심층 강화 학습을 위한 비동기식 방법

Volodymyr Mnih1 VMNIH@GOOGLE.COM
Adrià Puigdomènech Badia1 ADRIAP@GOOGLE.COM
Mehdi Mirza1,2 Alex Graves1 MIRZAMOM@IRO.UMONTREAL.CA
Tim Harley1 Timothy P. GRAVESA@GOOGLE.COM
Lillicrap1 David Silver1 Koray 탈리@GOOGLE.COM
Kavukcuoglu COUNTZERO@GOOGLE.COM
DAVIDSILVER@GOOGLE.COM
KORAYK@GOOGLE.COM

1

[1] 구글 딥마인드
[2] 몬트리올 대학교 학습 알고리즘 몬트리올 연구소(MILA)

추상적인

우리는 개념적으로 간단하고 제안
비동기 그라디언트를 사용하는 심층 강화 학습을 위한 경량 프레임
워크
심층 신경망 최적화를 위한 하강
컨트롤러. 우리는 의 비동기 변형을 제시합니다.
4가지 표준 강화 학습 알고리즘
병렬 배우-학습자가 가지고 있음을 보여줍니다.
네 가지 모두를 허용하는 훈련에 대한 안정화 효과
신경망을 성공적으로 훈련시키는 방법
컨트롤러. 최고의 성능을 내는 방법,
배우 비평가의 비동기 변형, 능가
Atari 도메인의 최신 기술
싱글로 반시간 훈련하면서
GPU 대신 멀티 코어 CPU. 뿐만 아니라,
우리는 비동기 배우 비평가가 성공했음을 보여줍니다
다양한 연속 모터 제어
탐색의 새로운 작업뿐만 아니라 문제
시각적 입력을 사용하여 임의의 3D 미로.

1. 소개

심층 신경망은 다음을 수행할 수 있는 풍부한 표현을 제공합니다.
강화 학습(RL) 알고리즘 수행 가능
효과적으로. 그러나 이전에는 다음과 같이 생각되었습니다.
간단한 온라인 RL 알고리즘과 심층
신경망은 근본적으로 불안정했습니다. 대신 알고리즘을 안정화하기 위한 다양한
솔루션이 제안되었습니다 (Riedmiller, 2005; Mnih et al., 2013; 2015;
Van Has selt et al., 2015; Schulman et al., 2015a). 이러한 접근 방식

공통된 아이디어를 공유하십시오: 온라인 RL 에이전트에 의해 반대되
는 관찰된 데이터의 시퀀스는 고정적이지 않으며,

제33회 기계에 관한 국제회의(International Conference on Machine)의 절차
Learning, New York, NY, USA, 2016. JMLR: W&CP 볼륨
48.

라인 RL 업데이트는 강한 상관 관계가 있습니다. 저장함으로써
경험 재생 메모리에 있는 에이전트의 데이터, 데이터는
일괄 처리 (Riedmiller, 2005; Schulman et al., 2015a) 또는
무작위로 샘플링 됨(Mnih et al., 2013; 2015; Van Hasselt
et al., 2015) 다른 시간 단계에서. 이상 집계
이러한 방식으로 메모리는 비정상성을 줄이고 업데이트를 조정하지만 동시에 메서
드를 다음으로 제한합니다.
정책 외 강화 학습 알고리즘.

경험 재생을 기반으로 하는 Deep RL 알고리즘은
도전적인 영역에서 전례 없는 성공 달성
예를 들면 Atari 2600입니다. 그러나 경험 재생에는 몇 가지
단점: 실제당 더 많은 메모리와 계산을 사용합니다.
상호 작용; 정책 외 학습 알고리즘이 필요합니다.
이전 정책에서 생성된 데이터에서 업데이트할 수 있습니다.

이 논문에서 우리는 딥러닝에 대한 매우 다른 패러다임을 제공합니다.
강화 학습. 우리는 경험 재생 대신
환경의 여러 인스턴스에서 여러 에이전트를 병렬로 비동기식으로 실행합니다. 이
병렬화도
에이전트의 데이터를 더 고정된 프로세스로 역상관시킵니다.
주어진 시간 단계에서 병렬 에이전트는 다양한 다른 상태를 경험할 것이기 때문입
니다. 이 간단한 아이디어
기본 on-policy의 훨씬 더 넓은 스펙트럼을 가능하게 합니다.
Sarsa, n-step 방법 및 Actor 비평가 방법과 같은 RL 알고리즘과 다음과 같은
오프 정책 RL 알고리즘
Q-learning으로 강력하고 효과적으로 적용하기 위해
심층 신경망.

병렬 강화 학습 패러다임도 제공합니다.
실용적인 이점. 심층 강화 학습에 대한 이전 접근 방식은 특수 하드웨어에 크게 의
존합니다.
GPU (Mnih et al., 2015; Van Hasselt et al., 2015;
Schhaul et al., 2015) 또는 대규모 분산 아키텍처
(Nair et al., 2015), 우리의 실험은 단일 기계에서 실행됩니다.
표준 멀티 코어 CPU를 사용합니다. 다양한 Atari 2600 도메인에 적용될 때 많
은 게임에서 비동기식
강화 학습은 훨씬 적은 비용으로 더 나은 결과를 얻습니다.

대규모 분산 접근 방식보다 훨씬 적은 리소스를 사용하여 이전 GPU 기반 알고리즘보다 시간이 단축됩니다. 제안된 방법 중 가장 좋은 A3C(Asynchronous Advantage Actor Critic)는 다양한 연속 모터 제어 작업을 마스터했을 뿐만 아니라 순수하게 시각적 입력에서 3D 미로를 탐색하기 위한 일반적인 전략을 배웠습니다. 우리는 2D 및 3D 게임, 이산 및 연속 행동 공간에서 A3C의 성공과 피드포워드 및 반복 에이전트를 훈련하는 능력이 현재까지 가장 일반적이고 성공적인 강화 학습 에이전트가 되었다고 믿습니다.

2. 관련 업무

(Nair et al., 2015) 의 일반 강화 학습 아키텍처(Gorila) 는 분산 환경에서 강화 학습 에이전트의 비동기식 훈련을 수행합니다. Go rila에서 각 프로세스는 환경의 자체 복사본에서 작동하는 액터, 별도의 재생 메모리, 재생 메모리에서 데이터를 샘플링하고 다음을 사용하여 DQN 손실의 기울기를 계산하는 학습자를 포함합니다 (Mnih et al., 2015) . 정책 매개변수를 고려합니다. 기울기는 모델의 중앙 사본을 업데이트하는 중앙 매개변수 서버로 비동기식으로 전송됩니다. 업데이트된 정책 매개변수는 고정된 간격으로 행위자 학습자에게 전송됩니다. 100개의 개별 행위자-학습자 프로세스와 30개의 매개변수 서버, 총 130개의 머신을 사용하여 Gorila는 49개의 Atari 게임에서 DQN을 훨씬 능가할 수 있었습니다. 많은 게임에서 Gorila는 DQN보다 20배 이상 빠르게 DQN이 달성한 점수에 도달했습니다. 우리는 또한 DQN을 병렬화하는 유사한 방법이 (Chavez et al., 2015) 에 의해 제안되었음을 주목합니다 .

이전 작업에서 (Li & Schuurmans, 2011) 선형 함수 근사를 사용하여 일괄 강화 학습 방법을 병렬화하기 위해 Map Reduce 프레임워크를 적용했습니다.

병렬 처리는 대규모 매트릭스 작업의 속도를 높이는 데 사용되었지만 경험 수집을 병렬화하거나 학습을 안정화하는 데 사용되지 않았습니다. (Grounds & Kudenko, 2008) 는 훈련을 가속화하기 위해 여러 개의 개별 행위자 학습자를 사용하는 Sarsa 알고리즘의 병렬 버전을 제안했습니다. 각 액터 학습자는 개별적으로 학습하고 P2P 통신을 사용하여 크게 변경된 가중치에 대한 업데이트를 다른 학습자에게 주기적으로 보냅니다.

(Tsitsiklis, 1994) 는 비동기 최적화 설정에서 Q 학습의 수렴 속성을 연구했습니다. 이러한 결과는 오래된 정보가 항상 결국 폐기되고 몇 가지 다른 기술적 가정이 충족되는 한 일부 정보가 오래된 경우에도 Q-learning이 여전히 수렴된다는 것을 보여줍니다. 더 일찍 (Bertsekas, 1982) 은 분산 동적 프로그래밍과 관련된 문제를 연구했습니다.

또 다른 관련 작업 영역은 여러 기계나 스레드에 적합성 평가를 배포하여 병렬화하기 쉬운 진화적 방법입니다 (Tomassini, 1999). 이러한 병렬 진화 ap

최근 몇 가지 시각적 강화 학습 작업에 접근 방식이 적용되었습니다. 한 가지 예에서 (Koutník et al., 2014) 8개의 CPU 코어에 대한 적합성 평가를 병렬로 수행하여 TORCS 운전 시뮬레이터용 컨볼루션 신경망 컨트롤러를 발전시켰습니다.

