

# Battlesnake Challenge: A Multi-agent Reinforcement Learning Playground with Human-in-the-loop

2022. 06. 27  
김도현

# 0. How Am I?

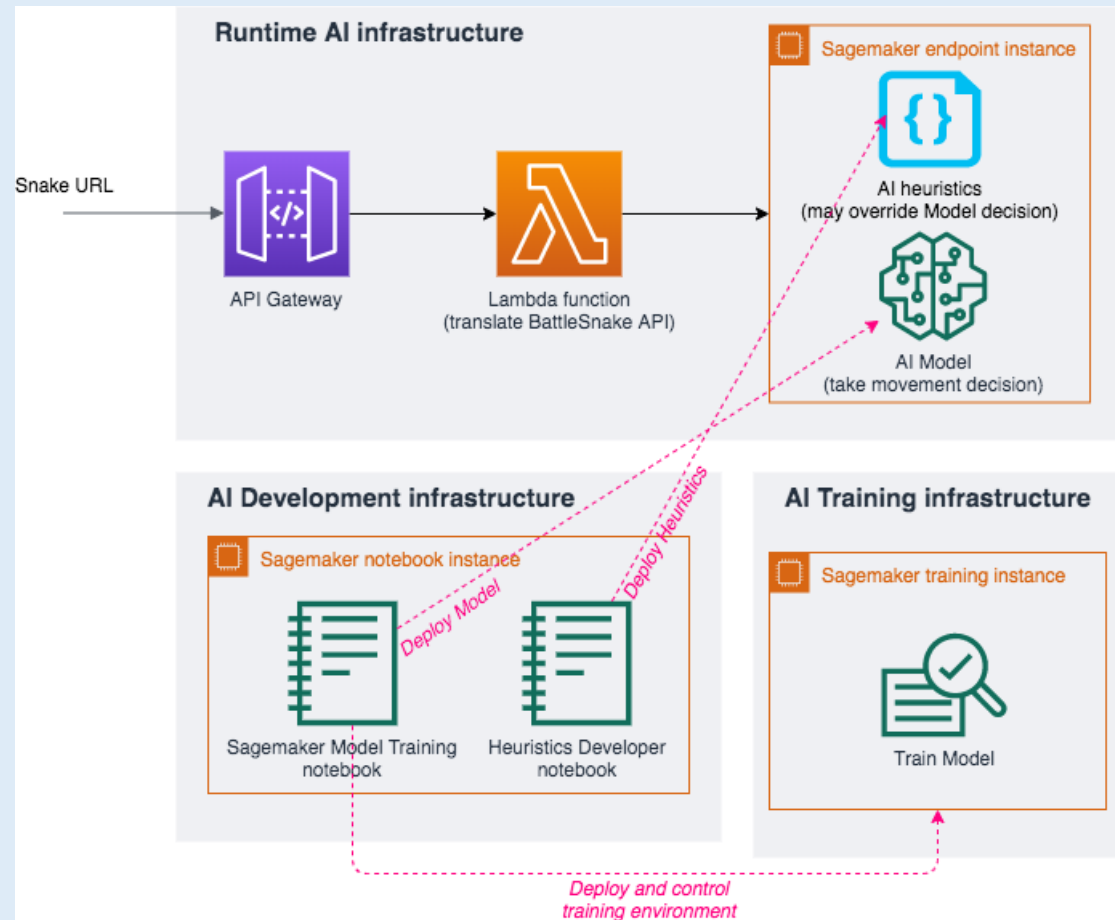
- 경북대학교 컴퓨터학부 4학년 재학 중
- 처음 RL 논문 스터디에 참여했습니다! 부족하지만 잘 부탁드립니다.
- <https://github.com/kimdo331>

# 1. Introduction

- battlesnake 환경
- multi agent RL
- Human-In-the-Loop Learning, HILL
  - feedback, teachers, overseers
  - state space 차원을 줄임
- AWS labs에서 연구하고 발표한 논문
- Amazon Sagemaker 라이브러리를 활용하여 구현됨



