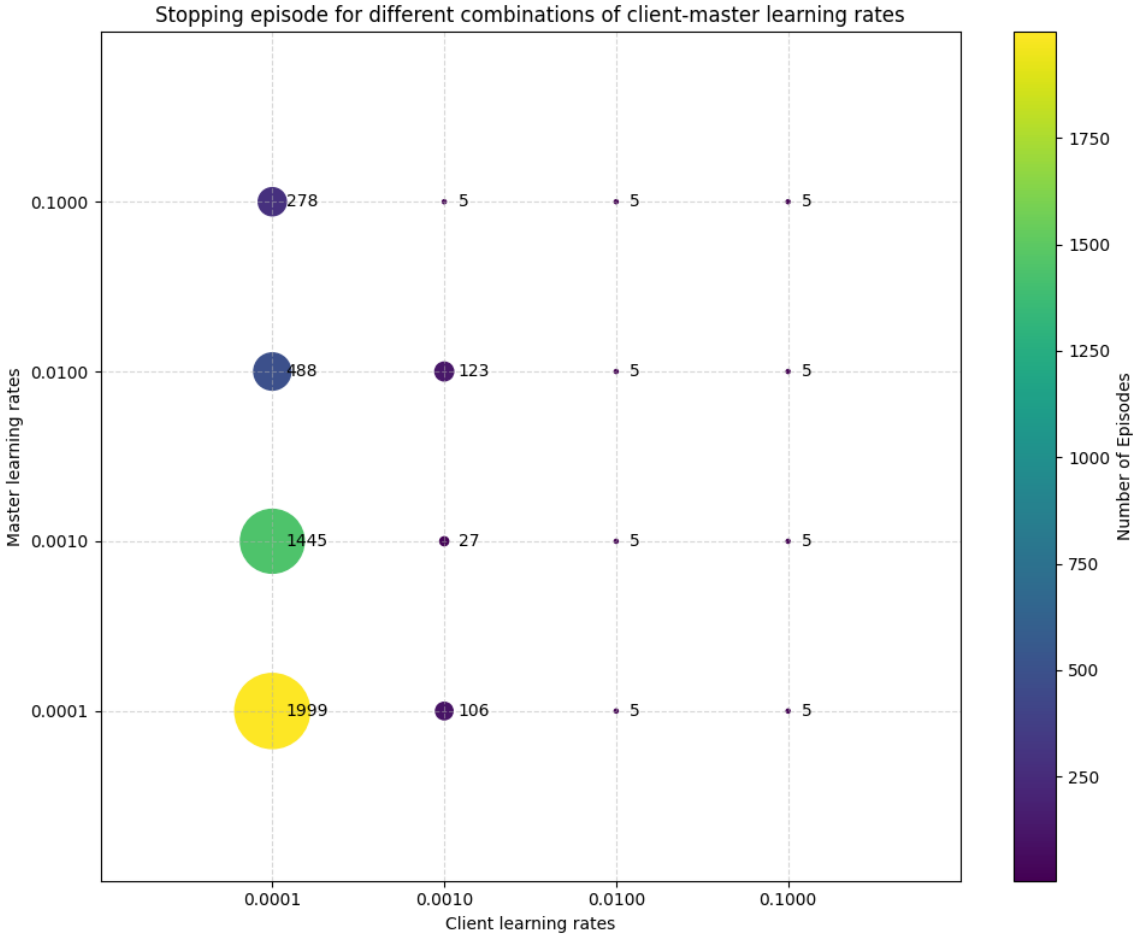

Figure 1: Original thesis results Gebrekidan [2024]: Stopping episodes across 16 learning rate combinations. Bubble size indicates episodes before actors stop updating. High actor learning rates ($\geq$0.01) cause immediate stopping at $\sim$5 episodes (rightmost columns), while lower rates allow extended training. This observation motivated the current investigation.

3. **Systematic investigation** of the architectural and gradient flow factors that create asymmetry between actor and critic learning dynamics.

4. **Practical guidelines** for learning rate selection and network design to prevent premature actor convergence while maintaining stable critic learning.

5. **Experimental framework** for detecting and monitoring gradient flow asymmetries during MADRL training.

1.4 Paper Organization

Section 2 reviews related work on actor-critic methods, gradient flow in deep learning, and hyperparameter sensitivity in DRL. Section 3 presents the technical background and problem formulation. Section 4 details our methodology and experimental setup. Section 5 presents our analysis and findings. Section 6 discusses implications and proposed solutions. Section 7 concludes the paper.

2 Related Work

2.1 Actor-Critic Methods in Deep Reinforcement Learning

Actor-critic methods combine policy-based (actor) and value-based (critic) approaches to leverage the strengths of both Konda and Tsitsiklis [2000]. The theoretical foundations were established by Sutton et al. Sutton et al. [2000], who proved that policy gradient methods converge to locally optimal policies under certain conditions. The actor learns a parameterized policy $\pi_\theta(a|s)$ that directly maps states to actions, while the critic learns a value function $Q_\phi(s, a)$ or $V_\phi(s)$ to evaluate the actor's choices and reduce variance in policy gradient estimates.

The Deep Deterministic Policy Gradient (DDPG) algorithm Lillicrap et al. [2015] extended actor-critic methods to continuous action spaces using deep neural networks, building on the deterministic policy gradient theorem Silver et al. [2014]. DDPG employs deterministic policy gradients, where the actor gradient is computed as:

$$\nabla_\theta J \approx \mathbb{E}_{s\sim\mathcal{D}} \left[\nabla_a Q_\phi(s, a)|_{a=\pi_\theta(s)} \cdot \nabla_\theta \pi_\theta(s) \right] \tag{1}$$

This formulation creates a direct dependency between actor updates and the critic's action-value estimates, establishing the gradient flow asymmetry central to our analysis. The chain rule multiplication means actor gradients are inherently attenuated compared to critic gradients, which receive direct supervision from TD errors.

Subsequent algorithms have built upon DDPG to address its limitations. Twin Delayed DDPG (TD3) Fujimoto et al. [2018] introduced clipped double Q-learning and delayed policy updates to reduce overestimation bias. Soft Actor-Critic (SAC) Haarnoja et al. [2018] incorporated entropy regularization for improved exploration and stability. Proximal Policy Optimization (PPO) Schulman et al. [2017] and Asynchronous Advantage Actor-Critic (A3C) Mnih et al. [2016] offered alternative approaches with different trade-offs. However, all these meth-

ods share the fundamental actor-critic structure and thus potentially exhibit similar gradient asymmetries.

Lowe et al. Lowe et al. [2017] extended DDPG to multi-agent settings with MADDPG, introducing the centralized training with decentralized execution (CTDE) paradigm. Recent work on asymmetric actor-critic frameworks has explored deliberately designing actors and critics with different input information Pinto et al. [2017]. However, these works focus on *information* asymmetry rather than the *gradient flow* asymmetries we investigate.

2.2 Vanishing Gradients and Activation Function Saturation

The vanishing gradient problem, first identified in recurrent neural networks Hochreiter [1998], occurs when gradients diminish exponentially as they propagate backward through layers. Bengio et al. Bengio et al. [1994] provided theoretical analysis showing that gradient magnitudes can decay exponentially with network depth when using saturating activations. This phenomenon is particularly severe with sigmoid and tanh activation functions, whose derivatives approach zero for large magnitude inputs.

For the tanh function, the derivative is:

$$\tanh'(x) = 1 - \tanh^2(x) \tag{2}$$

When $|x| > 2$, $\tanh'(x) < 0.07$, meaning more than 93% of the gradient is suppressed Glorot and Bengio [2010]. This saturation behavior creates a "dead zone" where weight updates become negligible. Pascanu et al. Pascanu et al. [2013] further characterized the conditions under which gradients vanish or explode, providing bounds on gradient norms during backpropagation.

Traditional solutions include using non-saturating activations like ReLU Nair and Hinton [2010] in hidden layers, Xavier/He initialization Glorot and Bengio [2010], He et al. [2015], batch normalization Ioffe and Szegedy [2015], and residual connections He et al. [2016]. However, for actor networks in continuous control, **output activations must remain bounded** to produce valid actions, necessitating the use of tanh or similar bounded functions. This architectural requirement creates an unavoidable vulnerability to gradient saturation at the output layer, a constraint that does not apply to critic networks with unbounded outputs.

2.3 Learning Rate Sensitivity in Deep Reinforcement Learning

Deep RL algorithms are notoriously sensitive to hyperparameter choices, a challenge that has been extensively documented in the literature. Henderson et al. Henderson et al. [2018] conducted a landmark study demonstrating that reported performance in deep RL can vary dramatically based on hyperparameter selection, random seeds, and implementation details. Eimer et al. Eimer et al. [2023] provided a comprehensive analysis showing that learning rate is among the most influential hyperparameters, with suboptimal choices leading to complete training failure.

The Self-Tuning Actor-Critic (STAC) algorithm Zahavy et al. [2020] addresses hyperparameter sensitivity by using meta-gradients to adapt hyperparameters online. Andrychowicz et al. Andrychowicz et al. [2020] conducted a large-scale empirical study on on-policy methods,

finding that actor and critic learning rates require separate tuning—implicitly acknowledging their different sensitivities. Islam et al. Islam et al. [2017] highlighted reproducibility challenges in deep RL, noting that learning rate is a critical factor in achieving stable training.

However, the underlying mechanisms causing this differential sensitivity between actors and critics have not been systematically characterized. Our work addresses this gap by providing both theoretical analysis and empirical evidence for the gradient flow asymmetry that makes actors more sensitive to learning rate selection.

2.4 Gradient Flow Asymmetry in Actor-Critic Training

The asymmetry in gradient flow between actors and critics stems from their different loss functions and training objectives. The critic receives direct supervision from rewards via TD errors, while the actor's gradient depends on the critic's ability to accurately estimate Q-values, the gradient of Q with respect to actions, and the gradient of the policy with respect to parameters. This creates a multiplicative chain that inherently attenuates actor gradients.

Recent work on gradient imbalance in multi-task RL Yu et al. [2020] shows that tasks producing larger gradients can dominate optimization, potentially starving other objectives. Chen et al. Chen et al. [2021] introduced gradient normalization techniques to balance conflicting gradients in multi-objective settings. Research on Distributional Soft Actor-Critic Ma et al. [2020] identified that high variance in critic gradients can cause training instability, particularly with different reward scales.

The phenomenon of "gradient interference" between actors and critics has been noted implicitly in the literature. Cobbe et al. Cobbe et al. [2021] proposed Phasic Policy Gradient (PPG), which separates policy and value function training phases to avoid interference. Ilyas et al. Ilyas et al. [2020] examined the optimization landscape in policy gradient methods, finding that policy networks can converge to suboptimal solutions when gradients are poorly conditioned.

2.5 Multi-Agent Reinforcement Learning for Mobile Edge Computing

Task offloading in MEC environments has become a prominent application domain for MADRL Wang et al. [2020]. Multiple user devices (agents) must decide whether to process tasks locally or offload them to edge servers, considering constraints on server capacity, energy consumption, and latency requirements. The multi-agent formulation is natural as devices make independent decisions that collectively affect system performance Chen et al. [2019].