3. 강화학습 배경

에이전트가 여러 개별 시간 단계에 걸쳐 환경 E와 상호 작용하는 표준 강화 학습 설정을 고려합니다. 각 시간 단계 t에서 에이전트는 상태 st 를 수신 하고 정책 π 에 따라 가능한 행동 A의 일부 세트에서 행동을 선택합니다. 여기서 π는 상태 st 에서 행동 at으로 의 매핑입니다. 그 대가로 에이전트는 다음 상태 st+1 을 받고 스칼라 보상 rt를 받습니다. 프로세스는 에이전트가 터미널 상태에 도달한 후 프로세스가 다시 시작될 때까지 계속됩니다. 수익 Rt = P∞ 시간 단계 t, 할인 계수 γ ∈ (0, 1] 에이전트의 목표는 각 상태 st에서 기대 수익을 최대화하는 것입니다.

k=0 c rt+k 는 총 누적 수익입니다.

행동 값 Qπ (s, a) = E [Rt|st = s, a]는 상태 s에서 행동 a를 선택하고 정책 π를 따를 때 예상되는 수익입니다. 최적값 함수 Q* (s, a) = maxπ Qπ (s, a)는 상태 s 및 모든 정책에서 달성할 수 있는 조치 a에 대한 최대 조치 값을 제공합니다. 유사하게, 정책 π에 따른 상태 s의 값은 V π (s) = E [Rt|st = s]로 정의되며 단순히 상태 s에서 다음 정책 π에 대한 기대 수익입니다.

가치 기반 모델 프리 강화 학습 방법에서 행동 가치 함수는 신경망과 같은 함수 근사기를 사용하여 표현됩니다. Q(s, a; θ)를 매개변수 θ가 있는 대략적인 행동 가치 함수라고 합시다. θ에 대한 업데이트는 다양한 강화 학습 알고리즘에서 파생될 수 있습니다. 이러한 알고리즘의 한 예는 최적의 행동 가치 함수인 Q* (s, a) ≈ Q(s, a; θ) 를 직접적으로 근사하는 것을 목표로 하는 Q-러닝 입니다. 1단계 Q-학습에서 행동 가치 함수 Q(s, a; θ)의 매개변수 θ는 손실 함수 시퀀스를 반복적으로 최소화하여 학습됩니다. 여기서 i번째 손실 함수는 다음과 같이 정의됩니다.

Li(θi) = E r + γ 최대 질문 , 에이0 ; θi 1) Q(s, a; θi)

어디 상태 s 이후에 발생한 상태입니다.

위의 방법을 1단계 Q 학습이라고 하는 이유는 행동 값 Q(s, a)를 1단계 반환 r + γ maxa0 Q(s a0 ; θ) 쪽으로 업데이트하기 때문입니다. 1단계 방법을 사용할 때의 한 가지 단점은 보상 r을 얻는 것이 보상의 직접적인 영향에만 영향을 미친다는 것입니다. 다른 상태 동작 쌍의 값은 업데이트된 값 Q(s, a)를 통해 간접적으로만 영향을 받습니다.

많은 업데이트가 관련 선행 상태 및 작업에 보상을 전파해야 하므로 학습 프로세스가 느려질 수 있습니다.

보상을 더 빨리 전파하는 한 가지 방법은 n단계 수익을 사용하는 것입니다 (Watkins, 1989; Peng & Williams, 1996).
n-step Q-learning에서 Q(s, a)는 rt + γrt+1 + · · · + γ maxa γ nQ(st+n, a) 로 정의된 n단계 리턴을 향해 업데이트됩니다 . 그 결과 n개 + 의 선행 상태 작업 쌍의 값에 직접적인 영향을 미치는 단일 보상이 발생합니다.
이것은 관련 상태-동작 쌍에 보상을 전파하는 프로세스를 잠재적으로 훨씬 더 효율적으로 만듭니다.

값 기반 방법과 달리 정책 기반 모델 자유 방법은 정책 π(a|s; θ)을 직접 매개변수화하고 E[Rt]에서 일반적으로 대략적인 기울기 상승을 수행하여 매개변수 θ를 업데이트합니다. 그러한 방법의 한 예는 Williams (1992)에 의한 알고리즘의 REINFORCE 계열입니다. 표준 REINFORCE 는 ∇θE[Rt] 의 편향되지 않은 추정값인 ∇θ log π(at|st; θ)Rt 방향으로 정책 매개변수 θ를 업데이트합니다. 기준선 (Williams, 1992) 으로 알려진 상태 bt(st) 의 학습된 함수를 수익에서 빼면 이 추정값의 분산을 줄이는 것이 가능합니다 . 결과 기울기는 ∇θ log π(at|st; θ)(Rt bt(st))입니다.

알고리즘 1 비동기식 1단계 Q 학습 - 각 액터 학습자 스레드에 대한 의사 docode.

// 전역 공유 θ, θ 및 카운터 T = 0이라고 가정합니다.
스레드 스텝 카운터 초기화 t ← 0 대상 네트워크
가중치 초기화 θ ← θ 네트워크 기울기 초기화 dθ ←
0 초기 상태 가져오기 s repeat Q(s, a; θ)에 기반한
-greedy 정책으로 a 조치 취

새로운 상태 s를 수신하고 r을 보상합니다.

y = r + γ maxa0 Q(s , 에이0 ; θ 터미널 s의 경우 0 비터미널 s의 경우

누적 기울기 wrt θ: dθ ← dθ + ∂ θ) ∂ (y Q(s,a;θ))2

0 초 = 초

T ← T + 1 및 t ← t + 1 if T mod

Itarget == 0 then

t mod IAsyncUpdate == 0 또는 s가 터미널
인 경우 대상 네트워크 θ ← θ end를 업데이트하

고 dθ를 사용하여 θ의 비동기 업데이트 를 수행합니다.

명확한 기울기 dθ ← 0. T >
Tmax 까지 종료

가치 함수의 학습된 추정은 일반적으로 기준 bt(st) ≈ V π (st) 로 사용 되어 정책 기울기의 훨씬 더 낮은 분산 추정으로 이어집니다. 근사값 함수가 기준선으로 사용되는 경우 정책 기울기를 조정하는 데 사용되는 양 Rt - bt 는 상태 st 또는 A(at, st) = Q(at , st) V (st), Rt 는 Qπ (at, st) 의 추정치이고 bt 는 V π (st) 의 추정치 이기 때문 입니다. 이 접근 방식은 정책 π가 행위자이고 기준 bt 가 비평가인 행위자-비평 아키텍처로 볼 수 있습니다 (Sutton & Barto, 1998; Degris et al., 2012).

4. 비동기식 RL 프레임워크

이제 우리는 1단계 Sarsa, 1단계 Q 학습, n단계 Q 학습 및 이점 행위자 비평가의 다중 스레드 비동기 변형을 제시합니다. 이러한 방법을 설계하는 목적은 심층 신경망 정책을 큰 리소스 요구 사항 없이 안정적으로 훈련할 수 있는 RL 알고리즘을 찾는 것이었습니다. 기본 RL 방법은 정책에 따른 정책 검색 방법인 Actor-Critic과 정책 외의 가치 기반 방법인 Q-learning과 같이 상당히 다르지만, 우리는 두 가지 주요 아이디어를 사용하여 우리의 설계에 따라 네 가지 알고리즘을 모두 실용적으로 만듭니다. 목표.

첫째, Gorila 프레임워크 (Nair et al., 2015) 와 유사하게 비동기 액터 학습자 를 사용하지만 별도의 시스템과 매개변수 서버를 사용하는 대신 단일 시스템에서 여러 CPU 스레드를 사용합니다. 학습자를 단일 머신에 유지하면 그라디언트 및 매개변수를 전송하는 통신 비용이 제거되고 Hogwild를 사용할 수 있습니다! (Recht et al., 2011) 훈련을 위한 스타일 업데이트.

둘째, 우리는 여러 행위자가

병렬로 실행되는 학습자는 환경의 다른 부분을 탐색할 가능성이 높습니다. 또한 이러한 다양성을 극대화하기 위해 각 행위자-학습자에서 서로 다른 탐색 정책을 명시적으로 사용할 수 있습니다. 서로 다른 스레드에서 서로 다른 탐색 정책을 실행함으로써 온라인 업데이트를 병렬로 적용하는 여러 액터-학습자가 매개변수에 대해 수행하는 전체 변경 사항이 온라인 업데이트를 적용하는 단일 에이전트보다 시간상 덜 관련될 가능성이 높습니다.

따라서 우리는 재생 메모리를 사용하지 않고 DQN 훈련 알고리즘에서 경험 재생이 수행하는 안정화 역할을 수행하기 위해 다른 탐색 정책을 사용하는 병렬 행위자에 의존합니다.

학습을 안정화하는 것 외에도 여러 병렬 액터 학습자를 사용하면 여러 가지 실용적인 이점이 있습니다. 첫째, 병렬 액터 학습자의 수에서 대략 선형인 훈련 시간의 감소를 얻습니다. 둘째, 학습 안정화를 위해 경험 재생에 더 이상 의존하지 않기 때문에 Sarsa 및 Actor-Critic과 같은 정책 기반 강화 학습 방법을 사용하여 안정적인 방식으로 신경망을 훈련할 수 있습니다. 이제 우리는 1단계 Q 학습, 1단계 Sarsa, n단계 Q 학습 및 이점 행위자 비평가의 변형을 설명합니다.

