동적 환경에서의 지속적인 다중 에이전트 강화 학습

정규열O, 김인철
경기대학교 컴퓨터과학과
jk96491@kyonggi.ac.kr, kic@kyonggi.ac.kr

Continual Multiagent Reinforcement Learning in Dynamic Environments

Kyuyeol JungO, Incheol Kim
Department of Computer Science, Kyonggi University

요 약

다양한 실세계 응용 분야들에서 공동의 목표를 위해 여러 에이전트들이 상호 유기적으로 협력할 수 있는 행동 정책을 배우는 것은 매우 중요하다. 이러한 다중 에이전트 강화 학습(MARL) 환경에서 기존의 연구들은 대부분 중앙-집중형 훈련과 분산형 실행(CTDE) 방식을 사실상 표준 프레임워크로 채택해왔다. 하지만 이러한 다중 에이전트 강화 학습 방식은 훈련 시간 동안에는 경험하지 못한 새로운 환경 변화가 실전 상황에서 끊임없이 발생할 수 있는 동적 환경에서는 효과적으로 대처하기 어렵다. 이러한 동적 환경에 효과적으로 대응하기 위해, 본 논문에서는 새로운 다중 에이전트 강화 학습 체계인 C-COMA를 제안한다. C-COMA는 에이전트들의 훈련 시간과 실행 시간을 따로 나누지 않고, 처음부터 실전 상황을 가정하고 지속적으로 에이전트들의 협력적 행동 정책을 학습해나가는 지속 학습 모델이다. 본 논문에서는 대표적인 실시간 전략게임인 Starcraft II를 토대로 동적 미니게임을 구현하고 이 환경을 이용한 다양한 실험들을 수행함으로써, 제안 모델인 C-COMA의 효과와 우수성을 입증한다.

1. 서 론

일반적으로 현실 세계는 여러 자율 에이전트들이 공존하는 다중 에이전트 환경이다. 예를 들어 교통 신호등 제어, 자율주행 차량의 제어, 다수의 플레이어가 활동하는 비디오 게임 등 다중 에이전트 환경으로 구성되어 있다. 이러한 다중 에이전트 환경에서 각 에이전트는 경우에 따라서 어떤 때에는 서로 유기적으로 협력해야 하고, 어떤 때로 서로 경쟁하거나 적대적으로 행동해야 한다. 단일 에이전트의 경우와 마찬가지로, 다중 에이전트 행동 정책 학습을 위해서도 그동안 많은 심층 강화 학습(deep reinforcement learning) 기술들이 소개되었으며, 바둑이나 atari 비디오 게임 등과 같은 비교적 고난※도 작업들에서도 큰 성공을 보여주고 있다. 하지만 다중 에이전트 환경은 다음과 같은 요소들 때문에 아직도 효율적으로 행동 정책을 학습하기 어려운 경우가 많다. 첫째 대부분의 다중 에이전트 환경에서는 각 에이전트에게 환경에 대한 완전한 상태 정보(complete satet information)는 주어지지 않고, 다만 부분적이고 불완전한 관측 정보(partial, incomplete observation)만 주어진다. 또한 많은 다중 에이전트 환경에서는 에이전트들 간의 통신(communication)이 전혀 불가능하거나 혹은 매우 제한적으로만 가능하다. 따라서 효율적인 팀워크 활동을 위해 필수적으로 요구되는 팀원들 간의 실시간 정보 교환(information exchange) 및 행위 조율(behavior coordination)이 어려워진다.

이러한 문제를 극복하고 에이전트 각각이 유기적으로 협동하여 팀 차원의 효율적인 행동을 수행하기 위해 그동안 다양한 다중 에이전트 강화 학습(Multi Agent Reinforcement Learning, MARL)기법들이 소개되었다. 특히 최근에는 실시간 전략게임인 Starcraft II 환경을 다루는 다중 에이전트 챌린지(StarCraft Multi-Agent Challenge, SMAC)[3]를 중심으로 중앙 집중형 훈련과 분산형 실행(Centralized Training with Decentralized Execution, CTDE) 체제가 널리 보급되고 있다. 이러한 다중 에이전트 강화 학습의 대표 모델들로는 행동자-비평가(actor-critic) 구조에 기반한 COMA[1]와 Q 학습에 기반한 VDN[4], QMIX[2] 등이 있다. 이 학습 모델들의 가장 큰 특징은 중앙의 조정자 도움을 받아 각 에이전트가 최적의 Q 함수나 개별 행동 정책을 학습하고 나면, 실행 중에는 더 이상 학습을 진행하지 않는다. 다시 말해 학습 시와는 달리 수행 시에는 환경으로부터 추가적인 피드백과 이에 기초한 각 에이전트의 가치 함수(value function)나 정책 네트워크(policy network)의 변화는 발생하지 않는다.