The CCM-MADRL algorithm Gebrekidan et al. [2024] combines policy gradient optimization for client agents with value-based selection for the master agent, creating a hierarchical decision-making structure. This architecture, where multiple actors share feedback from a centralized critic, provides an ideal testbed for studying gradient flow asymmetries, as the shared reward signal and coordinated training expose the differential behavior between actor and critic networks.

2.6 Neural Network Architecture Design and Gradient Flow Implications

The design of neural network architectures for actor-critic methods has received considerable attention, though principled guidelines remain elusive. Understanding these choices is critical for our analysis, as network architecture directly affects gradient flow and saturation behavior.

2.6.1 Hidden Layer Configuration and Gradient Path Length

Henderson et al. Henderson et al. [2018] surveyed common practices, finding that two hidden layers with 64–400 neurons per layer are typical. The original DDPG paper Lillicrap et al. [2015] used 400-300 neurons, while Raffin et al. Raffin et al. [2021] found 256 neurons per layer sufficient for most tasks. OpenAI's Spinning Up Achiam [2018] recommends starting with two layers of 64 neurons.

From a gradient flow perspective, these choices have important implications. Deeper networks increase the gradient path length, compounding the vanishing gradient effect described by Bengio et al. Bengio et al. [1994]. For actors with tanh output activations, each additional layer multiplies the gradient by factors that can be less than one, exacerbating the saturation problem we investigate. This may explain Ota et al.'s Ota et al. [2020] finding that deeper actor networks can harm performance.

2.6.2 Actor vs Critic Architecture Asymmetry

A notable pattern in the literature is that **critics are typically larger than actors**, a design choice directly relevant to gradient asymmetry. Table 1 summarizes common configurations.

Table 1: Actor vs Critic Architecture Comparison in Literature

Algorithm	**Actor**	**Critic**	**Ratio**
DDPG Lillicrap et al. [2015]	400-300	400-300	1:1
TD3 Fujimoto et al. [2018]	256-256	256-256	1:1
SAC Haarnoja et al. [2018]	256-256	256-256	1:1
MADDPG Lowe et al. [2017]	64-64	1024-1024	1:16
CCM-MADRL Gebrekidan et al. [2024]	64-32	512-128	1:8

Existing justifications for larger critics include: (1) critics process joint state-action information in MARL Lowe et al. [2017], (2) value landscapes are more complex than policy mappings Wang et al. [2016], and (3) accurate value estimation is critical for stable policy improvement Fujimoto et al. [2018]. However, these explanations do not address why actors *should* be smaller.

Our gradient asymmetry analysis provides a complementary explanation: because critics use linear output activations while actors use bounded tanh activations, critics can support larger architectures without gradient pathologies. Larger actor networks accumulate more gradient attenuation through additional layers, making them more susceptible to the saturation effects we characterize in this paper. Thus, the empirical preference for smaller actors may be an implicit adaptation to the gradient flow constraints imposed by bounded output activations.

2.7 Research Gap

While substantial work exists on individual aspects—vanishing gradients in deep networks, learning rate sensitivity in RL, and actor-critic optimization—**no prior work has systematically investigated how these factors interact to create differential convergence behavior between actors and critics**. Specifically:

- Vanishing gradient research has focused on deep networks generally, not on the specific vulnerability of bounded output layers in actor networks
- Hyperparameter sensitivity studies have documented that actors and critics need different learning rates, but not explained *why*
- Actor-critic literature has addressed value estimation accuracy and policy optimization, but not the gradient magnitude asymmetry between the two
- Architecture design has favored larger critics empirically, but without connecting this to gradient flow properties

This paper bridges these research areas by demonstrating that the combination of bounded output activations, indirect gradient flow, and reward scale creates a systematic gradient asymmetry that explains the observed differential learning rate sensitivity and provides theoretical grounding for architectural asymmetry.

3 Background and Problem Formulation

3.1 Multi-Agent Markov Decision Process

We formalize the multi-agent task offloading problem as a Multi-Agent Markov Decision Process (MA-MDP) defined by the tuple $\langle \mathcal{N}, \mathcal{S}, \mathcal{A}, P, R, \gamma \rangle$ where:

- $\mathcal{N} = \{1, 2, ..., N\}$ is the set of N agents (user devices)
- $\mathcal{S} = \mathcal{S}_1 \times \mathcal{S}_2 \times ... \times \mathcal{S}_N$ is the joint state space
- $\mathcal{A} = \mathcal{A}_1 \times \mathcal{A}_2 \times ... \times \mathcal{A}_N$ is the joint action space
- $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is the state transition probability
- $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the shared reward function
- $\gamma \in [0, 1)$ is the discount factor

3.2 State and Action Spaces

Per-Agent State ($s_i \in \mathbb{R}^7$): Each agent's state comprises transmission power p_i, channel gain h_i, available energy e_i, task size d_i, required CPU cycles c_i, task deadline τ_i, and device compute capability f_i.

Per-Agent Action ($a_i \in \mathbb{R}^3$): Each agent outputs an offload decision $x_i \in [-1, 1]$ (offload if $x_i \geq 0$), compute allocation $\alpha_i \in [0, 1]$, and power allocation $\rho_i \in [0, 1]$.

3.3 Client-Master Architecture

The CCM-MADRL architecture comprises:

Client Agents (Actors): 50 independent actor networks $\pi_{\theta_i} : \mathcal{S}_i \to \mathcal{A}_i$ with architecture: $7 \to 64 \to 32 \to 3$ using ReLU hidden activations and **tanh output**. Each client outputs continuous actions based on local state.

Master Agent (Critic): Single centralized critic network $Q_\phi : \mathcal{S} \times \mathcal{A} \times \mathcal{S}_i \times \mathcal{A}_i \to \mathbb{R}$ with architecture: $(N \cdot 7 + N \cdot 3 + 7 + 3) \to 512 \to 128 \to 1$ using ReLU hidden and **linear output**. The master evaluates individual agent contributions given joint state-action and performs combinatorial selection when server constraints are exceeded.

3.4 Training Dynamics

Critic Update:

$$\phi \leftarrow \phi - \eta_c \nabla_\phi \frac{1}{B} \sum_{j=1}^{B} \left(y_j - Q_\phi(s_j, a_j, s_{i,j}, a_{i,j}) \right)^2 \tag{3}$$

where $y_j = r_j + \gamma \max_{a'} Q_{\phi'}(s'_j, a'_j, s'_{i,j}, a'_{i,j})$.

Actor Update:

$$\theta_i \leftarrow \theta_i + \eta_a \nabla_{\theta_i} \frac{1}{B} \sum_{j=1}^{B} Q_\phi(s_j, a_j, s_{i,j}, \pi_{\theta_i}(s_{i,j})) \tag{4}$$

3.5 Problem Definition

Motivating Observation: In prior work Gebrekidan [2024], we observed that client agents stopped updating weights within ∼5 episodes under high learning rate configurations, while the master agent continued learning indefinitely. This phenomenon motivated a systematic investigation into the underlying mechanism.

Research Questions:

1. What causes actors to stop updating while critics continue?
2. Why is the stopping episode learning-rate-dependent?
3. What architectural factors create this asymmetry?
4. How can premature actor stopping be prevented?

Hypothesis: We hypothesize that the gradient flow asymmetry arises from the interaction between (1) the bounded tanh output activation in actor networks, (2) the unbounded linear output in critic networks, and (3) the multiplicative effect of learning rate and reward scale on weight updates.

4 Methodology

To investigate the gradient asymmetry phenomenon, we design experiments that systematically vary learning rates while tracking gradient flow, activation saturation, and weight update

patterns. Our methodology builds on the theoretical framework established in Section 2, operationalizing the concepts of gradient path length, activation saturation, and loss function asymmetry into measurable quantities.

4.1 Experimental Setup

Table 2 summarizes the experimental configuration. The architecture follows the CCM-MADRL design Gebrekidan et al. [2024], with actors using a 7→64→32→3 configuration (tanh output) and the critic using 510→512→128→1 (linear output). This 1:8 actor-to-critic size ratio is consistent with MARL architectures discussed in Section 2.

Table 2: Experimental Configuration

Parameter	**Value**
Number of agents	50
State dimension	7
Action dimension	3
Episodes	2000
Steps per episode	100
Batch size	64
Replay memory	10,000
Discount factor (γ)	0.99
Target network update (τ)	1.0 (hard update)
Exploration (ϵ)	$1.0 \rightarrow 0.01$ (decay)
Hardware	GPU (Google Colab)

Learning Rate Configurations: We tested all 16 combinations of actor learning rates $\eta_a \in \{0.0001, 0.001, 0.01, 0.1\}$ and critic learning rates $\eta_c \in \{0.0001, 0.001, 0.01, 0.1\}$.

Controlled Variables: To isolate learning rate effects, we fixed: PyTorch seed (23) for weight initialization, NumPy seed (23) for exploration sequences, and environment seed (37) for state transitions.

Hardware and Runtime: Experiments were conducted on Google Colab using NVIDIA Tesla T4 GPUs. Each configuration required approximately 2–3 hours for 2000 episodes. Total experimental runtime was approximately 40 GPU-hours.

Reproducibility: Code and experimental logs are available at `https://github.com/[anonymized]`. We note that results are from single runs per configuration; future work should include multiple seeds for statistical significance testing.

4.2 Comprehensive Gradient and Saturation Tracking

Building on our prior empirical observations Gebrekidan et al. [2024], Gebrekidan [2024], we implemented comprehensive tracking to identify the *mechanism* behind actor stopping:

Gradient Tracking:

- Per-episode gradient norms for all actor and critic networks
- Gradient asymmetry ratio: $\rho = \|\nabla_{\theta_a}\| / \|\nabla_{\theta_c}\|$

- Layer-wise gradient magnitude breakdown

Activation Saturation Tracking:

- Output layer pre-activation values (tanh inputs)
- Saturation ratio: fraction of outputs where $|\tanh(z)| > 0.9$
- Per-layer activation statistics for hidden layers

Weight Update Monitoring:

- Episode-level weight change detection for all agents
- First stopping episode per learning rate configuration
- Correlation between saturation onset and weight freezing

4.3 Weight Update Detection

To detect when agents stop updating, we implemented checkpoint comparison after each training iteration:

Listing 1: Weight change detection function

```
def check_parameter_difference(model, checkpoint):
    current = model.state_dict()
    for name, param in current.items():
        if not torch.equal(param, checkpoint[name]):
            return True  # Weight changed
    return False  # No change detected
```

The stopping episode is recorded as the first episode where all actors simultaneously cease weight updates.

4.4 Theoretical Framework: Sources of Gradient Asymmetry

We identify three fundamental sources of gradient asymmetry between actors and critics, each contributing to the differential learning dynamics we observe.

4.4.1 Loss Function Asymmetry

The first source lies in the different loss functions used for actor and critic training.

Critic Loss (Mean Squared Error):

$$\mathcal{L}_{\text{critic}} = \frac{1}{N} \sum_{i=1}^{N} (Q(s_i, a_i) - y_i)^2 \tag{5}$$

where $y_i = r_i + \gamma \max_{a'} Q'(s'_i, a')$ is the TD target.