Asynchronous one-step Q-learning: Asynchronous one-step Q-learning 이라고 부르는 Q-learning의 변형에 대한 의사 코드는 알고리즘 1에 나와 있습니다. Q 학습 손실의 기울기. 우리는 DQN 훈련 방법에서 제안한 것처럼 Q-학습 손실을 계산할 때 공유되고 천천히 변화하는 대상 네트워크를 사용합니다. 우리는 또한 적용되기 전에 여러 시간 단계에 걸쳐 그라디언트를 축적합니다. 이는 우리와 유사합니다.

미니배치. 이렇게 하면 여러 배우 학습자가 서로의 업데이트를 덮어쓸 가능성이 줄어듭니다. 여러 단계에 걸쳐 업데이트를 누적하면 데이터 효율성을 위해 계산 효율성을 절충할 수 있는 기능도 제공됩니다.

마지막으로 각 스레드에 서로 다른 탐색 정책을 부여하면 견고성을 향상시키는 데 도움이 됩니다. 이러한 방식으로 탐색에 다양성을 추가하면 일반적으로 더 나은 탐색을 통해 성능이 향상됩니다. 탐색 정책을 다르게 만드는 많은 가능한 방법이 있지만 각 스레드의 일부 배포에서 주기적으로 샘플링되는 -greedy 탐색을 사용하여 실험합니다.

비동기식 1단계 Sarsa: 비동기식 1단계 Sarsa 알고리즘은 Q(s, a)에 대해 다른 목표 값을 사용한다는 점을 제외하고 알고리즘 1 에 제공된 비동기식 1단계 Q 학습과 동일합니다. 1단계 Sarsa에서 사용하는 목표값은 r + γQ(s a0 ; θ -)이며, 여기서 a는 상태 s에서 취한 조치입니다 (Rummery & Niranjan, 1994; Sutton & Barto, 1998) 여러 [illegible] 를 다시 사용합니다.

비동기식 n단계 Q 학습: 다단계 Q 학습 변형에 대한 의사 코드는 보충 알고리즘 S2에 나와 있습니다. 알고리즘은 적격성 추적 (Sutton & Barto, 1998) 과 같은 기술에서 사용되는 보다 일반적인 역방향 보기와 달리 n 단계 수익을 명시적으로 계산하여 전방 보기에서 작동하기 때문에 다소 이례적 입니다. 우리는 운동량 기반 방법과 시간에 따른 역전파로 신경망을 훈련할 때 전방 보기를 사용하는 것이 더 쉽다는 것을 발견했습니다. 단일 업데이트를 계산하기 위해 알고리즘은 먼저 최대 tmax 단계 또는 최종 상태에 도달할 때까지 탐색 정책을 사용하여 작업을 선택합니다. 이 프로세스를 통해 에이전트 는 마지막 업데이트 이후 환경에서 최대 tmax 보상을 받습니다. 그런 다음 알고리즘은 마지막 업데이트 이후 발생한 각 상태-동작 쌍에 대한 n-단계 Q 학습 업데이트에 대한 기울기를 계산합니다. 각 n-단계 업데이트는 가능한 가장 긴 n-단계 반환을 사용하여 마지막 상태에 대한 1단계 업데이트, 두 번째 마지막 상태에 대한 2단계 업데이트 등 총 tmax 업데이트에 대해 계속됩니다. 누적된 업데이트는 단일 그래디언트 단계에서 적용됩니다.

Asynchronous Advantage Actor-Critic: A3C(비동기적 이점 행위자 비평가)라고 하는 알고리즘은 정책 π(at|st; θ) 및 가치 함수 V (st; θv)의 추정치를 유지합니다. n-step Q-learning의 변형과 마찬가지로 Actor-Critic의 변형도 전방 보기에서 작동하며 동일한 n-step 반환 조합을 사용하여 정책과 가치 기능을 모두 업데이트합니다. 정책 및 값 함수는 모든 tmax 작업 후 또는 터미널 상태에 도달할 때 업데이트됩니다. 알고리즘에 의해 수행된 업데이트는 ∇θ 0 log π(at|st; θ 여기서 A(st, at; θ, θv) 는 Pk 1 i=0 γ 로 주어진 이점 함수의 추정치입니다. 여기서 k는 상태마다 다를 수 있으며 상한입니다.

)A(st, at; θ, θv)

rt+i + γ kV (st+k, θv) - V (st, θv),

티맥스 로 알고리즘에 대한 의사 코드는 보충 알고리즘 S3에 나와 있습니다.

가치 기반 방법과 마찬가지로 우리는 병렬 액터 학습자와 학습 안정성 향상을 위해 누적된 업데이트에 의존합니다. 정책의 매개변수 θ와 가치 함수의 θv 는 일반성을 위해 별도의 것으로 표시되지만 실제로는 일부 매개변수를 항상 공유합니다. 우리는 일반적으로 정책 π(at|st; θ) 에 대한 하나의 softmax 출력 과 값 함수 V (st; θv)에 대한 하나의 선형 출력을 가지며 모든 비출력 레이어를 공유하는 합성곱 신경망을 사용합니다.

우리는 또한 정책 π의 엔트로피를 목적 함수에 추가하면 최적이 아닌 결정론적 정책으로의 조기 수렴을 방지함으로써 탐색이 향상된다는 것을 발견했습니다. 이 기술은 원래 (Williams & Peng, 1991)에 의해 제안되었으며, 계층적 행동이 필요한 작업에 특히 유용하다는 것을 발견했습니다. 정책 매개변수에 대한 엔트로피 정규화 항을 포함하는 전체 목적 함수의 기울기는 ∇θ 0 log π(at|st; θ β∇θ 0H(π(st; θ 라미터 β) 엔트로피 정규화 항.

)(Rt V (st; θv)) +
)), 여기서 H는 엔트로피입니다. 하이퍼파

최적화: 우리는 비동기 프레임워크에서 세 가지 최적화 알고리즘을 조사했습니다. 모멘텀이 있는 SGD, 공유 통계가 없는 RMSProp (Tieleman & Hinton, 2012) , 공유 통계가 있는 RMSProp입니다. 우리는 다음에 의해 주어진 표준 중심이 아닌 RMSProp 업데이트를 사용했습니다.

$$g = \alpha g + (1 - \alpha)\Delta\theta^2 \quad \text{및} \quad \theta \leftarrow \theta - \eta \frac{\Delta\theta}{\sqrt{g + \varepsilon}}, \qquad (1)$$

여기서 모든 작업은 요소별로 수행됩니다. Atari 2600 게임의 하위 집합에 대한 비교는 통계 g가 스레드 간에 공유되는 RMSProp의 변형이 다른 두 가지 방법보다 훨씬 더 강력하다는 것을 보여주었습니다.

방법 및 비교에 대한 자세한 내용은 보충 섹션 7에 포함되어 있습니다.

5. 실험

제안된 프레임워크의 속성을 평가하기 위해 4가지 다른 플랫폼을 사용합니다. 우리는 대부분의 실험 을 Atari 2600 게임용 시뮬레이터를 제공하는 아케이드 학습 환경 (Bellemare et al., 2012)을 사용하여 수행합니다. 이것은 RL 알고리즘에 대해 가장 일반적으로 사용되는 벤치마크 환경 중 하나입니다. 우리는 Atari 도메인을 사용하여 최신 결과와 비교합니다 (Van Hasselt et al., 2015; Wang et al., 2015; Schaul et al., 2015; Nair et al., 2015; Mnih et al., 2015) . , 뿐만 아니라 제안된 방법의 세부적인 안정성 및 확장성 분석을 수행합니다.

우리는 TORCS 3D 자동차 경주 시뮬레이터를 사용하여 추가 비교를 수행했습니다 (Wymann et al., 2013). 우리는 또한 사용

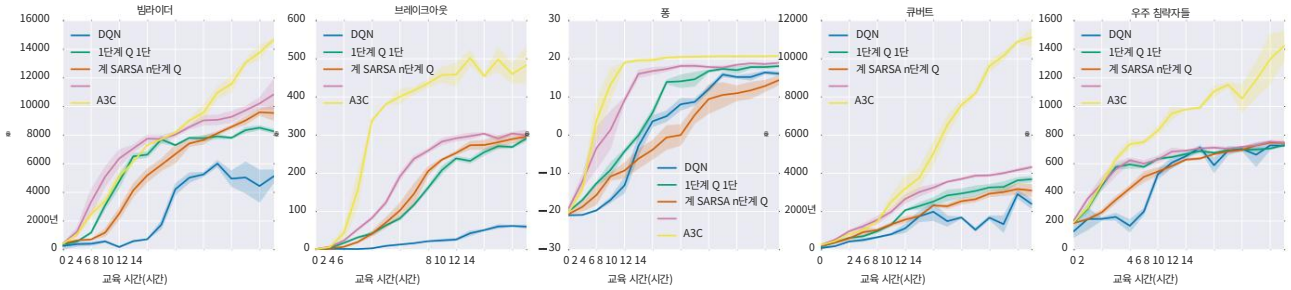

그림 1. 5개의 Atari 2600 게임에서 DQN과 새로운 비동기 알고리즘에 대한 학습 속도 비교. DQN은 단일 Nvidia K40 GPU에서 훈련된 반면 비동기 방식은 16개의 CPU 코어를 사용하여 훈련되었습니다. 플롯의 평균은 5회 실행됩니다. DQN의 경우 실행은 고정된 하이퍼파라미터가 있는 다른 시드에 대한 것이었습니다. 비동기식 방법의 경우 LogUniform(10 4 , 10 2) 에서 샘플링된 학습률 과 고정된 다른 모든 하이퍼파라미터를 사용하여 50개의 실험에서 최고의 5개 모델에 대해 평균을 냅니다.