※ 이 논문은 정보통신기획평가원의 재원으로 정보통신방송 기술개발사업의 지원을 받아 수행한 연구 과제(클라우드에 연결된 개별 로봇 및 로봇그룹의 작업 계획 기술 개발, 2020-0-00096)입니다.

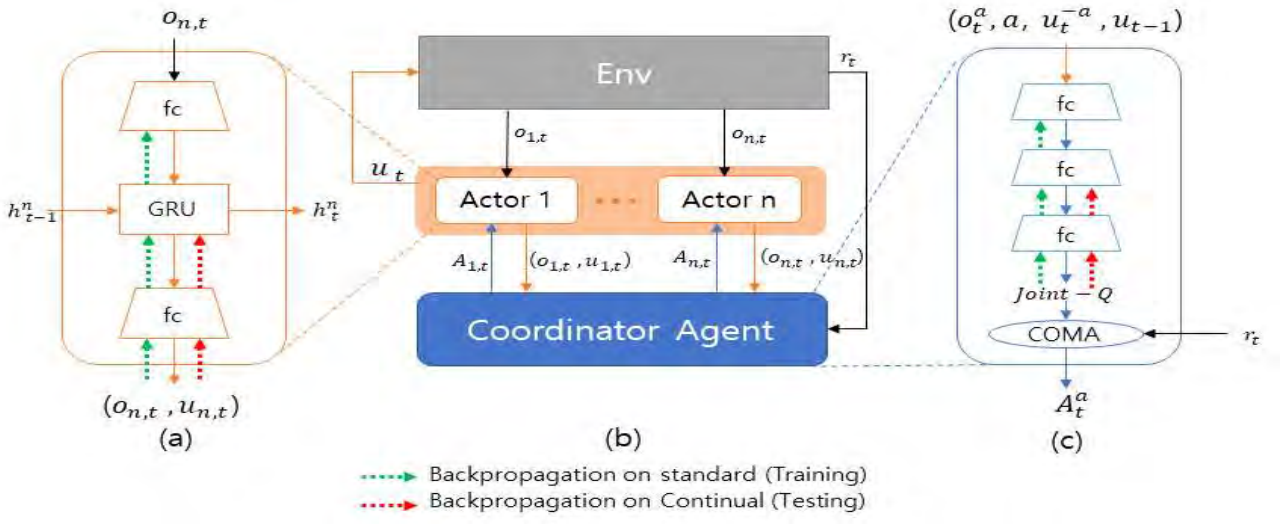
이러한 CTDE 기반의 다중 에이전트 강화 학습 모델들은 에피소드마다 아군 에이전트와 적군의 개체 수, 구성, 등장 위치 등등의 요소들이 고정되어있는 정적 환경(static environment)에서는 좋은 성능을 발휘할 수 있다. 하지만 (그림 1)의 (a)와 같이 에피소드마다 적군의 구성이 달라지거나, (그림 1)의 (b)와 같이 적군의 등장 위치가 변동되는 동적 변화가 학습 시간 이후에도 계속 발생할 수 있는 동적 환경에서는 큰 효과를 기대하기 어렵다. 따라서 경험하지 못한 새로운 환경 변화가 계속 발생하는 동적 환경에서도 이 변화에 적응하면서 효율적으로 팀 단위의 협력적 행동 정책을 학습할 수 있는 새로운 다중 에이전트 강화 학습 모델의 개발이 필요하다.

(a) 상대 팀 구성의 변화

(b) 전투 위치의 변화

(그림 1) Starcraft II 미니게임에서 동적 환경 변화

본 논문에서는 경험하지 못한 상황이 계속해서 발생하는 동



(그림 2) 지속적인 다중 에이전트 강화 학습을 위한 C-COMA의 구조

적인 환경에서도 좋은 성능을 발휘하기 위한 새로운 지속적인 다중 에이전트 강화 학습 모델 C-COMA(Continual COMA)를 제안한다. 이 모델은 행동자-비평가(actor-critic) 구조를 기반으로 비-사실적 추론(conterfactual reasoning)이 가능한 COMA[2]를 지속 학습(continual learning)이 가능하도록 확장한 모델이다. 즉 C-COMA는 에이전트들의 훈련 시간과 실행 시간을 따로 나누지 않고, 처음부터 실전 상황을 가정하고 지속적으로 에이전트들의 협력적 행동 정책을 학습해나가는 지속 학습 모델이다. 본 논문에서는 대표적인 실시간 전략게임인 Starcraft II를 토대로 동적 미니게임을 구현하고 이 환경을 이용한 다양한 실험들을 수행함으로써, 제안 모델인 C-COMA의 효과와 우수성을 입증한다.