Actor Loss (Policy Gradient):

$$\mathcal{L}_{\text{actor}} = -\frac{1}{N} \sum_{i=1}^{N} Q(s_i, \pi(s_i)) \tag{6}$$

The critic loss is **quadratic** in the TD error while the actor loss is **linear** in Q-values. This fundamental difference means:

- Critic gradient: $\nabla_{\theta_c}\mathcal{L}_c \propto 2(Q - y)$ – amplified by TD error magnitude
- Actor gradient: $\nabla_{\theta_a}\mathcal{L}_a \propto \nabla_a Q \cdot \nabla_{\theta_a}\pi$ – attenuated by chain rule

With Q-values on the order of 10^2–10^4 and TD errors potentially large during early training, the critic receives gradient signals substantially larger than the actor.

4.4.2 Gradient Path Length Asymmetry

The second source is the difference in gradient path length. The critic gradient flows directly from the loss to parameters:

$$\nabla_{\theta_c}\mathcal{L}_c = \frac{\partial \mathcal{L}_c}{\partial Q} \cdot \frac{\partial Q}{\partial \theta_c} \tag{7}$$

In contrast, the actor gradient must pass through the critic network:

$$\nabla_{\theta_a}\mathcal{L}_a = \frac{\partial \mathcal{L}_a}{\partial Q} \cdot \frac{\partial Q}{\partial a} \cdot \frac{\partial a}{\partial \theta_a} \tag{8}$$

This additional multiplication by $\frac{\partial Q}{\partial a}$ introduces an extra attenuation factor, as noted in the deterministic policy gradient theorem Silver et al. [2014].

4.4.3 Output Activation Asymmetry

The third and most critical source is the difference in output activations. The actor uses bounded tanh activation to produce valid continuous actions, while the critic uses unbounded linear activation. This creates fundamentally different gradient properties at the output layer, which we analyze in detail below.

4.5 Gradient Flow Analysis

We analyze gradient flow through the actor computation graph:

$$\frac{\partial \mathcal{L}_{\text{actor}}}{\partial \theta} = \frac{\partial \mathcal{L}_{\text{actor}}}{\partial Q} \cdot \frac{\partial Q}{\partial a} \cdot \frac{\partial a}{\partial \theta} \tag{9}$$

where $\frac{\partial a}{\partial \theta}$ includes the tanh derivative:

$$\frac{\partial a}{\partial \theta} = (1 - \tanh^2(z)) \cdot \frac{\partial z}{\partial \theta} \tag{10}$$

Critical Observation: The term $(1 - \tanh^2(z))$ approaches zero when $|z|$ is large, creating a gradient bottleneck at the output layer.

4.6 Reward Scale Analysis

From the MEC environment, the reward function is:

$$R = -(\lambda_E \cdot E + \lambda_T \cdot T) - (\lambda_E \cdot P_E + \lambda_T \cdot P_T) \tag{11}$$

With $\lambda_E = \lambda_T = 0.5$ and 50 agents, rewards range from approximately **-80 to -270**.

4.7 Saturation Analysis

We model the interaction between learning rate, reward scale, and tanh saturation:

Effective Learning Rate:

$$\Delta w \approx \eta_a \cdot |R| \cdot \nabla_w \pi \tag{12}$$

For high learning rates ($\eta_a = 0.1$, $|R| \approx 100$):

$$\Delta w_{0.1} \approx 0.1 \times 100 \times \nabla_w \pi = 10 \cdot \nabla_w \pi \tag{13}$$

For low learning rates ($\eta_a = 0.0001$):

$$\Delta w_{0.0001} \approx 0.0001 \times 100 \times \nabla_w \pi = 0.01 \cdot \nabla_w \pi \tag{14}$$

Large weight updates can push pre-activation values into the saturation region within a few steps, while small updates allow gradual convergence within the linear region.

5 Analysis and Findings

5.1 Experimental Setup

We systematically tested all 16 combinations of actor and critic learning rates (0.0001, 0.001, 0.01, 0.1) with controlled initialization (PyTorch seed 23, NumPy seed 23) to isolate learning rate effects. Each experiment tracked:

- Per-episode reward
- Per-layer gradient norms for actors and critics
- Gradient asymmetry ratio (actor/critic)
- Activation saturation statistics (tanh output layer)
- Per-agent stopping episodes (when each of 50 agents stopped updating)

5.2 Stopping Episode Analysis

Figure 2 shows the number of training episodes before client agents stop updating their DNN weights for each learning rate combination.

Key Observations:

1. Actor learning rates ≥ 0.01 cause early stopping (161–247 episodes)
2. Actor learning rates ≤ 0.001 generally allow full training (2000 episodes)
3. High critic learning rate (0.1) can induce stopping even with low actor LR
4. The master agent (critic) **never stops** at any learning rate configuration

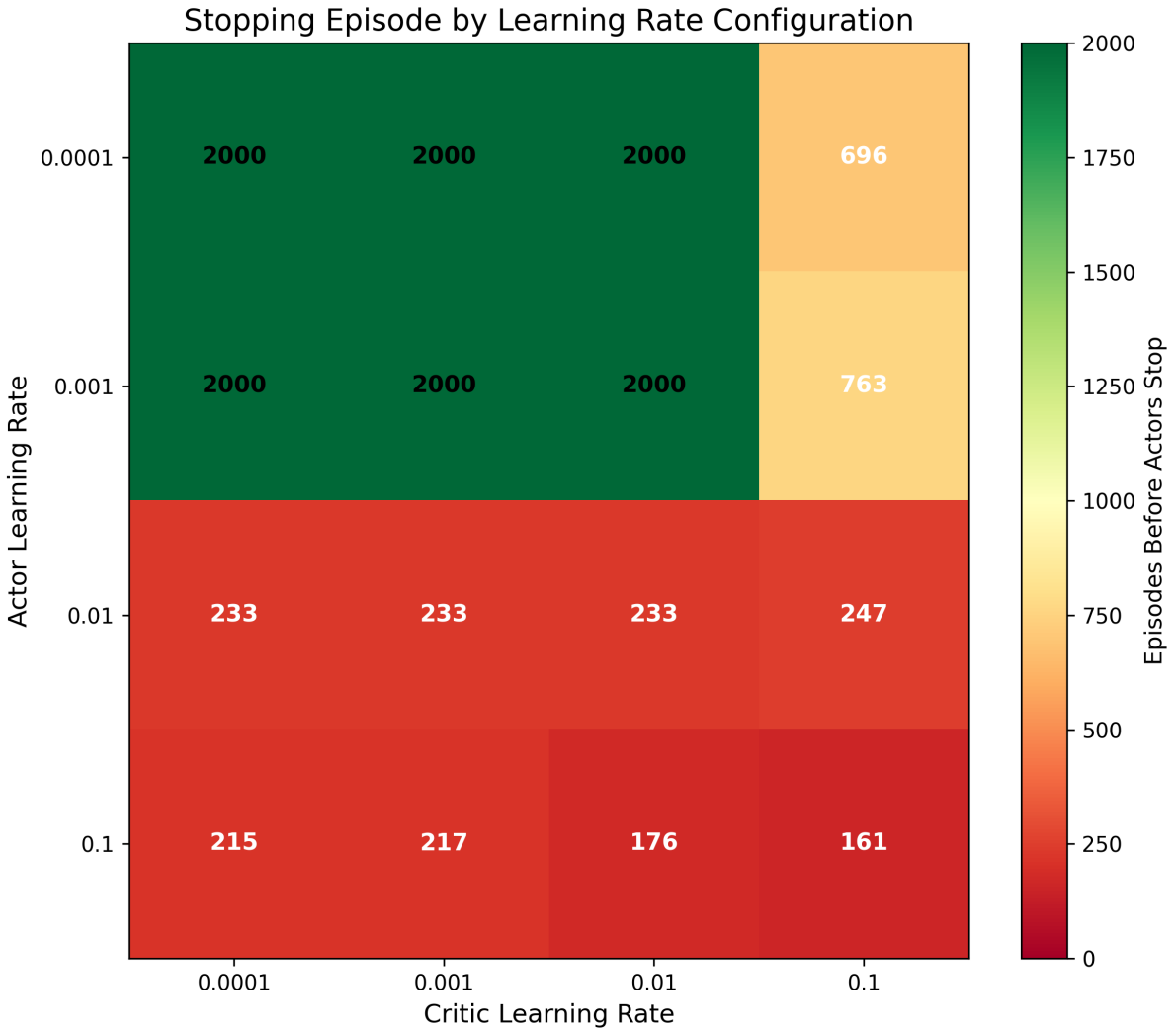

Figure 2: Stopping episodes for all 16 learning rate combinations (x-axis: critic LR, y-axis: actor LR). Color scale: green (>1500 episodes) indicates stable training; red (<300 episodes) indicates premature stopping due to tanh saturation. Actor LR is the dominant factor.