# 1. Introduction



# 1. Introduction

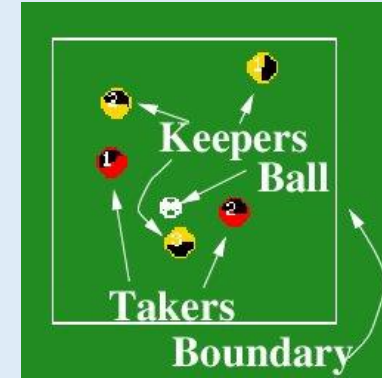
- Framework: Battlesnake Challenge
  - multi agent + HILL 환경 구축 (첫번째 시도)
  - battlesnake는 직관적인 동시에 복잡한 전략이 나올 수 있음
  - offline의 시뮬레이션된 battlesnake 환경
  - online의 battlesnake arena
  - 다양한 heuristic 적용 가능

# 1. Introduction

- Our contributions are as follows:
  - end-to-end framework
    - training - deployment - testing
  - heuristic with HILL 적용가능한 simulator
  - validate
    - HILL agent가 no HILL agent보다 우수
    - heuristic 중 reward manipulation 가장 우수
  - Framework의 Open Source 공개

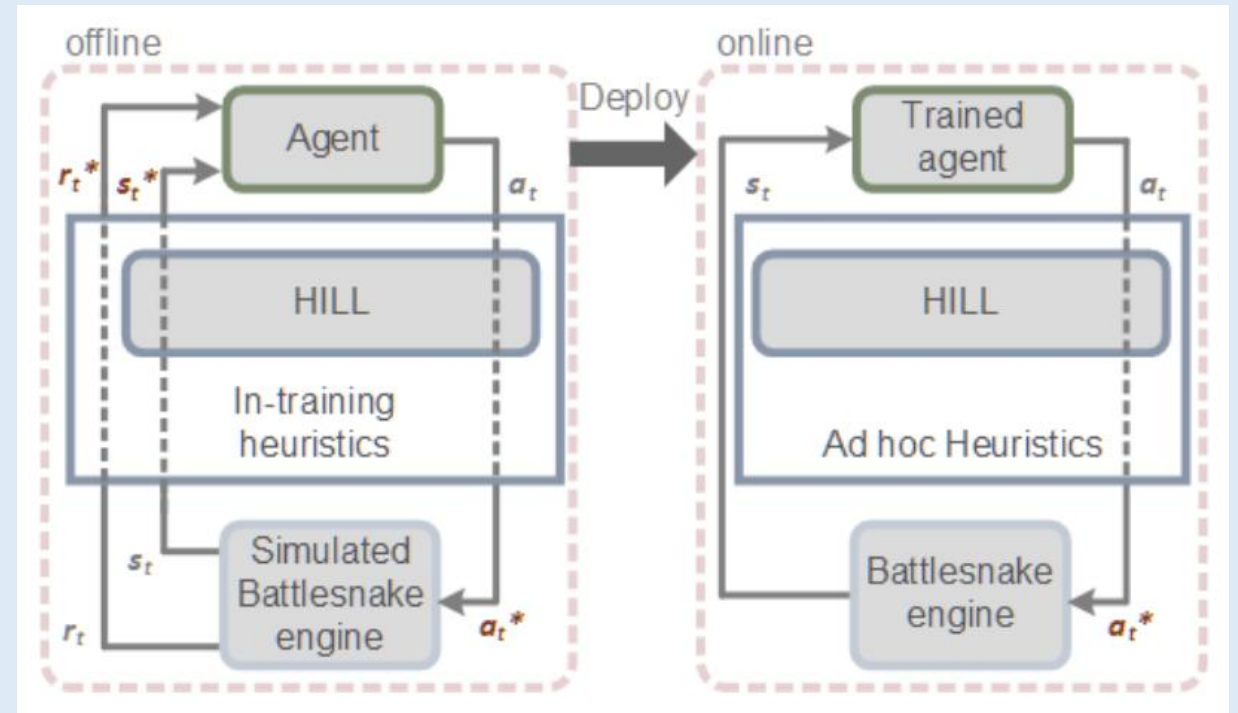
## 2. Related Works

- Multi-agent RL Testbed
  - Keepaway soccer, Pommerman -> no HILL
  - StarCraft 2, Honor of Kings(AOS장르) -> 규칙 복잡
- Human-in-the-loop RL
  - Human intuition
    - 1) by evaluating the actions during training through real-time feedback or intervention(개입)
    - 2) by defining handcrafted rules based on human intuition
  - action masking: 정책 중 불가능한 액션을 금지 (확률을 0으로)
  - reward manipulation: specifically designing a reward function



# 3. Description of the Battlesnake Challenge

- training phase (offline)
  - 인간의 지식이 state, reward, action에 영향
- inference phase (online)
  - heuristic이 action을 뒤킴





## 3.1. Battlesnake description

- battlesnake 게임 규칙
  - 매 턴마다 모든 뱀의 체력 1 감소
  - 매 턴마다 상, 하, 좌, 우 1칸 이동
  - 매 턴마다 음식 무작위 생성
  - 음식 먹으면 체력 = 100, 길이 += 1
  - 다음 경우 뱀 사망
    - 지도 경계를 벗어남
    - 다른 뱀의 몸을 침
    - 자신의 몸을 침
    - 생명력 == 0
    - 서로의 머리를 쳤을 때, 본인 뱀의 머리가 더 작을 경우 (길이 같으면 같이 사망)

## 3.2. Battlesnake as a Reinforcement Learning Environment

- (standard) Markov game

- $\mathcal{N}$ : set of agents
- $\mathcal{S}$ : state space by all agents
- $\mathcal{A}^i$ : action space of agent  $i$
- $\mathcal{T}$ : transition function,  $s_t, a_t$  pair for each agent  $i$
- $R^i$ : reward on each transition for each agent  $i$
- $\gamma$ : discount factor

$$M = (\mathcal{N}, \mathcal{S}, \{\mathcal{A}^i\}_{i \in \mathcal{N}}, \mathcal{T}, \{R^i\}_{i \in \mathcal{N}}, \gamma)$$