2. 문제 정형화

본 논문에서는 부분 관측 정보를 이용하여 진행하는 기법인 Dec-POMDP(Decentralized Partially Observable Markov Decision Process)를 기초로, 다중 에이전트 강화 학습 문제를 $G = \langle U, P, r, O, n, \gamma \rangle$ 같은 튜플로 정의하며 에이전트 n은 순차적으로 선택된 행동 $a \in A \equiv \{1, ..., n\}$의해 식별된다. O는 에이전트들의 협동 관측 공간(joint observation space)이며 $O = \{O_1, O_2, \cdots, O_n\}$와 같이 개별 에이전트의 관측공간 O_i의 조합으로 정의한다. 개별 에이전트 i의 부분관측(partial observation)은 $o_i \in O$이다. 매 시간 반복 마다 각 에이전트는 행동 $u^a \in U$을 선택하며 협동 행동(Joint-Action) $\boldsymbol{u} \in \boldsymbol{U} \equiv U^n$을 이룬다. 이는 환경으로부터 관측 전이 함수(observation transition function)가 유도 $P(o^{'}|o, \boldsymbol{u}) : O \times U \times O \rightarrow [0,1]$된다. 에이전트 모두 동일한 보상 함수 $r(o^{'}, \boldsymbol{u}) : O \times U \rightarrow R$를 공유하며 보상에 대한 감가율은 $\gamma \in [0,1]$이다. 각 에이전트들은 행동 관측의 과거 정보 $\tau^a \in T \equiv (O \times U)^*$를 가지고 있고 확률적 정책(stochastic policy)은 $\pi^a(u^a|\tau^a) : T \times U \rightarrow [0,1]$이다. 에이전트 본인은 a, 다른 에이전트들은 $-a$로 정의한다. 감가된 보상(discounted return)은 $R_t = \sum_{l=0}^{\infty} \gamma^l r_{t+l}$이다. 에이전트의 협동 정책은 가치함수를 유도하는데 R_t의 기댓값 $V^{\pi}(s_t) = E_{s_{t+1:\infty}, \boldsymbol{u}_{t:\infty}}[R_t|s_t]$로 정의한다. 그리고 행동 가치함수(action value function)는 $Q^{\pi}(s_t, \boldsymbol{u_t}) = E_{s_{t+1:\infty}, \boldsymbol{u}_{t:\infty}}[R_t|s_t, \boldsymbol{u_t}]$로 정의한다. 우세 함수(advantage function)는 $A^{\pi}(s_t, \boldsymbol{u_t}) = Q^{\pi}(s_t, \boldsymbol{u_t}) - V^{\pi}(s_t)$로 정의한다.

3, 지속적인 다중 에이전트 강화 학습

3.1 C-COMA 학습 모델

다중 에이전트 문제에서 가장 중요한 요소는 팀 보상을 통하여 각 에이전트 개인의 기여도를 적절히 부여하는 것이다. 더욱이 동적 환경에서는 복잡성이 증가하게 되어 문제가 어려워진다. 이를 효과적으로 해결하기 위하여 본 논문에서는 CTDE 기법의 대표적인 COMA[1]기법을 지속학습의 목적에 맞게 변형하여 적용하였다. 왜냐하면 다른 CTDE 기법들과 다르게 비-사실적 추론을 도입하여 팀 보상에 대한 행동 에이전트들의 기여도를 보다 효율적으로 산출한다. 이 기여도를 이용하여 에이전트의 정책을 팀 차원의 협력에 맞추도록 조정한다.

이에 기존 COMA의 전체적인 구조는 (그림 2)와 같이 두 가지 구성 요소로 이루어져 있다. 첫 번째는 (그림 2)의 (a)와 같이 실질적으로 행동을 취하는 행동 에이전트(Actor Agent)와 두 번째는 (그림 2)의 (c)와 같이 행동 에이전트들이 전달해주는 정보를 모두 받아 조율 역할을 하는 조정자 에이전트(Coordinator Agent)로 이루어져 있다. 행동 에이전트의 역할은 환경으로부터 부분 관측된 정보와 과거 정보를 이용하여 정책을 결정하고 정보를 조정자 에이전트에게 전달한다. 조정자 에이전트는 행동 에이전트들로부터 정보를 제공 받아서 팀 차원의 관점으로 가치를 학습 하고 행동 에이전트들이 개인이 아닌 팀 관점에 맞는 효율적인 행동을 수행하도록 조정한다. 이러한 조정에 의하여 행동 에이전트의 학습이 진행된다.