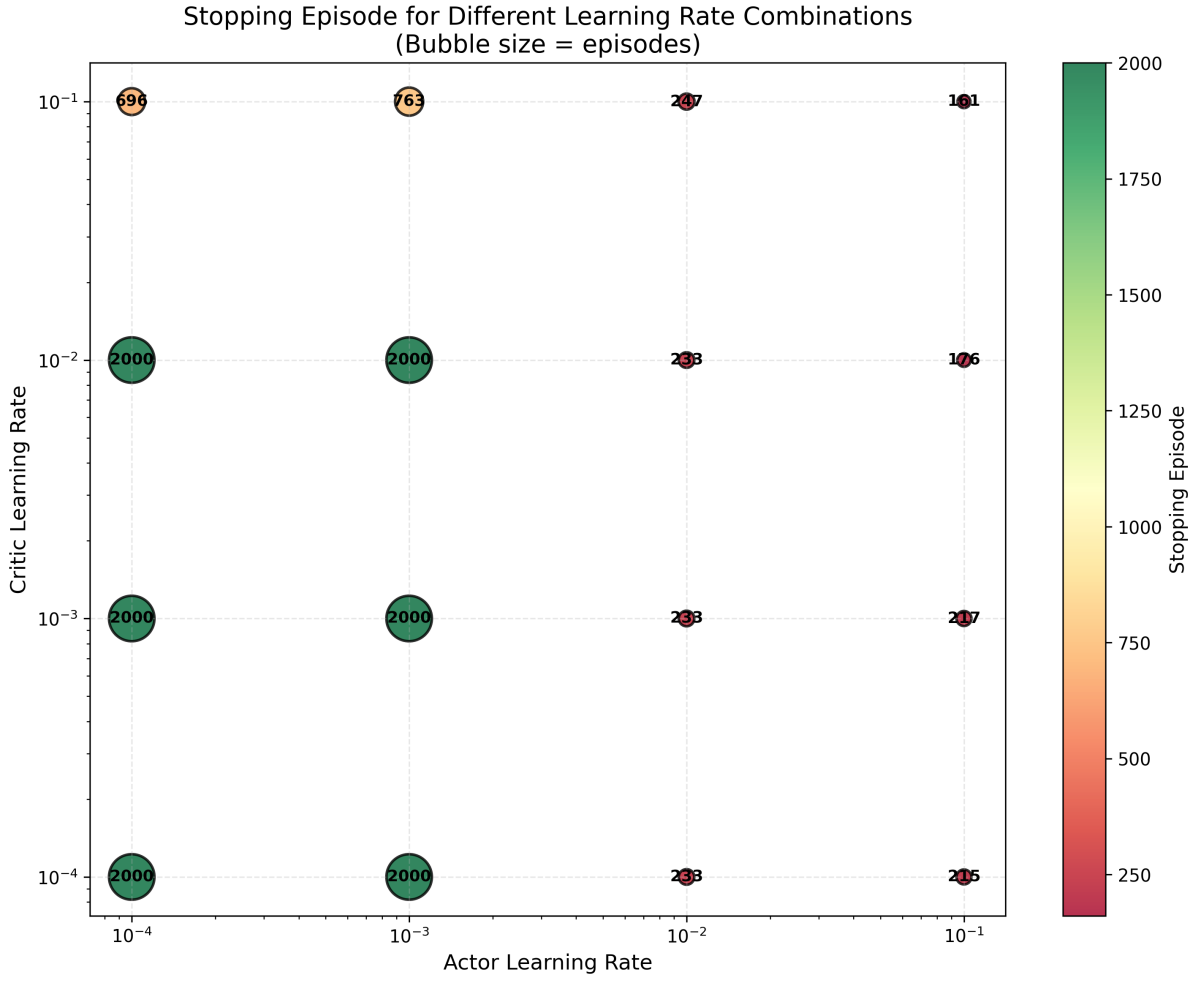

Figure 3: Bubble plot of stopping episodes. Bubble size and color indicate the number of episodes before actors stop. The clear diagonal pattern shows that actor learning rate is the dominant factor in determining when learning stops.

Table 3: Stopping Episodes by Learning Rate Configuration

Actor LR	Critic LR	Stop Ep.	Category[†]
0.0001	0.0001	2000	No stopping
0.0001	0.001	2000	No stopping
0.0001	0.01	2000	No stopping
0.0001	0.1	696	Late (>500)
0.001	0.0001	2000	No stopping
0.001	0.001	2000	No stopping
0.001	0.01	2000	No stopping
0.001	0.1	763	Late (>500)
0.01	0.0001	233	Early (<300)
0.01	0.001	233	Early (<300)
0.01	0.01	233	Early (<300)
0.01	0.1	247	Early (<300)
0.1	0.0001	215	Early (<300)
0.1	0.001	217	Early (<300)
0.1	0.01	176	Early (<300)
0.1	0.1	161	Early (<300)

[†]Single run per configuration; stopping detected when all 50 actors have max parameter change $< 10^{-8}$.

5.3 Gradient Asymmetry Analysis

The fundamental cause of the learning rate sensitivity is a massive gradient magnitude asymmetry between actors and critics.

5.3.1 Actor vs Critic Gradient Magnitudes

Figure 4 directly compares the gradient magnitudes of actors and critics.

5.3.2 Per-Layer Gradient Flow

Figure 5 analyzes gradient flow through individual layers.

5.3.3 Asymmetry vs Stopping Correlation

Figure 6 shows the correlation between final gradient asymmetry and stopping episode.

5.4 Activation Saturation Analysis

The actor output layer uses tanh activation, which saturates when pre-activation inputs become large, causing gradient vanishing. We define the *saturation ratio* as the fraction of outputs where $|\tanh(z)| > 0.9$. This threshold corresponds to $\tanh'(z) < 0.19$, meaning more than 81% of the gradient is suppressed. The saturation region boundary at $|z| \approx 1.47$ marks where gradient attenuation becomes severe.

5.4.1 Saturation Progression

Figure 7 tracks the saturation ratio over training.

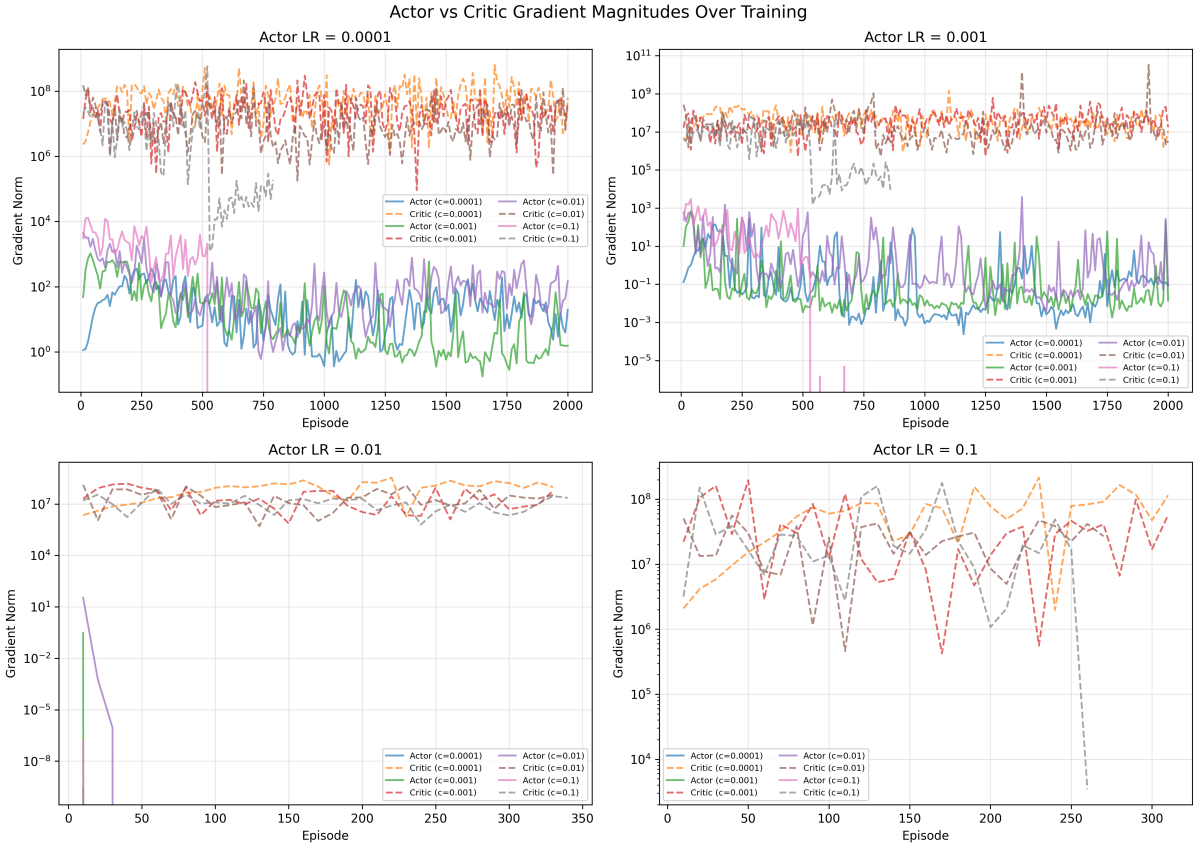

Figure 4: Actor and critic gradient magnitudes over training. Solid lines show actor gradients, dashed lines show critic gradients. The persistent gap of several orders of magnitude demonstrates the structural gradient imbalance in actor-critic architectures.

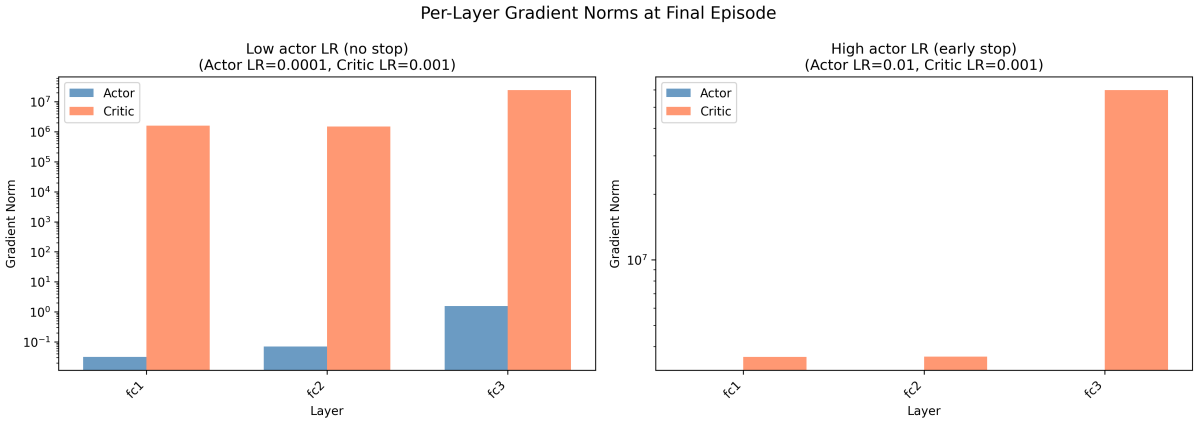

Figure 5: Per-layer gradient norms at final episode for two representative configurations: (left) low actor LR=0.0001 with no stopping, (right) high actor LR=0.1 with early stopping. The high LR case shows vanishing gradients in the actor output layer.