probability distribution, 확률분포  
to a PD over  $s_{t+1}$

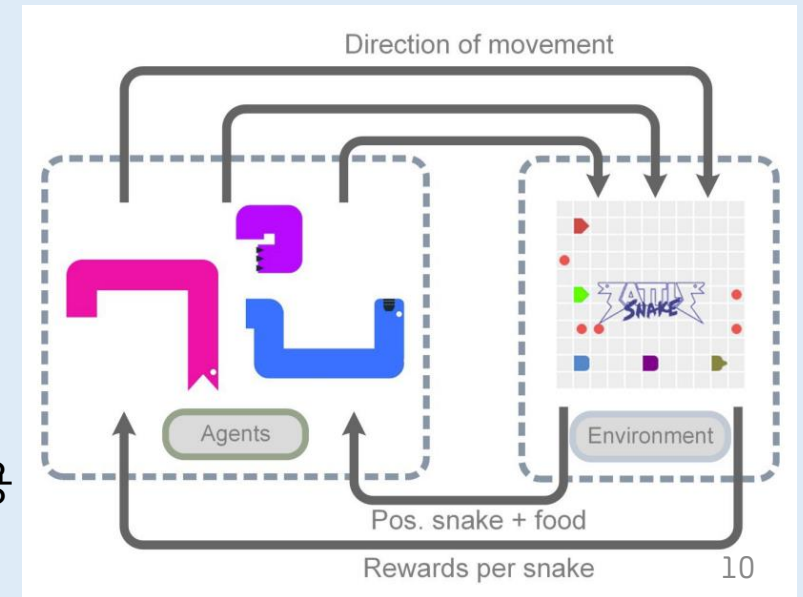


Figure 2. 에이전트와 OpenAI Gym 시뮬레이터의 상호작용

## 3.2. Battlesnake as a Reinforcement Learning Environment

- State
  - 11x11 격자 공간
  - snakes, food 좌표, turn\_count
  - [https://github.com/awslabs/sagemaker-battlesnake-ai/blob/main/source/BattlesnakeGym/battlesnake\\_gym/snake.py#L57](https://github.com/awslabs/sagemaker-battlesnake-ai/blob/main/source/BattlesnakeGym/battlesnake_gym/snake.py#L57)
- action
  - [위, 아래, 왼쪽, 오른쪽]
- reward
  - 마지막까지 유일하게 생존: +1
  - 매 턴마다 생존: +0.002
  - [https://github.com/awslabs/sagemaker-battlesnake-ai/blob/main/source/BattlesnakeGym/battlesnake\\_gym/rewards.py#L28](https://github.com/awslabs/sagemaker-battlesnake-ai/blob/main/source/BattlesnakeGym/battlesnake_gym/rewards.py#L28)

## 3.3. Training Algorithm

- Proximal Policy Optimization
- 이 framework는 알고리즘에 독립적
  - QMIX, SAC 등 discrete action-based algorithm 적용 가능

## 3.4. Heuristics with HILL

- 이 framework는 휴리스틱과 독립적
- 테스트를 위해 4가지 휴리스틱 시도해봄
  1. 벽 부딪힘 방지
  2. 이동 방향의 반대 방향으로 이동 금지
  3. 체력 낮을 때 음식으로 이동
  4. 다른 뱀을 공격
- 휴리스틱에 의한 편향 가능성 -> 학습 충분히 되면 휴리스틱 비활성화
  - 규칙3에 의해 음식에만 집착
  - 규칙4에 의한 과도한 공격성

Rule	Prevention/ Promotion	Interaction	Training phase
1	Prev.	Env.	Early
2	Prev.	Env.	Early
3	Promo.	Env.	Middle
4	Promo.	Agents	Late

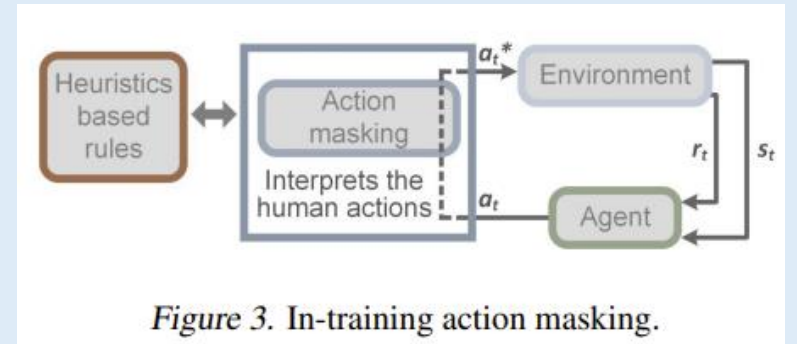
Table 1. Properties of the heuristics. Prev. refers to action pre-

## 3.4. Heuristics with HILL

- 휴리스틱을 RL에 포함시키는 세가지 방법
- In-training action masking
- Ad-hoc action overwriting
- Reward manipulation