동적인 환경에 대응하기 위한 지속 학습을 위하여 COMA에 몇 가지 사항을 변경하였다. 첫 번째로 학습과 수행에 관계 없이 계속 학습하도록 구조 변경하였다. 구체적으로 일반적인 CTDE 기법에서 학습시에는 행동 에이전트(그림 2)의 (a)와 조정자 에이전트(그림 2)의 (c) 모두 사용된다. 그러나 수행 시에는 조정자 에이전트는 사라지고 행동 에이전트만 남는다. 따라서 행동 에이전트들은 본인들의 학습된 정책을 통하여 수행만 하게 된다. 이렇게 되면 학습 시에 경험하지 못한 상황에 대처 능력이 떨어진다. 이에 따라 학습 시와 동일하게 수행 시에도 조정자 에이전트의 도움을 받아 새로운 환경에서도 지속적으로 피드백이 가능 하도록 변형하였다. 이 때 기존에 학습한 경험을 어느정도 유지하며 새로운 경험에 적응해야 한다. 따라서 조정자 에이전트와 행동 에이전트 모두 과거의 학습 기록을 어느정도 기억을 하기 위하여 지속학습을 적용하였다. 학습 중에는 (그림 2)의 (a), (c) 에서 초록색 화살표와 같이 역전파 과정을 모두 적용한다. 즉, 학습을 진행할 때는 모든 신경망의 파라미터들을 수정한다. 이와 반대로 수행 중에는 붉은색 화살표와 같이 파라미터의 일부분만 역전파를 진행하여 수정한다. 이 방법에 따라서 행동 에이전트와 조정자 에이전트의 신경망이 학습된다.

3.2 조정자 에이전트

조정자 에이전트는 환경에 등장하지 않는 가상의 에이전트이며 환경의 모든 정보를 수집하여 행동 에이전트들의 정책을 제어한다. 만일 행동 에이전트끼리만 정책을 수립한다면 팀 차원의 협동에 따른 정책을 수립하기 어려울 것이다. 그 이유는 본인들의 부분관측 정보만 볼 수 있을 뿐 전체 정보를 보지 못하기 때문이다. 따라서 전체 정보를 볼 수 있는 조정자 에이전트를 추가하여 행동 에이전트들이 팀을 위한 협동에 맞는 정책을 수립하도록 도움을 준다.

조정자 에이전트는 (그림 2)의 (b), (c) 같이 행동 에이전트들로부터 부분관측 정보와, 정책을 전달받아 행동 에이전트들이 팀 관점에 맞추어 정책을 결정 하도록 유도한다. 다시 말해 조정자 에이전트는 행동 에이전트와 달리 전체 정보를 볼 수 있다. 그러나 행동 에이전트들은 전체 정보 중 본인이 볼 수 있

는 일부 정보만을 토대로 행동을 한다. 그러므로 팀 관점이 아닌 개인의 관점에서 행동을 수행 할 수 밖에 없다. 따라서 조정자 에이전트는 행동 에이전트들의 정책을 팀 관점에서 수행하도록 유도하는 역할을 한다. 이를 위하여 조정자 에이전트는 협동에 대한 Q(Joint-Q)값을 산출한다. 이 값을 통하여 조정자 에이전트는 팀 차원의 가치를 학습한다. 가치를 학습하기 위하여 조정자 에이전트에 대하여 (식 1)의 $y^{(\lambda)}$ 와 같이 타겟 네트워크를 설정한다. 이후 이 타켓 네트워크는 신경망에 최적인 TD(λ)기법을 통하여 값을 산출한다. TD(λ) 기법은 보상의 감가율(Discount Factor) λ이 반영된 n-step return의 혼합체이며 $G_t^{(n)} = \sum_{l=1}^{n} \gamma^{l-1} r_{t+l} + \gamma^n f^{co}(\bullet_{t+n}, \theta^{co})$ 같이 사용된다. 이후 조정자 에이전트 $f^{co}(\bullet, \theta^{co})$는 활성 정책(on-policy)방법으로 업데이트를 진행한다. 조정자 에이전트의 파라미터θ^{co}는 식(1)과 같이 매 시간마다 타겟 네트워크의 결과값과 평균 제곱 오차(MSE)로 손실 함수$L_t(\theta^{co})$를 정의할 수 있다. 이를 이용하여 경사 하강법(Gradient Descent)의 기법으로 조정자 에이전트는 보상을 학습한다.

$$L_t(\theta^{co}) = (y^{(\lambda)} - f^{co}(\bullet_t, \theta^{co}))^2 \quad \text{(식 1)}$$

$$where,\ y^{(\lambda)} = (1-\lambda)\sum_{n=1}^{\infty} \lambda^{n-1} G_t^{(n)}$$

$$\theta^{co} = \theta^{co} - \nabla_{\theta^{co}} L_t(\theta^{co})$$

이 과정을 통하여 학습된 조정자 에이전트는 각 행동 에이전트의 우세 값을 부여한다. 일반적으로 우세 값은 (식 2)와 같이 TD-error 기법을 사용한다.