A3C 알고리즘인 Mujoco 및 Labyrinth만 평가하기 위한 두 개의 추가 도메인. MuJoCo (Todorov, 2015) 는 접촉 역학을 사용하여 연속 모터 제어 작업에서 에이전트를 평가하기 위한 물리학 시뮬레이터입니다. Labyrinth는 에이전트가 시각적 입력에서 무작위로 생성된 미로에서 보상을 찾는 방법을 배워야 하는 새로운 3D 환경입니다. 실험 설정의 정확한 세부 정보는 보충 섹션 8에서 찾을 수 있습니다.

방법	훈련 시간 DQN	평균 중앙값
GPU 8일 121.9% 47.5% Gorila 4일 215.2% 71.3% D-DQN GPU 8일 332.9% 110.9% D-DQN 듀얼 GPU 8일 343.8% D-DQN GPU 8일 유선상의 117.1% 465.6% 127.6% A3C, FF CPU에서 1일 344.1% 68.2% A3C, FF CPU에서 4일 496.8% 116.6% A3C, LSTM CPU에서 4일 623.0%		

표 1. 57 Atari의 인간 정규화 평균 및 중앙값
휴먼 스타트 평가 메트릭을 사용하는 게임. 보충 표 SS3 은 모든 게임의 원 점수를 보여줍니다.

5.1. 아타리 2600 게임

새로운 방법의 훈련 속도를 보여주기 위해 먼저 Atari 2600 게임의 하위 집합에 대한 결과를 제시합니다. 그림 1 은 Nvidia K40 GPU에서 훈련된 DQN 알고리즘의 학습 속도와 5개의 Atari 2600 게임에서 16개의 CPU 코어를 사용하여 훈련된 비동기식 방법을 비교합니다. 결과는 우리가 제시한 4가지 비동기 방식 모두가 Atari 도메인에서 신경망 컨트롤러를 성공적으로 훈련할 수 있음을 보여줍니다. 비동기식 방법은 DQN보다 빠르게 학습하는 경향이 있으며 일부 게임에서는 훨씬 더 빠르게 학습하는 반면 16개의 CPU 코어에서만 학습합니다. 또한 결과는 일부 게임에서 n단계 방법이 1단계 방법보다 더 빨리 학습함을 시사합니다. 전반적으로 정책 기반 이점 행위자 비평가 방법은 세 가지 가치 기반 방법을 모두 훨씬 능가합니다.

그런 다음 우리는 57개의 Atari 게임에서 비동기 이점 배우-비평가를 평가했습니다. Atari 게임 플레이의 최신 기술과 비교하기 위해 (Van Hasselt et al., 2015) 훈련 및 평가 프로토콜을 크게 따랐습니다.

특히, 6개의 Atari 게임(Beamrider, Breakout, Pong, Q*bert, Seaquest 및 Space Invaders)에 대한 검색을 사용하여 하이퍼파라미터(학습 속도 및 기울기 놈 클리핑 양)를 조정한 다음 모든 57개 게임에 대한 모든 하이퍼파라미터를 수정했습니다. 우리는 (Mnih et al., 2015; Nair et al., 2015; Van Hasselt et al., 2015) 동일한 아키텍처를 가진 피드포워드 에이전트와 최종 숨겨진 후 추가 256 LSTM 셀이 있는 순환 에이전트를 모두 훈련했습니다. 층. 결과를 원래 결과와 더 비교할 수 있도록 평가를 위해 최종 네트워크 가중치를 추가로 사용했습니다.

(Bellemare et al., 2012). 우리는 16개의 CPU 코어를 사용하여 에이전트를 4일 동안 교육했으며 다른 에이전트는 Nvidia K40 GPU에서 8~10일 동안 교육했습니다. 표 1 은 A3C(Asynchronous Advantage Actor-Critic)와 최신 기술에 의해 훈련된 에이전트가 얻은 평균 및 중간 인간 정규화 점수를 보여줍니다.

보충 표 S3 은 모든 게임의 점수를 보여줍니다.
A3C는 16개의 CPU 코어만 사용하고 GPU는 사용하지 않으면서 다른 방법의 훈련 시간의 절반으로 57개 게임 이상의 최신 평균 점수를 크게 향상시킵니다. Fur thermore, 단 하루의 훈련 후 A3C는 Dueling Double DQN의 평균 인간 정규화 점수와 일치하고 Gorila의 중간 인간 정규화 점수에 거의 도달합니다. Double DQN (Van Hasselt et al., 2015) 및 Dueling Double DQN (Wang et al., 2015) 에 제시된 많은 개선 사항은 제시된 1단계 Q 및 n단계 Q 방법에 통합될 수 있습니다 . 이 작업에서 유사한 잠재적 개선 사항이 있습니다.

5.2. TORCS 자동차 경주 시뮬레이터

또한 TORCS 3D 자동차 경주 게임에서 4가지 비동기 방식을 비교했습니다 (Wymann et al., 2013).
TORCS는 Atari 2600 게임보다 더 사실적인 그래픽을 제공할 뿐만 아니라 에이전트가 제어하는 자동차의 역학을 배워야 합니다. 각 단계에서 에이전트는 RGB 이미지 형식의 시각적 입력만 받았습니다.

현재 프레임에 비례하는 보상
에이전트의 트랙 중심을 따라 에이전트의 속도
현재 위치. 우리는 에 명시된 Atari 실험에서 사용된 것과 동일한 신경망
아키텍처를 사용했습니다.
보충 섹션 8. 느린 자동차를 제어하는 에이전트의 네 가지 설정을 사용하
여 실험을 수행했습니다.
상대 봇이 있거나 없는 상태에서 에이전트가
상대 봇이 있거나 없는 빠른 자동차. 전체 결과는
보충 그림 S6에서 찾을 수 있습니다. A3C는 성형제당 최고로 약 75%에
서 90% 사이에 도달했습니다.
인간 테스터가 약 12시간의 훈련으로 4가지 게임 구성 모두에 대해 얻은
점수입니다. 보여주는 동영상
A3C 에이전트의 학습된 운전 행동을 찾을 수 있습니다.
https://youtu.be/0xo1Ldx3L5Q 에서 .

5.3. MuJoCo를 사용한 연속 동작 제어
물리학 시뮬레이터

우리는 또한 작업 공간이
연속적이다. 특히, 우리는 엄밀한 세트를 살펴보았습니다.
접촉 역학이 있는 신체 물리 영역
작업에는 조작 및 로코 모션의 많은 예가 포함됩니다. 이러한 작업은
Mujoco를 사용하여 시뮬레이션되었습니다.
물리 엔진. 가치 기반과 달리 비동기식 advantage Actor-Critic 알고리
즘만 평가했습니다.
방법을 사용하면 연속 작업으로 쉽게 확장됩니다. 모두에서
Asynchronous Advantage-Critic은 물리적 상태나 픽셀을 입력으로
사용하여 좋은 솔루션을 찾았습니다.
24시간 미만의 교육 및 일반적으로 몇 시간 이내에
시간. 에이전트가 학습한 몇 가지 성공적인 정책은
다음 비디오에서 볼 수 있습니다 https://youtu.be/
Ajjc08-iPx8. 이 실험에 대한 자세한 내용은
보충 섹션 9에서 찾을 수 있습니다.

5.4. 미궁

A3C로 추가 실험을 수행했습니다.
Labyrinth라는 새로운 3D 환경에서 구체적인
무작위로 생성된 미로에서 보상을 찾는 방법을 학습하는 에이전트
가 포함된 것으로 우리가 고려한 작업입니다. 의 시작 부분에
각 에피소드마다 에이전트는 방과 복도로 구성된 무작위로 생성된 새로운
미로에 배치되었습니다. 각 미로
에이전트가 보상을 받은 두 가지 유형의 객체를 포함했습니다.
찾기 - 사과와 포털. 사과를 줍게 된 계기
보상 1. 포털에 입장하면 보상 10
에이전트가 새로운 무작위 위치에서 리스폰되었습니다.
미로와 이전에 수집된 모든 사과가 재생되었습니다. 에피소드는 60초 후
에 종료됩니다.
새로운 에피소드가 시작됩니다. 에이전트의 목적은 수집하는 것입니다.
제한 시간에 최대한 많은 포인트와 최적의
전략은 먼저 포털을 찾은 다음 반복적으로
각 리스폰 후 다시 돌아갑니다. 이 작업은 훨씬 더
TORCS 구동 도메인보다 도전적이기 때문에
에이전트는 각 에피소드에서 새로운 미로에 직면하고 있어야 합니다.
무작위 미로를 탐험하기 위한 일반적인 전략을 배웁니다.