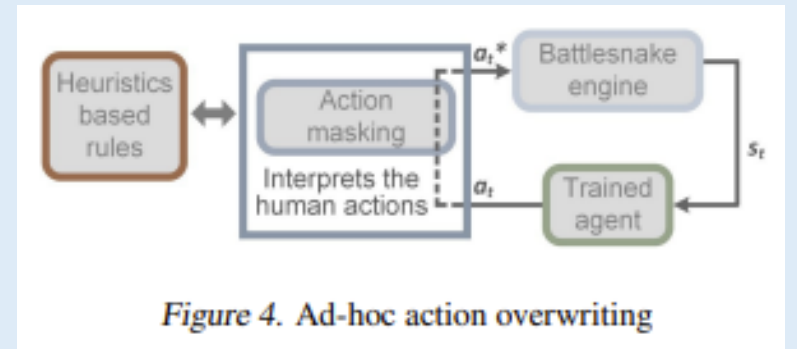
## 3.4. Heuristics with HILL

- In-training action masking
- 학습 단계 중 정책의 최종 출력에서, invalid action 확률을 0으로 만들고 softmax
- 규칙1, 규칙2에 적용
- <https://github.com/aws-labs/sagemaker-battlesnake-ai/blob/main/source/RLlibEnv/HILL-training.ipynb>