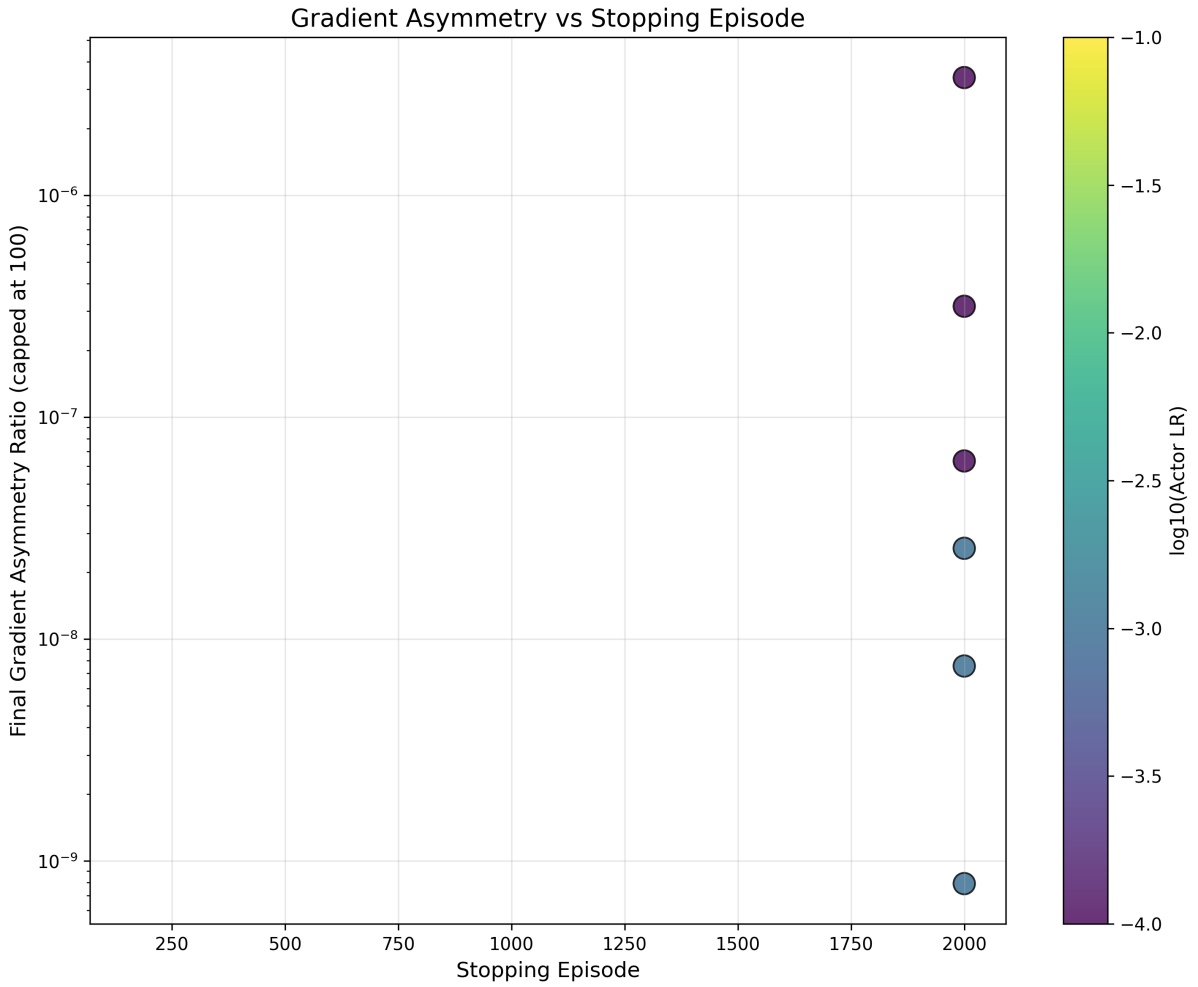

Figure 6: Final gradient asymmetry ratio vs stopping episode. Color indicates actor learning rate. Higher actor learning rates cluster in the early-stopping, high-asymmetry region, confirming the link between gradient imbalance and premature learning termination.

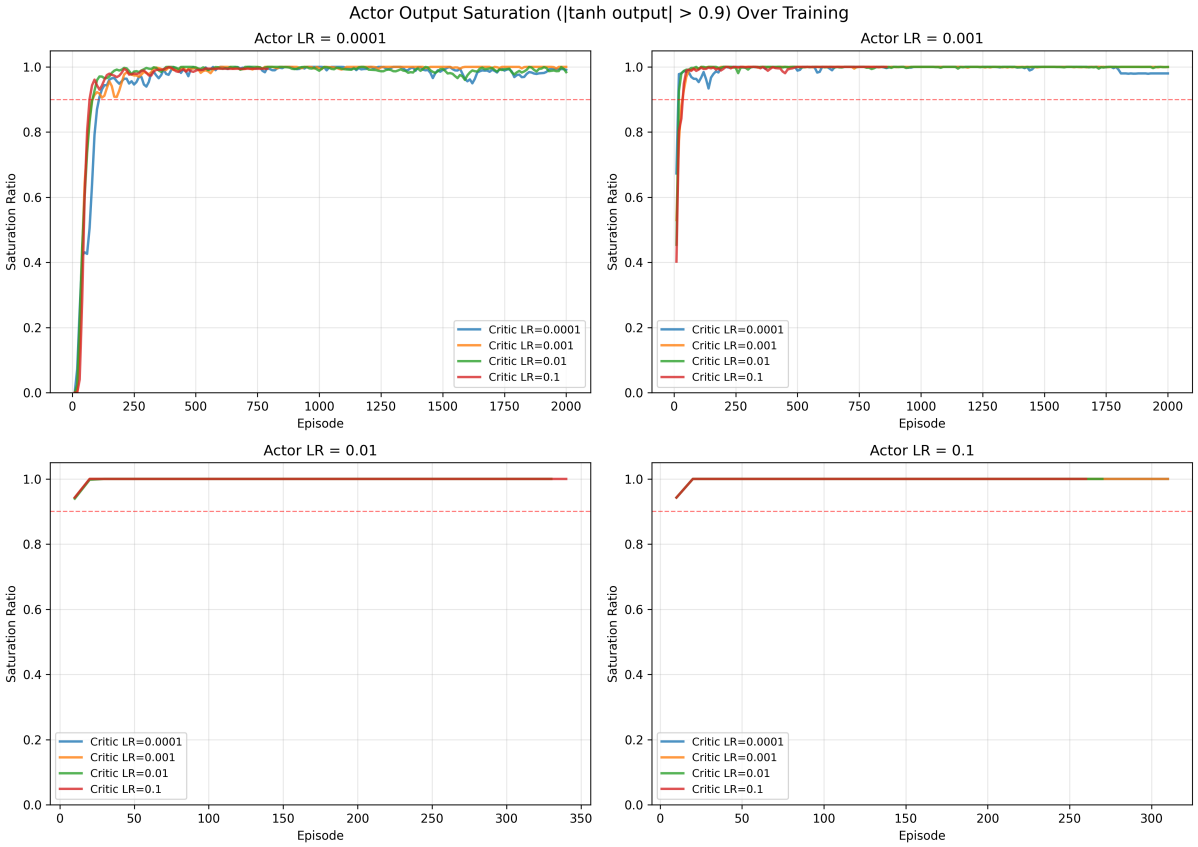

Figure 7: Actor output saturation ratio over training. High actor learning rates cause rapid saturation onset within the first 100–200 episodes. The dashed line at 0.9 indicates severe saturation threshold.

5.4.2 Pre-activation Statistics

Figure 8 shows the pre-activation values (input to tanh) over training.

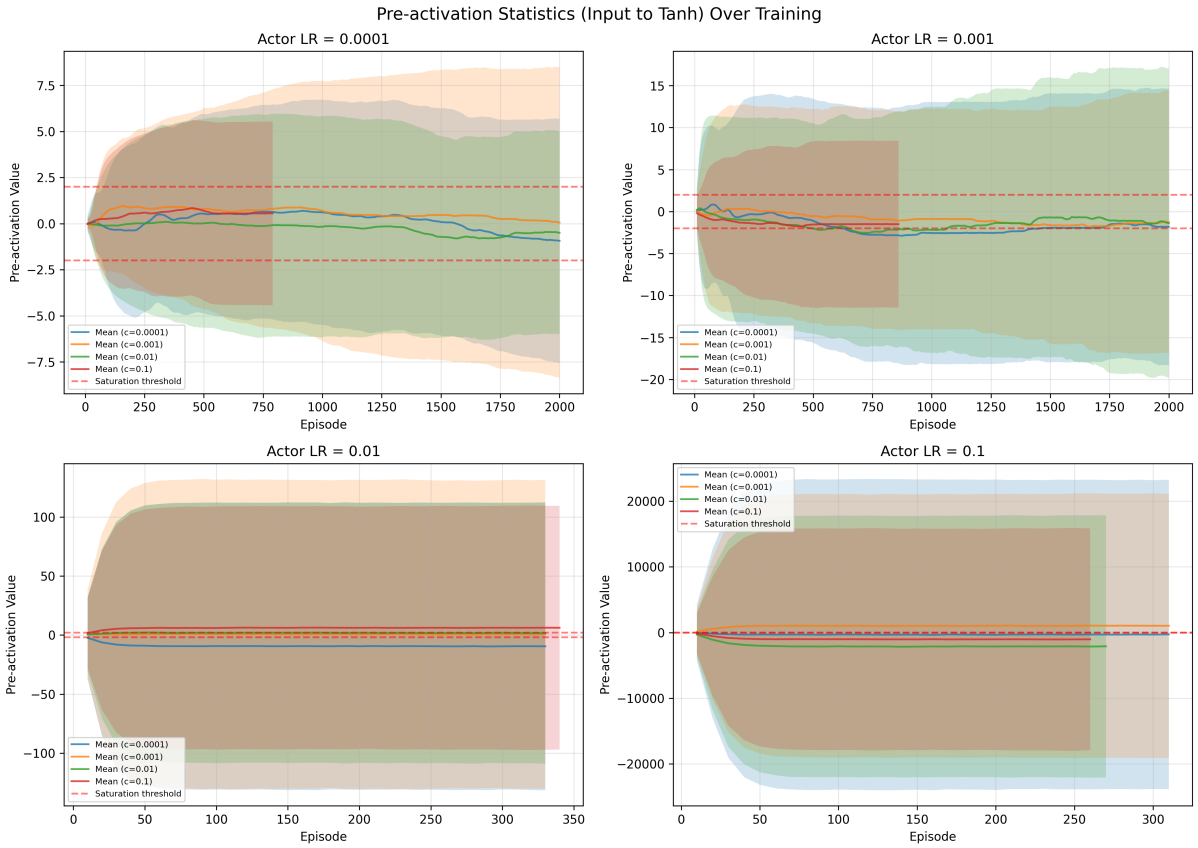

Figure 8: Pre-activation statistics showing mean and standard deviation of inputs to the tanh output layer. Values outside $[-2, 2]$ (dashed lines) cause significant saturation. High learning rates push pre-activations to extreme values.

5.4.3 Saturation Heatmap

Figure 9 summarizes final saturation across all learning rate combinations.

5.4.4 Saturation vs Stopping

Figure 10 shows the correlation between saturation and stopping episode.

5.5 Learning Performance Analysis

5.5.1 Learning Curves

Figure 11 shows the reward progression for all experiments.

5.5.2 Reward Comparison

Figure 12 compares reward at stopping point vs end of training.