$$g = \nabla_{\theta^{\pi}} \log \pi(u|\tau_t^a)(r + \gamma V(o_{t+1}) - V(o_t)) \quad \text{(식 2)}$$

그러나 이 방법은 팀 차원의 보상만 추론이 가능하고 에이전트 각각의 기여도를 측정하기는 어렵다. 이를 위하여 비-사실적 추론 기법[1] 으로 우세 값을 산출한다. 이 기법은 전체 팀 보상 $r(o,\boldsymbol{u})$에서 각 에이전트의 보상 $r(s,(\boldsymbol{u}^{-a}, c^a))$의 차(difference reward)를 이용한 $D^a = r(s,\boldsymbol{u}) - r(s,(\boldsymbol{u}^{-a}, c^a))$의 개념에서 출발한다. 다시 말해 행동 에이전트 a가 기본적 행동(default Action) c^a를 하였을 때 전체 보상에서 개별 보상의 차이를 나타낸다. 이 기법은 상당히 강력한 방법이나 모든 가능성과 경우의 수를 고려해야 한다. 따라서 연산의 과정이 상당히 복잡해지며 비효율적이다. 따라서 (식 3)과 같이 협동의 Q 값 $Q(o,\boldsymbol{u})$에서 다른 에이전트들의 행동 $\boldsymbol{u}^{-a}$은 고정하고 본인의 행동 u'^a에 따라서만 확률을 고려하는 비-사실적 추론(conterfactual reasoning)기법으로 산출된 값을 차감하여 우세 $A^a(o,\boldsymbol{u})$값을 산출한다.

$$A^a(o,\boldsymbol{u}) = Q(o,\boldsymbol{u}) - \sum_{u'^a} \pi^a(u'^a|\tau^a) Q(o,(\boldsymbol{u}^{-a}, u'^a)) \quad \text{(식 3)}$$

그러나 이 자체로도 신경망에서 사용한다면 연산량이 상당히 커진다. 또 한 협동의 공간(joint-action space)도 $|U|^n$ 만큼 커지게 되고 효율적인 학습을 어렵게 한다. 따라서 representation 기법을 도입하였다. 구체적으로 (그림 2)의 (c)와 같이 다른 에이전트의 행동 $\boldsymbol{u}_t^{-a}$을 신경망의 입력값으로 사용한다. 결과적으로 비-사실적 추론에 의한 우세(advantage by counterfactual reasoning)값이 각 행동 에이전트들에게 효율적으로 전달이 된다. 또 한 협동의 공간(joint-action space)도 $|U|^n$에서 $|U|$로 감소하게 된다.

결론적으로 조정자 에이전트는 행동 에이전트가 전달해준 정보를 이용하여 보상을 학습한다. 이를 이용하여 각 부분 정보만 관측 가능한 행동 에이전트의 정책을 팀 관점에 맞추도록 조정하는 역할을 수행한다.

3.3 행동 에이전트

행동 에이전트는 환경에서 실질적으로 정책을 수행한다. 정책을 수행하기 위해 행동 에이전트는 부분 관측된 정보에 의존하게 된다. 따라서 과거에 관측되었던 정보를 활용하여 정책을 수립한다. 이후 해당 정책을 조정자 에이전트에게 전달하여 팀 차원의 정책을 수립하도록 조정받는다.

행동 에이전트는 부분 관측된 정보에 의존하게 된다. 그러므로 행동 에이전트가 과거에 관측 정보 τ에 따른 수행 결과의 전체 정보를 활용해야 한다. 따라서 (그림 2)의 (a)를 보면 순환 신경망(recurrent neural network)을 도입하였다. 대표적인 순환 신경망인 장단기메모리(Long-Short Term Memory, LSTM), 게이트 순환 신경망(Gated Recurrent Units, GRU)가 있으며 본 논문에서는 GRU를 사용하였다. 또 한 조정자 에이전트에게 자신의 정책을 조정을 받기 위하여 본인의 부분관측 정보와 결정한 정책θ_a을 조정자 에이전트에게 전달한다. 이후 (그림 2)의 (b)와 같이 조정자 에이전트로부터 우세 값을 부여받아 본인의 정책을 팀 관점의 행동을 하도록 수정한다. (식 4)와 같이 목적 함수 $J(\theta)$를 행동 에이전트들의 정책을 경사 상승법(Gradient Ascent)을 이용하여 수정한다.

$$\nabla_{\theta} J(\theta) = E_{\pi}[\sum_{a} \nabla_{\theta_a} \log \pi_{\theta_a}(u^a|\tau^a) A^a(o,u)] \quad \text{(식 4)}$$

$$\theta = \theta + \alpha \nabla_{\theta} J(\theta)$$

4. 구현 및 실험

4.1 실험 환경과 모델 학습

본 논문의 제안 모델은 windows 10에서 Python 딥러닝 라이브러리인 PyTorch를 이용하여 구현하였다. 모델의 학습 및 평가를 위하여 Starcraft II의 미니게임을 이용한 다중 에이전트 학습환경(SMAC)[3]을 목적에 맞게 변형하여 실험을 수행 하였다.