	스레드 수				
방법	1	2 4 1.0		8	16
1단계 Q 1	3.0 6.3 13.3 24.1				
단계 SARSA 1.0 2.8 5.9 13.1 22.1					
n단계 Q 1.0 2.7 5.9 10.7 17.2					
A3C	1.0 2.1		3.7	6.9	12.5

표 2. 7개의 Atari 게임에서 평균한 각 방법 및 스레드 수에 대한 평균 교육 속도 향상. 계
산하려면
단일 게임에서 훈련 속도 향상을 위해 각 방법과 스레드 수를 사용하여 고정 참조 점수에
도달하는 데 필요한 시간을 측정했습니다. 게임에서 n개의 스레드를 사용하여 속도가 향
상되었습니다.
를 사용하여 고정된 참조 점수에 도달하는 데 필요한 시간으로 정의
하나의 스레드를 기준 점수에 도달하는 데 필요한 시간으로 나눈 값
n 스레드를 사용합니다. 다음 표는 평균 속도 향상을 보여줍니다.
7개의 아타리 게임(Beamrider, Breakout, Enduro, Pong, Q*bert,
Seaquest 및 Space Invaders).

우리는 이 작업에 대해서만 A3C LSTM 에이전트를 훈련했습니다.
84 × 84 RGB 이미지를 입력으로 사용합니다. 최종 평균 점수
약 50은 에이전트가

시각적 입력. 이전에 볼 수 없었던 미로를 탐험하는 에이전트 중 하나를
보여주는 비디오가 https에 포함되어 있습니다.
//youtu.be/nMR5mjCFZCw.

5.5. 확장성 및 데이터 효율성

제안된 프레임워크의 효율성을 분석했습니다.
학습 시간과 데이터 효율성을 살펴봄으로써
병렬 액터 학습자의 수에 따라 변경됩니다. 언제
여러 작업자를 병렬로 사용하고 공유 업데이트
이상적인 경우, 주어진 모델에 대해
작업 및 알고리즘, 달성할 교육 단계 수
특정 점수는 다양한 수의 작업자와 동일하게 유지됩니다. 따라서 이점은
전적으로
시스템에서 더 많은 데이터를 소비할 수 있기 때문에
동일한 양의 벽시계 시간 및 개선 가능
탐구. 표 2 는 달성된 훈련 속도 향상을 보여줍니다.
7개의 Atari 게임에서 평균적으로 병렬 배우-학습자의 수를 증가시켜 사용
합니다. 이 결과는 모든
4가지 방법은 다중 작업자 스레드를 사용하여 상당한 속도 향상을 달성하
며, 16개 스레드는
속도 향상의 순서. 이것은 우리가 제안한 프레임워크가 병렬 수에 따라 잘
확장된다는 것을 확인시켜줍니다.
자원을 효율적으로 사용하는 작업자.

다소 놀랍게도 비동기식 원스텝 Q-러닝
Sarsa 알고리즘은 다음과 같은 초선형 속도 향상을 나타냅니다.
순전히 계산상의 이득으로 설명할 수 없습니다. 우리
1단계 방법(1단계 Q 및 1단계
Sarsa)는 종종 특정 점수를 얻기 위해 더 적은 데이터를 필요로 합니다.
더 많은 병렬 액터 학습자를 사용할 때. 우리는 이것을 믿습니다
다중 스레드를 줄이는 긍정적인 효과 때문입니다.
원스텝 방식의 편향. 이러한 효과가 더 많이 나타납니다.
평균 점수의 플롯을 보여주는 그림 3에서 명확하게
서로 다른 훈련 프레임의 총 수에 대해

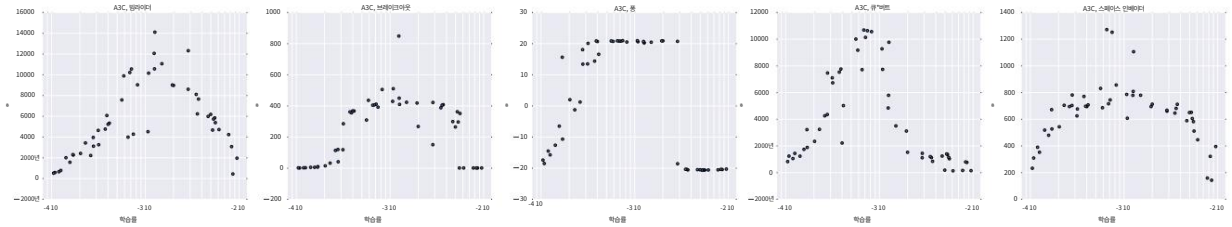

그림 2. 50가지 다른 학습률과 무작위 초기화에 대해 5가지 게임(Beamrider, Breakout, Pong, Q*bert, Space Invaders)에서 비동기 이점 배우 비평가가 얻은 점수의 산점도. 각 게임에는 모든 무작위 초기화가 좋은 점수를 얻는 광범위한 학습률이 있습니다. 이것은 A3C가 학습률과 초기 무작위 가중치에 대해 매우 강력하다는 것을 보여줍니다.

5개의 Atari 게임에 대한 배우 학습자의 수와 훈련 방법, 그리고 벽시계 시간에 대한 평균 점수의 플롯을 보여주는 그림 4.

5.6. 견고성과 안정성

마지막으로 제안된 4가지 비동기 알고리즘의 안정성과 견고성을 분석했습니다. 4가지 알고리즘 각각에 대해 50가지 다른 학습률과 무작위 초기화를 사용하여 5가지 게임(Break out, Beamrider, Pong, Q*bert, Space Invaders)에서 모델을 훈련했습니다. 그림 2 는 A3C에 대한 결과 점수의 산점도를 보여주고 보충 그림 S11 은 다른 세 가지 방법에 대한 플롯을 보여줍니다. 일반적으로 각 방법과 게임 조합에 대해 다양한 학습률이 있어 좋은 점수를 얻습니다. 이는 모든 방법이 학습률 및 무작위 초기화 선택에 대해 매우 강력함을 나타냅니다. 학습률이 좋은 영역에 점수가 0인 포인트가 거의 없다는 사실은 방법이 안정적이며 일단 학습되면 붕괴되거나 발산하지 않는다는 것을 나타냅니다.

6. 결론 및 논의

우리는 4가지 표준 강화 학습 알고리즘의 비동기 버전을 제시했으며 안정적인 방식으로 다양한 도메인에서 신경망 컨트롤러를 훈련할 수 있음을 보여주었습니다. 우리의 결과는 우리가 제안한 프레임워크에서 강화 학습을 통한 신경망의 안정적인 훈련이 가치 기반 및 정책 기반 방법, 정책 방법뿐만 아니라 정책 방법과 이산 및 연속 영역 모두에서 가능하다는 것을 보여줍니다. 16개의 CPU 코어를 사용하여 Atari 도메인에서 훈련할 때 제안된 비동기식 알고리즘은 Nvidia K40 GPU에서 훈련된 DQN보다 더 빠르게 훈련하며 A3C sur는 훈련 시간의 절반으로 현재 최첨단을 통과합니다.

우리의 주요 발견 중 하나는 병렬 행위자 학습자를 사용하여 공유 모델을 업데이트하는 것이 우리가 고려한 세 가지 가치 기반 방법의 학습 프로세스에 안정화 효과가 있다는 것입니다. 이는 DQN에서 이를 위해 사용한 경험 리플레이 없이 안정적인 온라인 Q-learning이 가능함을 보여주지만 경험 리플레이가 유용하지 않다는 의미는 아니다. 경험 재생을 비동기 강화 학습 프레임워크에 통합하면 오래된 데이터를 재사용하여 이러한 방법의 데이터 효율성을 크게 향상시킵니다. 이는 환경과 상호 작용하는 것이 우리가 사용한 아키텍처에 대한 모델을 업데이트하는 것보다 비용이 많이 드는 TORCS와 같은 도메인에서 훨씬 더 빠른 교육 시간으로 이어질 수 있습니다.

기존의 다른 강화 학습 방법이나 심층 강화 학습의 최근 발전을 비동기 프레임워크와 결합하면 우리가 제시한 방법을 즉시 개선할 수 있는 많은 가능성이 있습니다. 우리의 n-step 방법은 수정된 n-step 턴을 직접 목표로 사용하여 전방 보기 (Sutton & Barto, 1998) 에서 작동하지만 적격 추적 (Watkins) 을 통해 서로 다른 수익을 암시적으로 결합하기 위해 후방 보기를 사용하는 것이 더 일반적이었습니다. , 1989; Sutton & Barto, 1998; Peng & Williams, 1996). 비동기식 이점 행위자 비평 방법은 일반화된 이점 추정과 같은 이점 기능을 추정하는 다른 방법을 사용하여 잠재적으로 개선될 수 있습니다 (Schulman et al., 2015b). 우리가 조사한 모든 가치 기반 방법은 Q-값의 과대 추정 편향을 줄이는 다양한 방법의 이점을 얻을 수 있습니다 (Van Hasselt et al., 2015; Belle mare et al., 2016). 보다 추측적인 또 다른 방향은 실제 온라인 시간차 방법 (van Seijen et al., 2015) 에 대한 최근 작업을 비선형 함수 근사와 결합하려고 시도하고 결합하는 것입니다.