## 3.4. Heuristics with HILL

- Ad-hoc action overwriting
- 이전 방식과 유사하나, 추론 단계에서 휴리스틱 적용
- 정책 업데이트 X





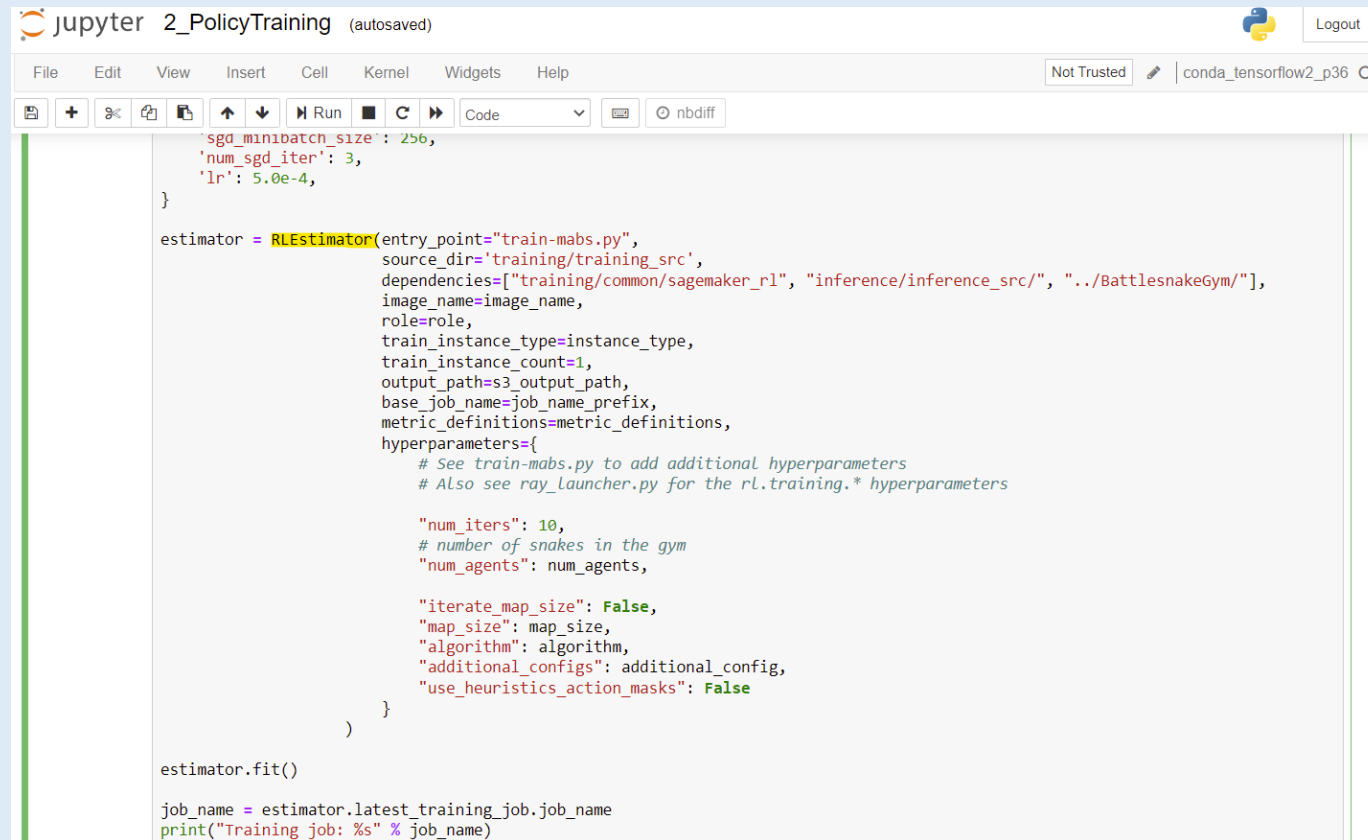
## 3.4. Heuristics with HILL

- Reward manipulation
- 리워드 조건을 재정의

```
forbidden_move_rewards = {"another_turn": 0.01,  
                           "ate_food": 0,  
                           "won": 5,  
                           "died": -5,  
                           "ate_another_snake": 0,  
                           "hit_wall": 0,  
                           "hit_other_snake": 0,  
                           "hit_self": 0,  
                           "was_eaten": 0,  
                           "other_snake_hit_body": 0,  
                           "forbidden_move": -2,  
                           "starved": 0}  
  
hit_wall_rewards = {"another_turn": 0.01,  
                    "ate_food": 0,  
                    "won": 5,  
                    "died": -5,  
                    "ate_another_snake": 0,  
                    "hit_wall": -2,  
                    "hit_other_snake": 0,  
                    "hit_self": 0,  
                    "was_eaten": 0,  
                    "other_snake_hit_body": 0,  
                    "forbidden_move": 0,  
                    "starved": 0}  
  
starved_rewards = {"another_turn": 0.01,  
                   "ate_food": 0,  
                   "won": 5,  
                   "died": 5,  
                   "ate_another_snake": 0,  
                   "hit_wall": 0,  
                   "hit_other_snake": 0,  
                   "hit_self": 0,  
                   "was_eaten": 0,  
                   "other_snake_hit_body": 0,  
                   "forbidden_move": 0,  
                   "starved": -2}  
  
kill_other_snake_rewards = {"another_turn": 0.01,  
                             "ate_food": 0,  
                             "won": 5,  
                             "died": -5,  
                             "ate_another_snake": 2,  
                             "hit_wall": 0,  