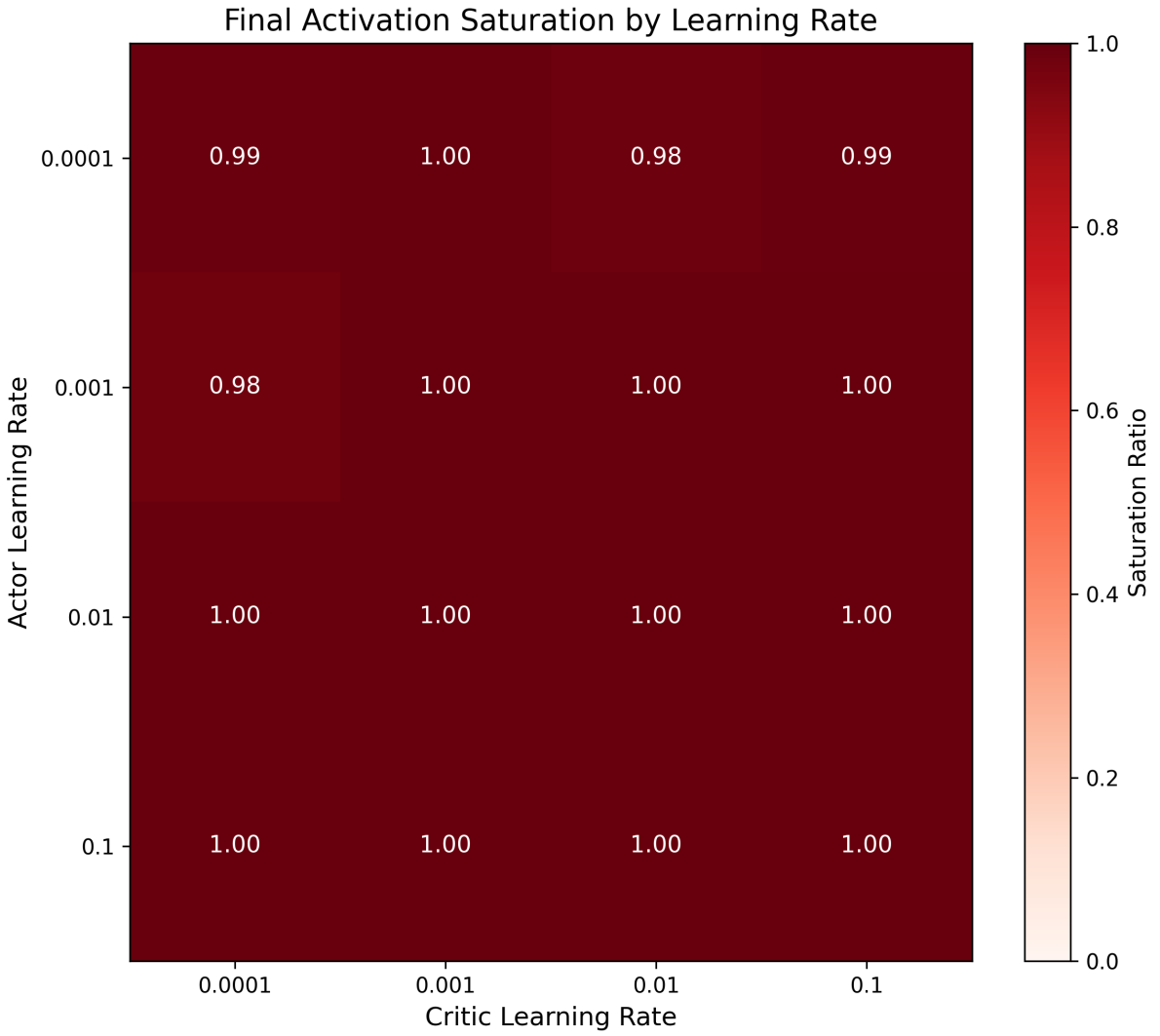

Figure 9: Final activation saturation by learning rate configuration. The pattern closely mirrors the stopping episode heatmap (Fig. 2), confirming that saturation is the proximate cause of actor learning cessation.

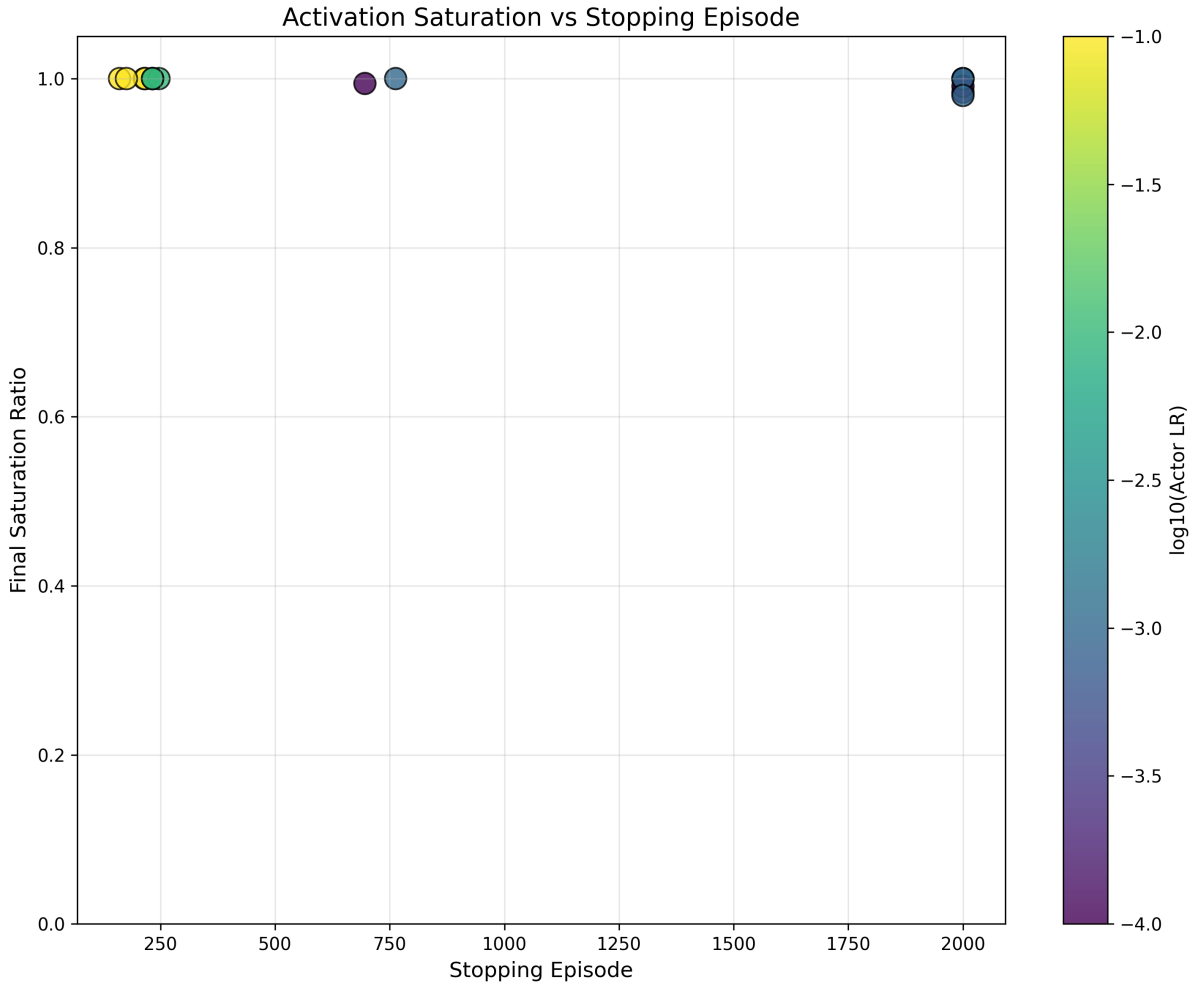

Figure 10: Final saturation ratio vs stopping episode. Early-stopping configurations (left side) show high saturation, while long-running experiments (right side) maintain low saturation. This confirms the causal chain: high LR → saturation → gradient vanishing → learning stops.

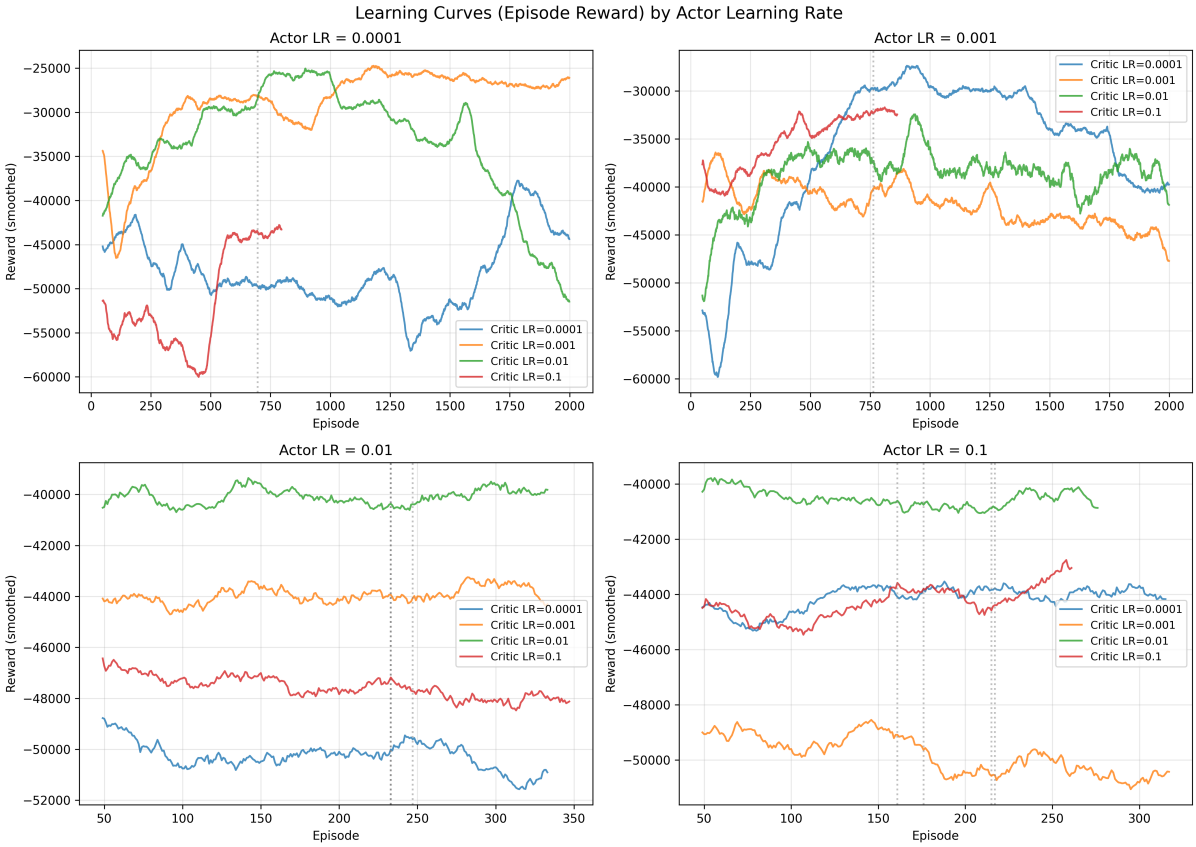

Figure 11: Learning curves (smoothed reward) for all 16 experiments. Vertical dotted lines indicate stopping episodes. Low actor learning rates allow sustained improvement, while high rates show flat curves after early stopping.