SMAC은 실시간 전략게임인 Starcraft II를 이용하여 다중 에이전트의 연구를 위해 제작된 미니게임을 제공한다. Starcraft II의 경우는 두 가지 게임 특성이 있는데 하나는 자원관리 및 건물관리(Macromanagement)이고 나머지 하나는 유닛을 직접 일일이 제어(Micromanagement)하는 하는 것이다. 본 논문에서는 후자 기반의 미니게임을 사용하여 실험을 진행한다. 또 한 에이전트들의 부분관측 정보는 게임 환경에서 직접적으로 주어진다. 본 논문에서는 학습 환경과 수행 환경에서 실험하였고 모두 적군과 아군의 개체 수는 동일하게 설정하였다. 학습 환경은 수행 환경은 기존 SMAC에서 제공하는 미니게임을 아군은 변형이 없고 적군의 등장 위치 및 구성을 변형한 환경으로 수정하였다. 해당 환경은 Starcraft II의 지도 편집기(Map Editor)를 이용하여 수정하였다. 등장 위치는 편집기 상에 임의로 배치했으며 구성 변경은 지도 편집기의 트리거 기능을 이용하여 구현하였다.

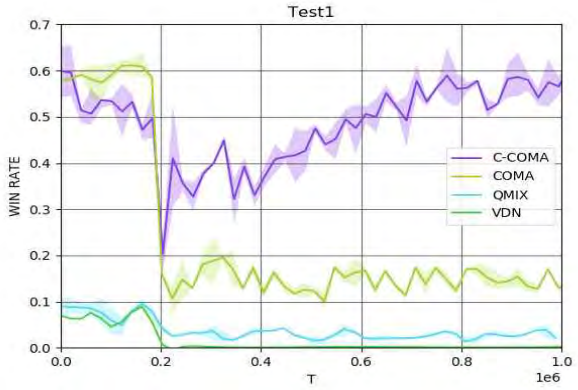
여기서 학습 환경은 위치 및 구성의 일부분만 변경된다. 반대로 수행 환경은 학습 환경에서 경험한 상황에 더 추가적으로 등장 위치와 구성 변형을 하도록 수정하였다. 구체적으로 아군의 유닛은 추적자 두 마리와 광전사 세 마리로 구성되며 적군의 구성은 총 세 가지 유닛으로 광전사, 추적자, 히드라로 구성된다. 총 다섯 마리로써 추적자 두 마리는 고정이며 나머지 세 마리는 광전사와 히드라로 무작위 구성된다. 단 학습 환경에서는 히드라가 한 마리 또는 등장 안 할 수도 있다. 반대로 실행환경에서는 광전사와 히드라의 구성이 에피소드 단위로 무작위 변형된다. 다시 말해 근거리 공격을 하는 광전사가 원거리 공격을 하는 히드라로 변형되면서 문제는 더욱 어려워지며 학습으로 일부는 경험했던 상황이 발생하나 대부분은 경험하지 못했던 상황이 발생하게 된다. 또 한 적군의 경우 학습 기반의 인공지능이 아닌 Starcraft II 내부의 게임 인공지능이며 난이도는 1부터 10까지 설정할 수 있고 본 실험은 난이도 7에서 수행 하였다.

모델 학습을 위하여 조정자 에이전트의 레이어 수(number of layers)는 3이며 일반적인 학습과 달리 지속학습을 위하여 신경망 일부만 업데이트하도록 수정하였다. 행동 에이전트와 조정자 에이전트의 학습률(Learning Rate)은 모두 0.0005로 설정 하였다. 보상을 학습하기 위한 TD(λ)의 값은 0.8로 설정하였으며 타겟 네트워크는 200회 반복마다 갱신하였다. 나머지 Q 학습 계열의 접근법들은 입실론 그리디(ε-greedy)를 이용하며 1로 설정하였다. 추 후 50K 반복마다 값을 감쇄시키며 최소 0.05까지 감쇄된다. 마지막으로 경험에 대한 버퍼크기(buffer size)는 5000으로 설정 하였다. 실험과정 모두 총 8개의 프로세스로 병렬적으로 실험을 수행하였다. 실험은 64GB의 메인 메모리와 Geforce RTX 2080 TI 2개를 포함한 컴퓨터 환경에서 수행되었다.