                             "hit_other_snake": 0,  
                             "hit_self": 0,  
                             "was_eaten": 0,  
                             "other_snake_hit_body": 2,  
                             "forbidden_move": 0,  
                             "starved": 0}
```

# 4.1. Implementation details

- Amazon SageMaker RL 패키지 내에서 RLlib 사용하여 구현됨



```
jupyter 2_PolicyTraining (autosaved)
File Edit View Insert Cell Kernel Widgets Help
Not Trusted conda_tensorflow2_p36
+ -> Run C Code nbdiff

'sgd_minibatch_size': 256,
'num_sgd_iter': 3,
'lr': 5.0e-4,
}

estimator = RLEstimator(entry_point="train-mabs.py",
                        source_dir='training/training_src',
                        dependencies=["training/common/sagemaker_rl", "inference/inference_src/", "../BattlesnakeGym/"],
                        image_name=image_name,
                        role=role,
                        train_instance_type=train_instance_type,
                        train_instance_count=1,
                        output_path=s3_output_path,
                        base_job_name=job_name_prefix,
                        metric_definitions=metric_definitions,
                        hyperparameters={
                            # See train-mabs.py to add additional hyperparameters
                            # Also see ray_launcher.py for the rl.training.* hyperparameters

                            "num_iters": 10,
                            # number of snakes in the gym
                            "num_agents": num_agents,

                            "iterate_map_size": False,
                            "map_size": map_size,
                            "algorithm": algorithm,
                            "additional_configs": additional_config,
                            "use_heuristics_action_masks": False
                        })

estimator.fit()

job_name = estimator.latest_training_job.job_name
print("Training job: %s" % job_name)
```