이러한 알고리즘 개선 외에도 신경망 아키텍처에 대한 여러 보완 개선이 가능합니다. (Wang et al., 2015) 의 결투 아키텍처 는 네트워크의 상태 가치와 이점에 대한 별도의 스트림을 포함하여 Q-값의 보다 정확한 추정치를 생성하는 것으로 나타났습니다. (Levine et al., 2015) 이 제안한 공간적 소프트 맥스 는 네트워크가 특징 좌표를 더 쉽게 나타낼 수 있도록 하여 가치 기반 방법과 정책 기반 방법을 모두 개선할 수 있습니다.

감사의 말

논문에 대한 많은 유용한 토론, 제안 및 논평을 해주신 Thomas Degris, Remi Munos, Marc Lanctot, Sasha Vezhnevets 및 Joseph Modayil에게 감사드립니다. 또한 논문에서 에이전트를 평가하는 데 사용되는 환경을 설정해 준 DeepMind 평가 팀에게도 감사합니다.

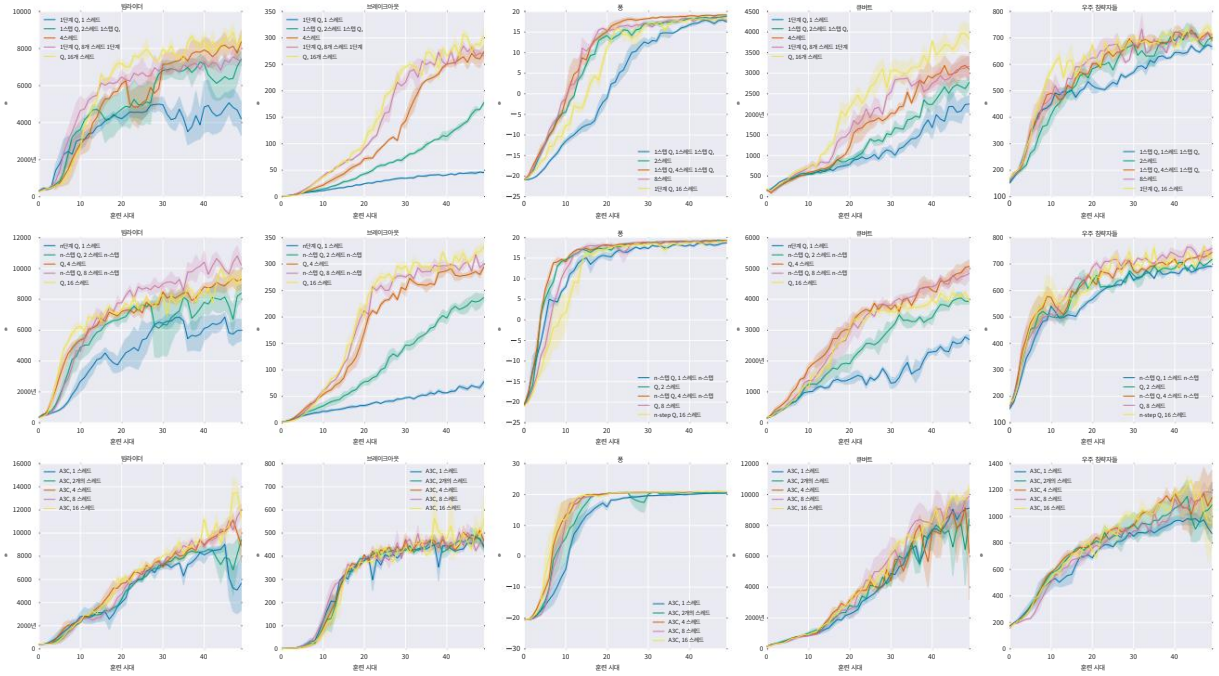

그림 3. 5가지 Atari 게임에서 3가지 비동기 방식에 대한 다양한 액터 학습자 수의 데이터 효율성 비교. x축은 에포크가 400만 프레임(모든 스레드에서)에 해당하는 훈련 에포크의 총 수를 보여줍니다. y축은 평균 점수를 보여줍니다. 각 곡선은 세 가지 최고의 학습률에 대한 평균을 보여줍니다. 단일 단계 방법은 더 많은 병렬 작업자로부터 데이터 효율성이 증가함을 보여줍니다. Sarsa에 대한 결과는 보충 그림 S9에 나와 있습니다.

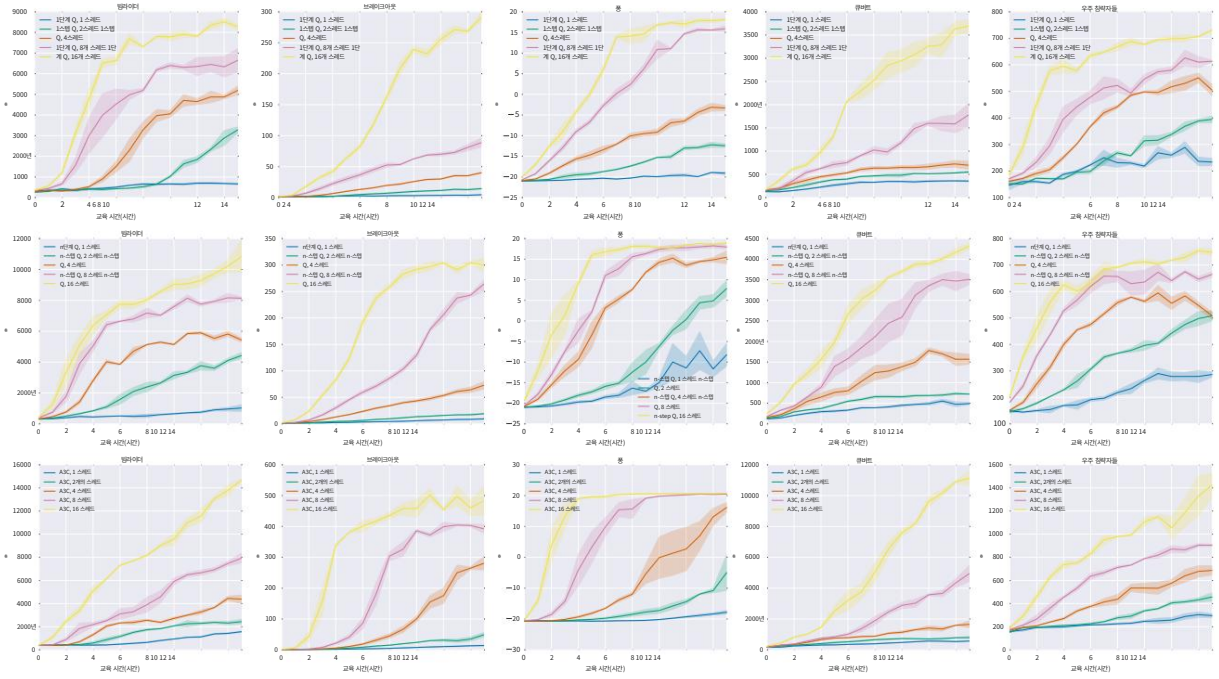

그림 4. 5개의 Atari 게임에서 다양한 배우-학습자의 교육 속도 비교. x축은 훈련 시간을 시간 단위로 표시하고 y축은 평균 점수를 표시합니다. 각 곡선은 세 가지 최고의 학습률에 대한 평균을 보여줍니다. 모든 비동기식 방법은 더 많은 병렬 액터 학습자를 사용함으로써 상당한 속도 향상을 보여줍니다. Sarsa에 대한 결과는 보충 그림 S10에 나와 있습니다.

참고문헌

Bellemare, Marc G, Naddaf, Yavar, Veness, Joel 및
볼링, 마이클. 아케이드 학습 환경:
일반 에이전트를 위한 평가 플랫폼입니다. 저널
인공 지능 연구, 2012.

Bellemare, Marc G., Ostrovski, Georg, Guez, Arthur,
Thomas, Philip S. 및 Munos, Rémi. 액션 갭 증가: 강화 학습을 위
한 새로운 연산자. ~ 안에
2016년 인공 인텔 라이젠스에 관한 AAAI 회의의 절차.

Bertsekas, Dimitri P. 분산 동적 프로그래밍.
자동 제어, IEEE 트랜잭션 on, 27(3):610–
616, 1982.

차베스, 케빈, 옹, 하오이, 아우구스투스 홍. 분산 딥 q-러닝. 기술 보고서,
Stanford University, 2015년 6월.

Degris, Thomas, Pilarski, Patrick M, Sutton, Richard S.
실습에서 지속적인 행동을 통한 모델 없는 강화 학습. ACC(American
Control Conference)에서는
2012, pp. 2177-2182. IEEE, 2012.