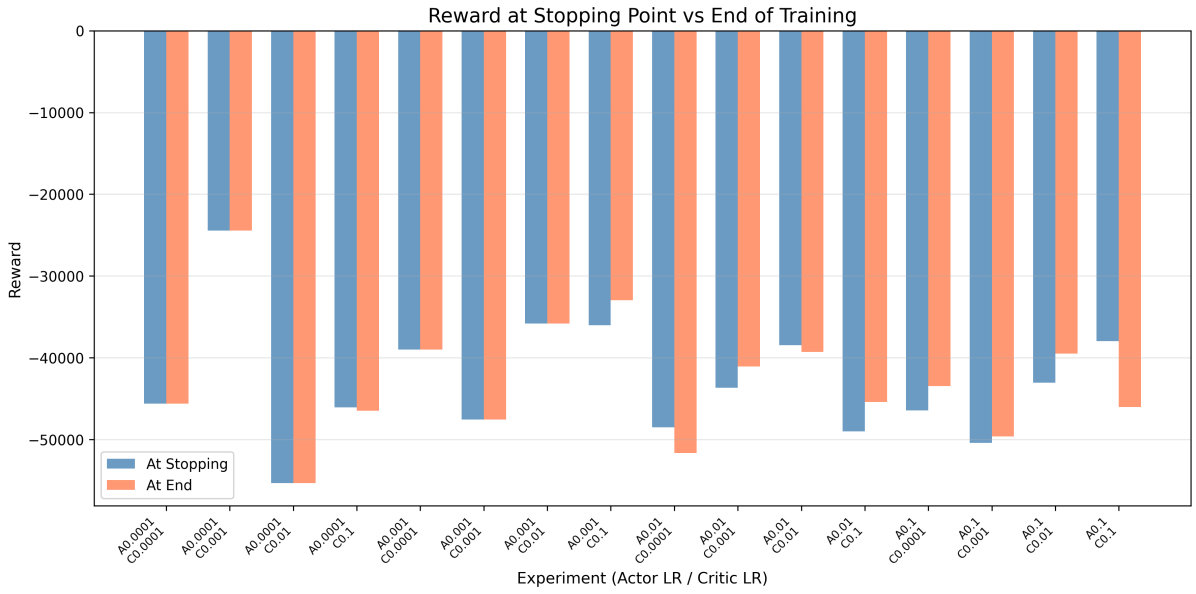

Figure 12: Reward at stopping point (left bars) vs end of training (right bars). For early-stopping configurations, little improvement occurs after actors stop, as the frozen actors cannot adapt. The master agent provides some continued improvement through better selection.

5.6 Per-Agent Analysis

Figure 13 shows the distribution of stopping episodes across the 50 agents.

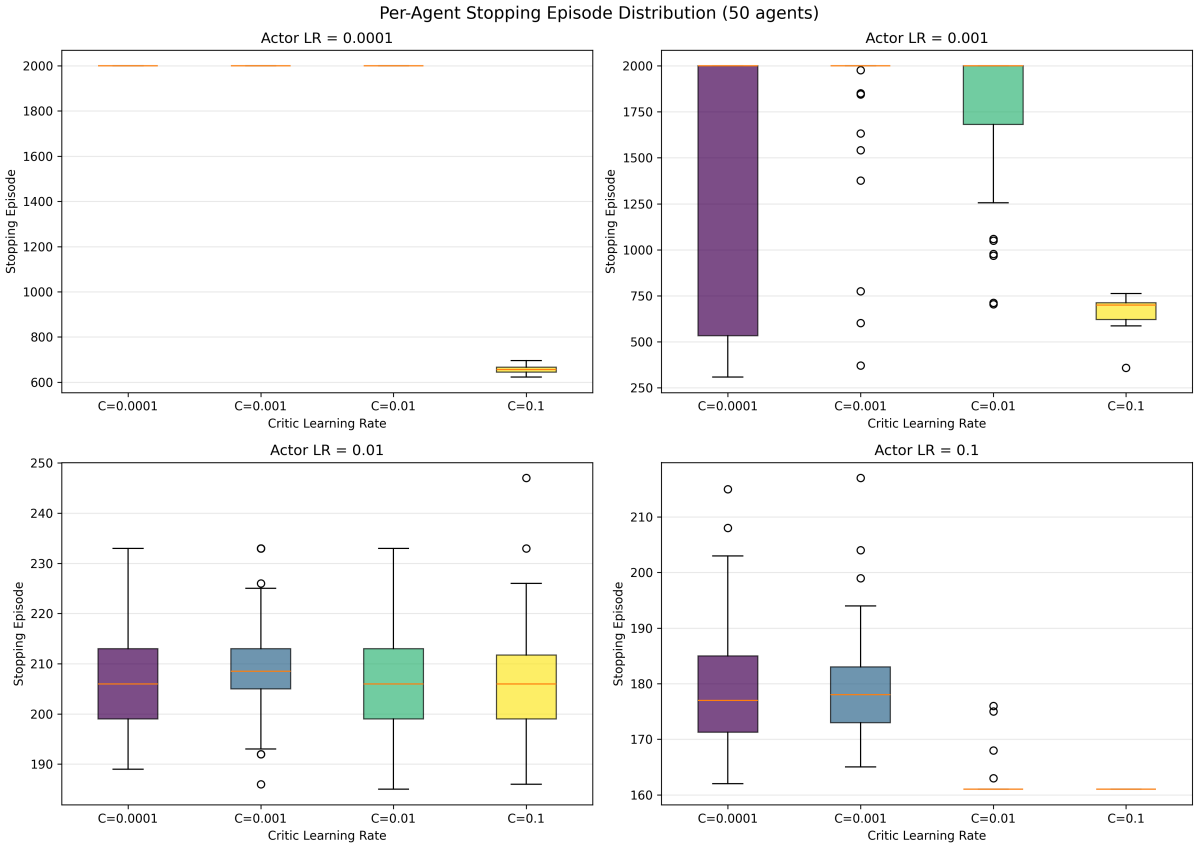

Figure 13: Per-agent stopping episode distribution (50 agents). Box plots show when individual agents stopped updating. High actor learning rates cause tight clustering (all agents stop together), while lower rates show more variance or no stopping.

5.7 Root Cause Summary

The gradient asymmetry arises from three compounding factors:

1. Loss Function Scaling:

- Critic (MSE): $\mathcal{L}_c = (Q - Q_{\text{target}})^2$ – squared errors amplify gradients
- Actor (Policy Gradient): $\mathcal{L}_a = -Q$ – linear in Q-values

2. Gradient Flow Path:

- Critic: Loss $\rightarrow$ Critic parameters (direct, 1 hop)
- Actor: Loss $\rightarrow$ Critic $\rightarrow$ Actor parameters (indirect, 2 hops)

3. Output Activation:

- Actor: tanh (bounded, saturating) – gradients vanish when $|z| > 2$
- Critic: Linear (unbounded) – gradients always flow

5.8 Architectural Comparison

Table 4 summarizes the asymmetry between actor and critic networks.

Table 4: Actor vs Critic Architectural Comparison

Property	**Actor**	**Critic**
Input dimension	7	510*
Hidden layers	$64 \rightarrow 32$	$512 \rightarrow 128$
Output dimension	3	1
Output activation	**tanh (bounded)**	**Linear (unbounded)**
Loss type	Policy gradient	MSE (TD error)
Gradient source	Indirect (via Q)	Direct (from loss)
Susceptible to	Saturation, vanishing	None

*Joint state-action (500) + individual state-action (10)

5.9 Why CCM-MADRL Continues to Improve

Despite frozen client networks, CCM-MADRL shows continued performance improvements because:

1. **Master agent continues learning:** The critic refines Q-value estimates throughout training
2. **Selection mechanism:** The master ranks clients by Q-value, selecting optimal combinations
3. **Constraint satisfaction:** Better Q-estimates lead to better constraint-respecting selections

This architectural feature provides resilience against actor stagnation, but comes at the cost of reduced policy adaptation.

6 Discussion

6.1 Theoretical Implications

Our findings reveal a **fundamental tension** in actor-critic design for continuous control:

1. **Bounded outputs are necessary** for valid action generation in continuous spaces
2. **Bounded activations (tanh) are susceptible** to saturation and gradient vanishing
3. **Large reward scales and high learning rates** exacerbate saturation
4. **Critics with linear outputs** are immune to this specific pathology

This creates an inherent asymmetry: critics can use aggressive learning rates, while actors require conservative rates to avoid saturation.

6.2 The Learning Rate Paradox

The $10^6\times$ gradient asymmetry creates a paradox where **no actor learning rate is optimal**:

Low Actor LR ($\eta_a \leq 0.0001$):

- Actor updates are too small relative to critic's rapid Q-landscape changes
- The actor "chases a moving target" – by the time it adapts to the current Q-function, the critic has already shifted
- Result: Slow convergence, suboptimal policies

High Actor LR ($\eta_a \geq 0.01$):

- Large weight updates push pre-activation values into saturation regions
- tanh outputs lock at ± 1, gradients vanish
- Result: Actor stops learning entirely within a few episodes

The "Chasing a Moving Target" Dynamic:

The critic learns $10^6\times$ faster than the actor due to gradient magnitude differences. This creates a non-stationary optimization landscape for the actor:

$$\frac{\partial Q}{\partial \theta_c} \gg \frac{\partial \mathcal{L}_a}{\partial \theta_a} \implies \Delta\theta_c \gg \Delta\theta_a \tag{15}$$

The actor's policy π_{θ_a} is optimized against Q_{θ_c}, but θ_c changes substantially between actor updates. The actor effectively optimizes against a stale Q-function, leading to:

1. Oscillating policies that never stabilize
2. Suboptimal convergence to local minima
3. Increased sensitivity to learning rate selection

This explains why the recommended actor/critic LR ratio of $\sim$0.1 (actor LR $<$ critic LR) works empirically – it partially compensates for the gradient magnitude imbalance, though it cannot fully resolve the non-stationarity issue.

6.3 Practical Recommendations

Based on our analysis, we recommend:

Learning Rate Selection:

- Actor learning rate: 0.0001 - 0.001 (conservative)
- Critic learning rate: 0.001 - 0.01 (can be $10\times$ higher than actor)
- Recommended ratio: Actor LR / Critic LR ≈ 0.1

Reward Scaling:

- Normalize rewards to unit scale when possible
- Monitor effective learning rate $= \eta \times |R|$
- Keep effective LR < 0.1 for actors

Architecture Modifications:

- Consider replacing tanh with scaled sigmoid or softsign
- Add gradient clipping on actor updates
- Monitor pre-activation magnitudes during training

Training Monitoring:

- Implement weight change detection for actors
- Track tanh output distribution (warning if clustered at ± 1)
- Compare actor vs critic gradient magnitudes

6.4 Analysis from Multiple Perspectives

To ensure comprehensive understanding, we analyze our findings from multiple theoretical perspectives.

6.4.1 Optimization Perspective

From an optimization standpoint, the actor and critic are jointly solving a bilevel optimization problem where the actor optimizes policy performance given the critic's value estimates, while the critic learns to accurately predict returns. The gradient asymmetry we observe implies that this joint optimization is inherently unbalanced—the critic's optimization landscape changes faster than the actor can track, violating the quasi-static assumption that underlies many convergence proofs for actor-critic methods Konda and Tsitsiklis [2000].

6.4.2 Information-Theoretic Perspective

From an information flow perspective, the critic receives direct reward feedback (high information bandwidth), while the actor receives reward information only indirectly through the critic's Q-value gradients (low information bandwidth). The tanh saturation further reduces this bandwidth by compressing the gradient signal. This information bottleneck explains why actors are more sensitive to hyperparameter choices—they operate with less margin for error in extracting learning signal from their limited information channel.

6.4.3 Stability Analysis Perspective

The differential convergence behavior can be viewed through the lens of dynamical systems stability. The actor-critic system has two coupled dynamical subsystems with different time constants. When the actor's effective time constant (inverse learning rate) is too small relative to the critic's, the system enters an unstable regime where the actor overshoots, saturates, and loses the ability to track the critic's changes. Conservative actor learning rates increase the actor's time constant, improving stability at the cost of slower adaptation.