## 4.2. Evaluation & 5. Results

- HILL을 이용한 RL agent 평가하는 방법
  1. Training 중 퍼포먼스 평가
  2. 블랙박스로 Battlesnake arena에서 다른 뱀과 경쟁

## 4.2. Evaluation & 5. Results

- baseline
  - agent 개수 (서로 다른 seed)
  - map size

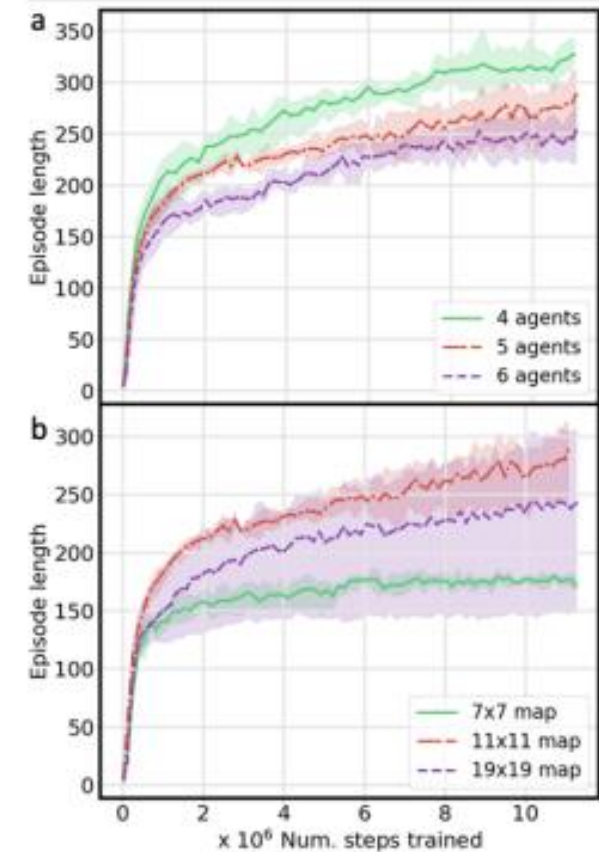


Figure 5. Episode lengths with (a) varying the number of agents on a  $11 \times 11$  map and (b) varying the map size with five agents.

## 4.2. Evaluation & 5. Results

### 1. Training 중 퍼포먼스 평가

- episode length
- event 빈도 (벽에 부딪히거나 금지된 행동, 음식먹는 횟수 등)
- in-training action masking
  - 규칙1(벽충돌): 약 250
  - 규칙2(금지동작): 약 300
  - action masking으로 학습 가속화

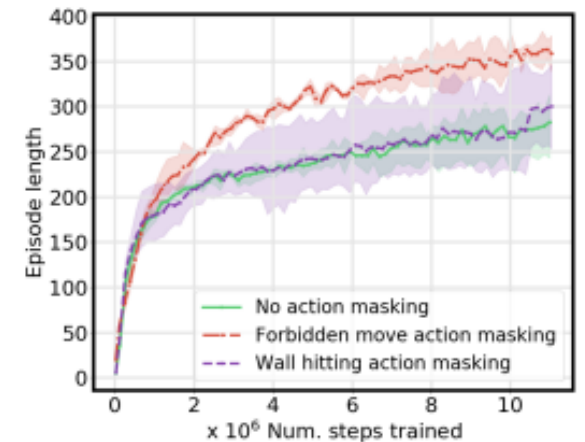


Figure 6. Experiments with action masking for rule 1 and 2 with 5 agents on a  $11 \times 11$  map

## 4.2. Evaluation & 5. Results

### 1. Training 중 퍼포먼스 평가

- reward manipulation
  - 금지된 이동은 주요 사망 요인
  - 평균 10에서 3으로 감소
  - reward manipulation 없이도 금지된 이동 감소함
- reward manipulation보다 action masking이 성능 우수