Grounds, Matthew 및 Kudenko, Daniel. 선형 함수 근사를 사용한 병
렬 강화 학습.
적응형 및 학습 에이전트 및 다중 에이전트에 대한 5차, 6차 및 7차 유럽
회의의 절차에서
시스템: 적응 및 다중 에이전트 학습, pp. 60–
74. Springer Verlag, 2008.

Koutnik, Jan, Schmidhuber, Jürgen 및 Gomez, Faustino.
진화하는 심층 비지도 컨볼루션 네트워크
비전 기반 강화 학습. 의 절차에서
유전 및 진화 계산에 관한 2014년 회의, pp. 541–548. ACM, 2014.

Levine, Sergey, Finn, Chelsea, Darrell, Trevor, Abbeel,
피터. 심층 시각 운동 정책의 종단 간 교육.
arXiv 사전 인쇄 arXiv:1504.00702, 2015.

Li, Yuxi 및 Schuurmans, Dale. 병렬 강화 학습을 위한 Mapreduce. 강
화 학습의 최근 발전 - 9차 유럽 워크숍, EWRL 2011,

아테네, 그리스, 2011년 9월 9-11일, 선택 사항 수정
논문, pp. 309–320, 2011.

Lillicrap, Timothy P, Hunt, Jonathan J, Pritzel, Alexander,
Heess, Nicolas, Erez, Tom, Tassa, Yuval, Silver, David,
그리고 Wierstra, Daan. 심층 강화 학습을 통한 지속적인 제어. arXiv
사전 인쇄 arXiv:1509.02971,
2015.

Mnih, Volodymyr, Kavukcuoglu, Koray, Silver, David,
그레이브스, 알렉스, 안토노글루, 이오아니스, 위어스트라, 단,
리드밀러, 마틴. 심층 강화 학습으로 아타리 연주하기. NIPS 딥 러닝 워
크샵에서. 2013.

Mnih, Volodymyr, Kavukcuoglu, Koray, Silver, David,
Rusu, Andrei A., Veness, Joel, Bellemare, Marc G.,
Graves, Alex, Riedmiller, Martin, Fidjeland, Andreas K.,
Ostrovski, Georg, Petersen, Stig, Beattie, Charles, Sadik,
Amir, Antonoglou, Ioannis, King, Helen, Kumaran,
Dharshan, Wierstra, Daan, Legg, Shane 및 Hassabis,
데미스. 심층 강화를 통한 인간 수준 제어
학습. Nature, 518(7540):529–533, 02 2015. URL
http://dx.doi.org/10.1038/nature14236.

Nair, Arun, Srinivasan, Praveen, Blackwell, Sam, Alci cek,
Cagdas, Fearon, Rory, Maria, Alessandro De, Pan neerselvam,
Vedavyas, Suleyman, Mustafa, Beattie,
Charles, Petersen, Stig, Legg, Shane, Mnih, Volodymyr,
Kavukcuoglu, Koray 및 Silver, David. 심층 강화 학습을 위한 대규
모 병렬 방법. ICML에서
딥러닝 워크샵. 2015.

Peng, Jing 및 Williams, Ronald J. Incremental 다단계
큐-러닝. 기계 학습, 22(1-3):283–290, 1996.

오른쪽, 벤자민, 레, 크리스토퍼, 라이트, 스티븐,
니우, 펑. Hogwild: 확률적 경사 하강법을 병렬화하기 위한 잠금 없는
접근 방식. 신경의 발전
정보 처리 시스템, pp. 693–701, 2011.

리드밀러, 마틴. 신경 적합 q 반복 - 데이터 효율적인 신경 강화 학습을 통한
첫 번째 경험
방법. 기계 학습: ECML 2005, pp. 317–
328. Springer Berlin Heidelberg, 2005.

Rummery, Gavin A 및 Niranjan, Mahesan. 연결주의 시스템을 사용
한 온라인 q 학습. 1994.

Schoul, Tom, Quan, John, Antonoglou, Ioannis 및 Silver ver,
David. 우선 순위 경험 재생. arXiv 사전 인쇄
arXiv:1511.05952, 2015.

슐만, 존, 레빈, 세르게이, 모리츠, 필립, 요르단,
Michael I, Abbeel, Pieter. 신뢰 지역 정책 최적화. 기계에 대한 국제
회의에서
학습(ICML), 2015a.

슐만, 존, 모리츠, 필립, 레빈, 세르게이, 요르단,
마이클, 그리고 Abbeel, Pieter. 일반화된 이점 추정을 사용한 고차원
연속 제어.
arXiv 사전 인쇄 arXiv:1506.02438, 2015b.

Sutton, R. 및 Barto, A. 강화 학습: In
소개. MIT 프레스, 1998.

Tieleman, Tijmen 및 Hinton, Geoffrey. 강의 6.5-
rmsprop: 기울기를 실행 평균으로 나눕니다.
최근 규모. COURSERA: 신경망
머신 러닝, 4, 2012.

Todorov, E. MuJoCo: 접촉이 있는 다중 관절 역학의 모델링, 시뮬레이션
및 시각화(ed 1.0).
로보틱스 퍼블리싱, 2015.

토마시니, 마르코. 병렬 및 분산 진화 알고리즘: 검토. 기술 보고서, 1999.

Tsitsiklis, John N. 비동기 확률적 근사 및 q-학습. 기계 학습,
16(3):185–202, 1994.

Van Hasselt, Hado, Guez, Arthur 및 Silver, David. 이중 q 학습을
통한 심층 강화 학습. arXiv 사전 인쇄 arXiv:1509.06461, 2015.

van Seijen, H., Rupam Mahmood, A., Pilarski, PM, Machado,
MC 및 Sutton, RS True Online Temporal-Difference Learning.
ArXiv 전자 인쇄, 2015년 12월.

Wang, Z., de Freitas, N. 및 Lanctot, M. 심층 강화 학습을 위한 결투
네트워크 아키텍처. ArXiv 전자 인쇄, 2015년 11월.

왓킨스, 크리스토퍼 존 코니시 헬라비. 지연된 보상에서 배우기. 1989년 영
국 케임브리지 대학교 박사 학위 논문.

Williams, RJ 연결주의 강화 학습을 위한 간단한 통계적 기울기 추종 알
고리즘. 기계 학습, 8(3):229–256, 1992.

Williams, Ronald J 및 Peng, Jing. 연결주의 강화 학습 알고리즘을 사
용한 함수 최적화.
연결 과학, 3(3):241–268, 1991.

Wymann, B., EspiÃl', E., Guionneau, C., Dimitrakakis, C.,
Coulom, R. 및 Sumner, A. Torcs: 개방형 레이싱 카 시뮬레이
터, v1.3.5, 2013년.

"Deep용 비동기식 방법"에 대한 보충 자료 강화학습"

2016년 6월 17일

7. 최적화 세부 사항

우리는 비동기 프레임워크를 사용하여 확률적 경사 하강법과 RMSProp의 두 가지 최적화 알고리즘을 조사했습니다. 이러한 알고리즘의 구현은 많은 수의 스레드를 사용할 때 처리량을 최대화하기 위해 잠금을 사용하지 않습니다.

Momentum SGD: 비동기 설정에서 SGD를 구현하는 것은 비교적 간단하고 잘 연구 되었습니다(Recht et al., 2011). θ를 모든 스레드에서 공유되는 매개변수 벡터라고 하고 Δθi 를 스레드 번호 i로 계산된 매개변수 θ에 대한 손실의 누적 기울기라고 합니다. 각 스레드 i는 표준 모멘텀 SGD 업데이트 mi = αmi + (1 α)Δθi 다음에 θ ← θ ηmi 를 학습률 η, 모멘텀 α 및 잠금 없이 적용합니다. 이 설정에서 각 스레드는 고유한 기울기 및 운동량 벡터를 유지합니다.

RMSProp: RMSProp (Tieleman & Hinton, 2012) 은 딥 러닝 문헌에서 널리 사용되었지만 비동기 최적화 설정에서는 광범위하게 연구되지 않았습니다. 표준 중심이 아닌 RMSProp 업데이트는 다음과 같이 제공됩니다.

$$g = \alpha g + (1 \quad \alpha)\Delta\theta \ \Delta\theta^{2} \tag{S2}$$

$$\theta \leftarrow \theta \quad \eta \frac{}{\sqrt{g +}}, \tag{S3}$$

여기서 모든 작업은 요소별로 수행됩니다. 비동기 최적화 설정에서 RMSProp을 적용하려면 요소별 제곱 그라디언트 g의 이동 평균이 공유인지 스레드 단위인지 결정해야 합니다. 우리는 두 가지 버전의 알고리즘을 실험했습니다. RM SProp이라고 하는 한 버전에서 각 스레드는 방정식 S2에 표시된 고유한 g를 유지합니다. Shared RMSProp이라고 하는 다른 버전에서는 벡터 g가 스레드 간에 공유되고 잠금 없이 비동기적으로 업데이트됩니다. 스레드 간에 통계를 공유하면 스레드당 매개변수 벡터의 복사본을 하나 더 적게 사용하여 메모리 요구 사항도 줄어듭니다.