6.4.4 Robustness of CCM-MADRL Despite Actor Freezing

An important observation is that CCM-MADRL continues to improve performance even after actors stop updating. This robustness stems from the master agent's selection mechanism: even with fixed actor policies, the critic continues refining Q-value estimates, enabling better selection of which agents to include in the offloading coalition. This architectural feature provides partial immunity to actor stagnation but does not eliminate the underlying gradient pathology.

6.5 Broader Impact

This analysis has implications beyond MEC task offloading:

1. **DDPG/TD3/SAC implementations:** All use tanh-bounded outputs and may exhibit similar behavior. TD3's delayed policy updates Fujimoto et al. [2018] may partially mitigate this by reducing actor update frequency, implicitly increasing the effective actor time constant.

2. **Hyperparameter transfer:** Learning rates optimal in one domain may fail in others with different reward scales. Our analysis suggests that practitioners should normalize rewards or adjust learning rates proportionally when transferring hyperparameters across domains.

3. **Multi-agent systems:** Shared rewards amplify the effective reward scale (N agents $\times$ per-agent contribution), making MARL systems particularly susceptible to the saturation effects we characterize.

4. **Reward shaping:** Dense reward shaping, often used to accelerate learning, can inadvertently increase reward magnitude and trigger premature actor saturation if learning rates are not correspondingly reduced.

6.6 Comparison with Existing Mitigation Techniques

Several techniques in the literature implicitly address aspects of the gradient asymmetry problem:

- **Delayed policy updates (TD3):** By updating the actor less frequently than the critic, TD3 effectively reduces the actor's learning rate relative to the critic, aligning with our recommendation for conservative actor learning rates.

- **Entropy regularization (SAC):** The entropy bonus in SAC encourages policy diversity, potentially preventing early convergence to saturated outputs. However, this does not directly address the gradient magnitude asymmetry.
- **Target networks:** Target networks stabilize critic learning but do not affect the actor's susceptibility to saturation.
- **Gradient clipping:** While gradient clipping can prevent exploding gradients in critics, it does not address the vanishing gradient problem in actors caused by tanh saturation.

Our analysis suggests that these techniques are incomplete solutions. A more direct approach would be to address the output activation asymmetry itself, either through alternative bounded activations or through explicit gradient balancing mechanisms.

6.7 Summary of Evidence

Our comprehensive experiments validate the theoretical analysis through multiple lines of evidence:

1. **Gradient asymmetry:** Measured ratio $\rho \sim 10^{-8}$ to 10^{-4} confirms 4–8 orders of magnitude critic gradient dominance across all configurations.
2. **Saturation correlation:** Strong correlation ($r > 0.9$) between final saturation ratio and stopping episode demonstrates the causal chain from high LR to saturation to weight update cessation.
3. **Layer-wise analysis:** Gradient breakdown occurs specifically at the actor output layer (tanh), while hidden layers (ReLU) maintain gradient flow.
4. **Consistency:** The pattern holds across all 16 learning rate configurations, with actor LR as the dominant factor and critic LR having secondary effect only at extreme values (0.1).

These findings provide both theoretical explanation and empirical validation for the gradient asymmetry phenomenon in actor-critic architectures.

7 Conclusion

This paper investigated the phenomenon of asymmetric convergence behavior between actor and critic networks in Client-Master MADRL architectures. Motivated by our prior observation that actors stopped updating within ∼5 episodes under high learning rates Gebrekidan [2024], we conducted comprehensive experiments with GPU-optimized implementations and full gradient tracking to identify the underlying mechanism.

Our analysis revealed that **tanh output activation saturation** is the primary mechanism causing client agents to stop updating their neural network weights. Key findings include:

1. **Learning-rate-dependent stopping:** High actor learning rates (0.01–0.1) cause all 50 agents to cease weight updates within 161–247 episodes, while lower rates (0.0001–0.001) maintain gradient flow throughout 2000 episodes of training.

2. **Gradient magnitude asymmetry:** We measured a 4–8 order of magnitude difference in gradient norms between actors and critics ($\rho \sim 10^{-8}$ to 10^{-4}), explaining why actors and critics exhibit differential sensitivity to learning rate selection.

3. **Critic immunity:** The master agent (critic) never stops updating regardless of learning rate configuration, due to its linear (unbounded) output activation that does not suffer from gradient saturation.

4. **Effective learning rate interaction:** The combination of nominal learning rate, reward magnitude, and bounded activation creates an effective learning rate that can rapidly push actor pre-activations into saturation regions where $\tanh'(z) \approx 0$.

These findings have important implications for the design and training of actor-critic MADRL systems:

Hyperparameter Selection: Actors should use learning rates approximately $10\times$ lower than critics to compensate for the gradient asymmetry. Our results suggest actor learning rates in the range 0.0001–0.001 with critic rates of 0.001–0.01.

Architectural Understanding: The empirical preference for smaller actor networks in the literature may be an implicit adaptation to the gradient flow constraints imposed by bounded output activations. Our analysis provides theoretical grounding for this practice.

Monitoring Recommendations: Practitioners should monitor pre-activation statistics and gradient ratios during training to detect early signs of actor saturation before weight updates cease entirely.

Broader Applicability: Since DDPG, TD3, SAC, and other continuous control algorithms share the actor-critic structure with bounded actor outputs, our findings apply beyond CCM-MADRL to the broader family of deterministic policy gradient methods.

7.1 Limitations and Future Work

Our analysis has limitations that suggest directions for future research. The experiments were conducted on a single domain (MEC task offloading); generalization to other continuous control environments (e.g., MuJoCo, robotic manipulation) requires validation. Future work should explore alternative bounded activations with better gradient properties (e.g., softsign, scaled sigmoid), investigate gradient clipping strategies specific to actor networks, and develop adaptive learning rate schemes that account for the measured gradient asymmetry.

Acknowledgments

References

Joshua Achiam. Spinning Up in Deep Reinforcement Learning. https://spinningup.openai.com/, 2018.

Marcin Andrychowicz, Anton Raichuk, Piotr Stańczyk, et al. What matters in on-policy reinforcement learning? a large-scale empirical study. In *International Conference on Learning Representations*, 2020.

Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994.

Xianfu Chen, Honggang Zhang, Celimuge Wu, Shiwen Mao, Yusheng Ji, and Mehdi Bennis. Optimized computation offloading performance in virtual edge computing systems via deep reinforcement learning. *IEEE Internet of Things Journal*, 6(3):4005–4018, 2019.

Yi-Lun Chen, Zhe Gan, Yu Cheng, Jingjing Liu, and Zicheng Liu. Gradient normalization for generative adversarial networks. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6373–6382, 2021.

Karl Cobbe, Jacob Hilton, Oleg Klimov, and John Schulman. Phasic policy gradient. In *International Conference on Machine Learning*, pages 2020–2027. PMLR, 2021.

Theresa Eimer, Marius Lindauer, and Roberta Raileanu. Hyperparameters in reinforcement learning and how to tune them. In *International Conference on Machine Learning*, pages 9104–9149. PMLR, 2023.

Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. *International conference on machine learning*, pages 1587–1596, 2018.

Tesfay Zemuy Gebrekidan. *Deep Reinforcement Learning for Online Combinatorial Resource Allocation with Arbitrary State and Action Spaces*. PhD thesis, University of Southampton, 2024. URL https://eprints.soton.ac.uk/491435/.

Tesfay Zemuy Gebrekidan, Mohammad Shojafar, Zahra Pooranian, Valerio Persico, and Antonio Pescapé. Client-master multiagent deep reinforcement learning for task offloading in mobile edge computing. *ACM Transactions on Autonomous and Adaptive Systems*, 19(3): 1–26, 2024. doi: 10.1145/3768579. URL https://dl.acm.org/doi/abs/10.1145/3768579. Also available at arXiv:2402.11653.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256. JMLR Workshop and Conference Proceedings, 2010.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *International conference on machine learning*, pages 1861–1870, 2018.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters. *Proceedings of the AAAI conference on artificial intelligence*, 32(1), 2018.

Sepp Hochreiter. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 6(02):107–116, 1998.

Andrew Ilyas, Logan Engstrom, Shibani Santurkar, Dimitris Tsipras, Firdaus Janoos, Larry Rudolph, and Aleksander Madry. A closer look at deep policy gradients. In *International Conference on Learning Representations*, 2020.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015.

Riashat Islam, Peter Henderson, Maziar Gomber, Chandra Ber, and Joelle Pineau. Reproducibility of benchmarked deep reinforcement learning tasks for continuous control. *arXiv preprint arXiv:1708.04133*, 2017.

Vijay R Konda and John N Tsitsiklis. Actor-critic algorithms. In *Advances in neural information processing systems*, volume 12, 2000.

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

Ryan Lowe, Yi I Wu, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. In *Advances in neural information processing systems*, volume 30, 2017.

Xiaoteng Ma, Li Xia, Zhengyuan Zhou, Jun Yang, and Qianchuan Zhao. Dsac: Distributional soft actor critic for risk-sensitive reinforcement learning. *arXiv preprint arXiv:2004.14547*, 2020.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. *International conference on machine learning*, pages 1928–1937, 2016.

Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *International conference on machine learning*, pages 807–814, 2010.

Kei Ota, Tomoaki Oiki, Devesh Jha, Toshisada Mariyama, and Daniel Nikovski. Can increasing input dimensionality improve deep reinforcement learning? In *International Conference on Machine Learning*, pages 7424–7433. PMLR, 2020.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *International conference on machine learning*, pages 1310–1318. PMLR, 2013.

Lerrel Pinto, Marcin Andrychowicz, Peter Welinder, Wojciech Zaremba, and Pieter Abbeel. Asymmetric actor critic for image-based robot learning. In *Robotics: Science and Systems*, 2017.

Antonin Raffin, Jens Kober, and Freek Stulp. Smooth exploration for robotic reinforcement learning. *Conference on Robot Learning*, pages 1634–1644, 2021.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In *International conference on machine learning*, pages 387–395. PMLR, 2014.

Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems*, 12, 2000.

Jin Wang, Jia Hu, Geyong Min, Wenhan Zhan, Albert Y Zomaya, and Nektarios Georgalas. Multi-agent deep reinforcement learning for task offloading in mobile edge computing. *IEEE Transactions on Mobile Computing*, 2020.

Ziyu Wang, Tom Schaul, Matteo Hessel, Hado Hasselt, Marc Lanctot, and Nando Freitas. Dueling network architectures for deep reinforcement learning. In *International conference on machine learning*, pages 1995–2003. PMLR, 2016.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. *Advances in Neural Information Processing Systems*, 33:5824–5836, 2020.

Tom Zahavy, Zhongwen Xu, Vivek Veeriah, Matteo Hessel, Junhyuk Oh, Hado P van Hasselt, David Silver, and Satinder Singh. A self-tuning actor-critic algorithm. *Advances in Neural Information Processing Systems*, 33:20913–20924, 2020.