4.2 성능 평가 실험

본 논문에서는 제안 모델인 C-COMA의 효과와 우수성을 입증하기 위해 Starcraft II 동적 미니게임 환경을 이용한 다양한 실험들을 수행한다. 첫 번째 실험은 본 논문에서 제안하는 지속적인 다중 에이전트 강화 학습 모델인 C-COMA가 지속 학습인 불가능한 기존 모델들에 비해 동적 환경에서 우수성을 입증하는 실험이다. 이 실험을 위해 C-COMA를 비 지속 학습 모델들인 COMA[1], QMIX[2], VDN[4] 등과 비교하였다. 이들은 모두 중앙의 조정자가 팀 단위의 가치 평가를 담당하는 모델들로서, VDN[4]는 각 에이전트의 개별 Q 함수값의 선형 조합으로, QMIX[2]는 비-선형 함수로 각각 팀 전체의 가치함수인 Q를 표현

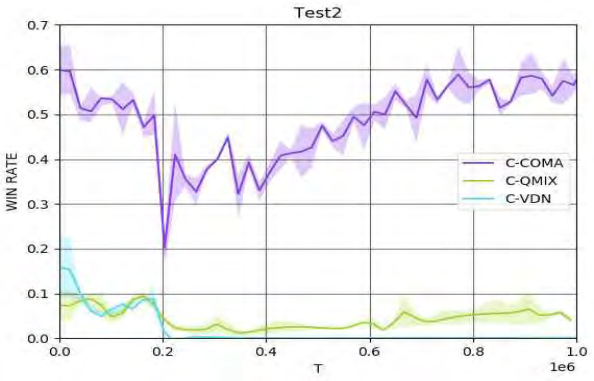
한다. COMA[1]의 경우는 행동자-비평가(actor-critic) 강화 학습을 기초로, 중앙의 조정자는 팀 단위의 가치 함수인 Q를 학습하고, 각 개별 에이전트는 자신의 지역 행동 정책(local policy)를 학습하는 역할을 담당한다. 대신 중앙의 조정자가 에이전트들의 협동 행동(joint action)에 관한 팀 전체 가치를 평가할 때, 반-사실적 추론(countfactual reasoning)을 적용한다. 이 실험에서 모델의 성능 평가 척도로는 총 30개의 에피소드 동안의 평균 승률(win rate)을 사용하였다. 에피소드 단위별로 변화가 일어나는 학습 환경에서 5M만큼의 학습 주기까지 진행 시킨 후, 수행 환경에서 1M 주기까지 테스트 실험을 진행하였다. 단 200K 주기까지는 학습 환경과 동일한 상황들이 발생 되지만, 그 이후로는 이전에 경험하지 못한 새로운 상황들이 동적으로 발생하도록 설정하였다.



(그림 3) 지속적인 강화 학습의 효과

첫 번째 실험 결과는 (그림 3)에서 볼 수 있듯이, 지속학습을 진행 유무에 따라 성능의 차이가 상당하였음을 확인할 수 있다. 200K 반복을 기준으로 경험하지 않은 상황들이 발생하여 지속학습의 여부와 상관없이 모든 접근법들의 성능이 저하되었다. 지속 학습을 진행한 C-COMA의 경우는 이후 성능이 다시 향상되었으나 지속 학습을 진행하지 않은 접근법들은 저하된 성능이 회복되지 않았다. 이는 지속 학습을 진행한 경우 새로운 경험에 대하여 피드백 과정을 수행하기 때문이다. 반대로 비 지속 학습을 진행한 경우 새로운 경험에 효과적으로 대처하지 못하기 때문에 성능이 저하된 것으로 판단된다. 따라서 경험하지 못한 상황이 발생하는 동적 환경에서는 지속학습이 필요하다는 것을 알 수 있다. 그리고 지속학습을 진행하지 않는 경우 COMA의 경우가 나머지 QMIX와 VDN과 비교하여 성능이 우수하다. 그 이유는 팀 보상에 대한 행동 에이전트의 기여도를 비-사실적 추론을 도입한 COMA가 효과적으로 작동한 것으로 판단된다.

두 번째 실험은 COMA에 기초한 지속 학습 모델인 C-COMA가 다른 지속 학습 모델들에 비해 우수성을 입증하는 실험이다. 이 실험에서는 제안 모델인 C-COMA를 QMIX 기반의 C-QMIX, VDN 기반의 C-VDN들과 성능을 비교한다.