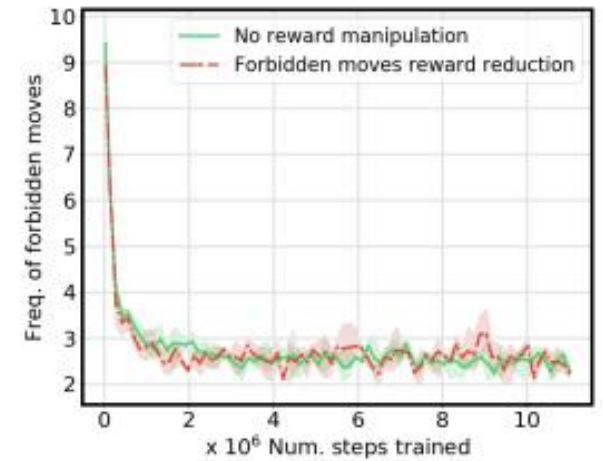


Figure 7. Experiments with reward manipulation for rule 2, forbidden moves with 5 agents on a  $11 \times 11$  map.

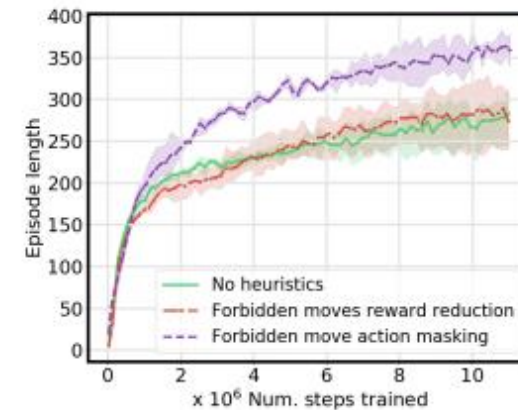


Figure 8. Comparison between the performance of in-training action masking, reward manipulation, and no HILL with 5 agents on a  $11 \times 11$  map.

## 4.2. Evaluation & 5. Results

### 2. 블랙박스로 Battlesnake arena에서 다른 뱀과 경쟁

- 목적: 최고 수준의 agent 학습 (X)  
heuristic이 학습에 어떤 영향을 주는지 평가
- Table 2. 경기장에 4마리 뱀이 서로 경쟁, 30게임
  - 훈련: 2500 episode, 11x11 map
  - 1등 -> 4점, 2등 -> 3점, ...
  - 금지된 이동 횟수 기록

HILL type	Arena score	%Forbidden moves
No HILL	$2.200 \pm 0.846$	26.6%
In-training AM	$2.533 \pm 1.074$	0%
RM	$2.900 \pm 1.296$	13.3%
Ad-hoc AO	$2.333 \pm 1.154$	0 %

Table 2. Scores in the Battlesnake arena and the % of deaths caused by forbidden moves. AM refers to action masking, RM refers to reward manipulation, and AO refers to action overwriting.

## 4.2. Evaluation & 5. Results

### 2. 블랙박스로 Battlesnake arena에서 다른 뱀과 경쟁

- Table 3. 1대1 경기, 각 10경기씩
  - HILL 적용 시 향상 (?)

	No HILL	IT AM	RM	AH AO
No HILL	-	4	3	6
IT AM	6	-	1	2
RM	7	9	-	1
AH AO	4	8	9	-

*Table 3.* Scores (with respect to the rows) in the Battlesnake arena in a 1 vs 1 format. IT AM, AH AO and RM refer to in-training action masking, Ad-hoc action overwriting and reward manipulation respectively.



## 6. Conclusion

- Battlesnake Challenge: Human-in-the-loop에서의 multi-agent RL 프레임워크를 도입
- 휴리스틱 성능 측정, 각 휴리스틱 별 offline, online 성능이 다름
- 전반적으로 HILL 적용 시 성능 향상
- in-training action masking 적용 시 에피소드 길이 확장
- 반대로, battlesnake arena에서는 reward manipulation이 AM 능가