(그림 4) 지속 학습 모델 간의 성능 비교

두 번째 실험 결과는 (그림 4)에서 볼 수 있듯이, C-COMA의 경우 성능이 안정적으로 상승하였다. 반면에 C-QMIX의 경우, 승률이 10% 안팎에서 머물렀다. 또 한 C-VDN의 경우 성능 향상을 하지 못했다. 이는 조정자 에이전트의 역할이 중요함을 알 수 있는 결과이다. Q 학습 기반의 조정자들을 보면 C-QMIX의 성능이 C-VDN보다 약간 좋다는 것을 알 수 있다. 이는 행동 에이전트들의 Q 함수를 비선형으로 조합한 조정자 에이전트가 좀 더 효율적으로 행동 에이전트들을 제어한다고 판단된다. 그러나 C-COMA와 비교하면 성능의 차이가 상당하다. 이는 COMA처럼 비-사실적 추론이 행동 에이전트들의 행동을 팀 차원에 맞춰 수행하도록 효과적인 제어를 하였음을 알 수 있다.

다음은 본 논문에서 제안하는 지속적인 다중 에이전트 학습 모델인 C-COMA의 효과를 실제 사례를 통해 분석해보는 정성적 평가를 수행하였다. (그림 5)는 C-COMA의 지속 학습 효과를 잘 보여주는 예이다. 동적 변화가 심한 전투 환경에서는 행동 에이전트들의 시야에서 적군이 탐지가 안 될 수도 있다. 이런 경우 C-COMA는 (그림 5)와 같이 적군의 위치로 다가가서 전투를 진행하였다. 반대로 기존의 다른 다중 에이전트 강화 학습 모델들의 경우는 적군에게 다가가지 못하고 그 자리를 맴도는 경우가 많았다.

(그림 5) 시야 범위 밖에 대한 대응

이것은 행동 에이전트들은 전투 상황 전체를 보지 못하기 때문에, 전체 상황을 볼 수 있는 조정자 에이전트의 역할이 매우 중요하다. 이 같은 상황이 발생하였을 때 C-COMA의 경우만 목적에 맞게 잘 행동하였다. 이는 기존의 다른 다중 에이전트 강화학습 모델들의 조정자 에이전트의 접근 방법에 대한 차이 때문에 발생한 것으로 판단된다. Q 학습의 기반 모델인 C-VDN과 C-QMIX의 경우 행동 에이전트들의 Q 함수 값을 각자의 방법대로 통합한 후 행동 에이전트들의 정책과 상황에 대한 상관관계를 단순하게 분할 하여 조정한다. 이 기법은 정적 환경과 달리 동적 환경과 같이 복잡한 경우 잘 작동하지 않았다. 반면에 C-COMA의 경우는 상황에 따른 다른 행동 에이전트들의 정책과 본인의 정책을 비교하며 가치평가를 하여 행동 에이전트의 정책에 도움을 준다. 이러한 특성의 결과로 행동 에이전트들이 적군의 위치를 탐지하지 못하고 경험해 보지 못한 동적 상황에서 조정자 에이전트가 효과적으로 대응하도록 유도하였다고 판단된다. 따라서 C-COMA의 조정자 에이전트가 VDN, QMIX 등을 토대로 한 다른 지속 다중 에이전트 강화 학습 모델들의 조정자 에이전트보다 팀 승리를 위한 협력적 행동을 하도록 행동 에이전트들의 정책을 더 효과적으로 수립하게 도움을 주었다고 판단된다.

5, 결론

본 논문에서는 동적 환경에 효과적으로 대응하기 위해, 새로운 다중 에이전트 강화 학습 체계인 C-COMA를 제안하였다. C-COMA는 에이전트들의 훈련 시간과 실행 시간을 따로 나누지 않고, 처음부터 실전 상황을 가정하고 지속적으로 에이전트들의 협력적 행동 정책을 학습해나가는 지속적인 다중 에이전트 강화 학습 체계이다. 본 논문에서는 대표적인 실시간 전략 게임인 StarcraftⅡ를 토대로 동적 미니게임을 구현하고 이 환경을 이용한 다양한 실험들을 수행함으로써, 제안 모델인 C-COMA의 효과와 우수성을 입증하였다.

참고문헌

[1] Foerster, J., Farquhar, and G, Afouras, et all. “Counterfactual Multi-Agent Policy Gradients”. *Proc of AAAI,* 2018

[2] Tabish Rashid, Mikayel Samvelyan, and Christian Schroeder Witt, et all. “Qmix: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning”. *Proc of ICML*, 2018.

[3] Mikayel Samvelyan, Tabish Rashid, and Christian Schroeder de Witt, et all. “The StarCraft Multi-Agent Challenge”. CoRR, abs/1902.04043, 2019.

[4] Sunehag, P., Lever, and Gruslys et all. “Value-Decomposition Networks For Cooperative Multi-Agent Learning Based On Team Reward” *Proc of ICAAMS,* 2017.