우리는 서로 다른 학습률과 무작위 네트워크 초기화에 대한 민감도 측면에서 이 세 가지 비동기 최적화 알고리즘을 비교했습니다. 그림 S5는 4가지 다른 게임(Breakout, Beamrider, Seaquest 및 Space Invaders)에서 두 가지 다른 강화 학습 방법(Async n-step Q 및 Async Advantage Actor-Critic)에 대한 방법을 비교한 것입니다. 각 곡선은 50개의 서로 다른 무작위 학습률 및 초기화에 해당하는 50개의 실험에 대한 점수를 보여줍니다. x축은 최종 평균 점수를 기준으로 내림차순으로 정렬한 후 모델의 순위를 나타내고 y축은 해당 모델이 달성한 최종 평균 점수를 나타냅니다. 이 표현에서 더 잘 수행되는 알고리즘은 y축에서 더 높은 최대 보상을 달성하고 가장 강력한 알고리즘은 수평에 가장 가까운 기울기를 가지므로 곡선 아래 영역을 최대화합니다. 공유 통계를 사용하는 RMSProp은 스레드당 통계를 사용하는 RMSProp보다 더 강력한 경향이 있으며, 결과적으로 Momentum SGD보다 더 강력합니다.

8. 실험 설정

Atari 게임의 하위 집합(그림 1, 3, 4 및 표 2)과 TORCS 실험(그림 S6) 에서 수행된 실험 은 다음 설정을 사용했습니다. 각 실험은 GPU 없이 단일 머신에서 실행되는 16개의 배우-학습자 스레드를 사용했습니다. 모든 메서드는 5번의 작업 (t_{max} = 5 및 I_{Update} = 5)마다 업데이트를 수행했으며 공유 RMSProp이 최적화에 사용되었습니다. 세 가지 비동기 값 기반 방법은 40000 프레임마다 업데이트되는 공유 대상 네트워크를 사용했습니다. Atari 실험은 (Mnih et al., 2015) 와 동일한 입력 전처리와 4의 동작 반복을 사용했습니다. 에이전트는 (Mnih et al., 2013)의 네트워크 아키텍처를 사용했습니다. 네트워크는 보폭이 4인 크기 8 × 8 필터 16개가 있는 컨볼루션 레이어를 사용하고, 보폭 2를 포함하는 크기 4 × 4 필터 32개를 포함하는 컨볼루션 레이어를 사용하고, 256개의 은닉 유닛을 포함하는 완전 연결 레이어를 사용했습니다. 3개의 은닉층 모두 정류기 비선형성이 뒤따랐다. 가치 기반 방법은 행동 가치를 나타내는 각 행동에 대해 단일 선형 출력 단위를 가졌습니다. 배우 비평가 에이전트가 사용하는 모델에는 두 가지 출력 세트가 있습니다. 하나는 행동당 하나의 항목이 있는 소프트맥스 출력은 행동 선택 확률을 나타내고, 하나는 선형 출력이 가치 함수를 나타냅니다. 모든 실험은 γ = 0.99의 할인과 α = 0.99의 RMSProp 감쇠 계수를 사용했습니다.

가치 기반 방법은 1, 2, 3 확률 0.4, 0.3, 0.3의 세 값을 취하는 분포에서 탐사 비율을 샘플링했습니다 . 1, 2, 3 의 값은 처음 4000만 프레임에 걸쳐 각각 1에서 0.1, 0.01, 0.5로 어닐링되었습니다. Advantage Actor-Critic은 Atari 및 TORCS 실험 모두에 대해 0.01의 엔트로피 정규화를 사용했습니다. Atari 57개 게임 및 TORCS 3개 레벨에 대해 각각 다른 무작위 초기화 및 초기 학습률을 사용하여 50개의 실험 세트를 수행했습니다. 초기 학습률은 LogUniform(10^{-4}, 10^{-2}) 분포에서 샘플링되었으며 훈련 과정에서 0으로 어닐링되었습니다.

이전 작업(표 1 및 S3) 과 비교하여 표준 평가 프로토콜을 따르고 고정 하이퍼파라미터를 사용했습니다.

9. MuJoCo 물리 시뮬레이터를 사용한 연속 동작 제어

Mujoco 작업에 비동기 이점 행위자 비평 알고리즘을 적용하려면 필요한 설정이 개별 작업 영역에서 사용되는 것과 거의 동일하므로 여기서는 연속 작업 영역에 필요한 차이점만 나열합니다. 많은 작업(즉, 물리학 모델 및 작업 목표)에 대한 필수 요소는 (Lillicrap et al., 2015) 에서 조사한 작업과 거의 동일합니다 . 그러나 연락처 모델을 변경한 Mujoco 개발자의 변경으로 인해 대부분의 작업에 대해 보상과 성능이 비교할 수 없습니다.

모든 영역에 대해 물리적 상태를 입력으로 사용하여 작업을 학습하려고 시도했습니다. 물리적 상태는 관절 위치와 속도, 작업에 목표가 필요한 경우 목표 위치로 구성됩니다. 또한 세 가지 작업(pendulum, pointmass2D 및 그리퍼)에 대해 RGB 픽셀 입력에서 직접 훈련을 조사했습니다. 저차원 물리적 상태의 경우 입력은 200 ReLU 단위가 있는 하나의 은닉 레이어를 사용하여 은닉 상태에 매핑됩니다. 픽셀을 사용한 경우 입력은 비선형성이나 풀링 없이 공간 컨볼루션의 두 레이어를 통과했습니다. 두 경우 모두 인코더 계층의 출력은 128개의 LSTM 셀로 구성된 단일 계층에 공급되었습니다. 아키텍처에서 가장 중요한 차이점은 정책 네트워크의 출력 계층에 있습니다. 동작 출력이 Softmax인 이산 동작 영역과 달리 여기에서 정책 네트워크의 두 출력은 구형 공분산을 갖는 다차원 정규 분포의 평균 벡터 μ와 스칼라 분산 σ로 취급하는 두 개의 실수 벡터입니다. 작동하기 위해 입력은 모델을 통해 출력 계층으로 전달되며 여기에서 SoftPlus 연산 log(1 + exp(x))에 의해 결정된 정규 분포에서 샘플링합니다. 선형 레이어의 출력. 지속적인 제어 문제에 대한 실험에서 정책 네트워크 및 가치 네트워크에 대한 네트워크는 매개변수를 공유하지 않지만 이 세부사항이 중요하지는 않습니다. 마지막으로, 에피소드는 일반적으로 최대 수백 시간 단계이기 때문에[2] 정책 또는 가치 기능 업데이트에서 부트스트랩을 사용하지 않고 각 에피소드를 단일 업데이트로 일괄 처리했습니다.

[2]. 실제로 μ는 선형 레이어로 모델링되고 σ^2

이산 행동의 경우와 같이 탐색을 장려하는 엔트로피 비용을 포함했습니다. 지속적인

(log(2πσ2) + 1) 의 출력으로 정의된 정규 분포의 미분 엔트로피에 대한 비용을 사용한 경우 모든 작업 행위자 네트워크에서 이에 대한 솔루션을 찾습니다. 그림 S8은 서로 다른 시간에서 발견된 솔루션을 보여주며 정책 평가에서 다양한 이점 및 모든 영역 될 수 있음을 보여줍니다. 픽셀 기반 관찰에서 수행된 실험을 포함하여 모든 실험은 CPU에서 실행되었습니다. 픽셀 입력에서 직접 도메인을 해결하는 경우에도 24시간 이내에 솔루션을 안정적으로 찾을 수 있음을 발견했습니다. 그림 S7 은 샘플링된 학습률에 대한 최고 점수의 산점도를 보여줍니다. 대부분의 영역에는 작업에서 지속적으로 우수한 성과를 달성하는 광범위한 학습률이 있습니다.

알고리즘 S2 비동기식 n-단계 Q-학습 - 각 액터-학습기 스레드에 대한 의사 코드.

// 전역 공유 매개변수 벡터 θ를 가정합니다.
// 전역 공유 대상 매개변수 벡터 θ를 가정합니다.
// 전역 공유 카운터 T = 0이라고 가정합니다.
스레드 스텝 카운터 초기화 t ← 1 대상 네
트워크 매개변수 초기화 θ ← θ 스레드별 매개변수
초기화 θ = θ 네트워크 기울기 초기화 dθ ← 0 repeat 기울
기 지우기 dθ ← 0 스레드별 매개변수 동기화 θ tstart
= t 상태 가져오기 st repeat 조치 취하기 at Q(st, a;
θ)에 기반한 -greedy 정책에 따라 보상 rt 와 새
로운 상태를 받습니다. st+1 t ← t + 1 T ← T + 1 = 나는
터미널 st 까지 또는 t tstart == tmax 터미널
st maxa 에 대해 0 Q(st, a; θ) i ∈ {t
1, ... , tstart} 에 대한 비단말 st 에 대해 R ← ri
+ γR 수행)

R =

dθ를 사용하여 θ의 비동기식 업데 0)) : dθ ← dθ + ∂ (R Q(si,ai;θ 2 ∂θ
이트를 수행 하기 위해 기울기 wrt θ
end를 누적합니다. if T mod Itarget == 0 then θ
← θ T > Tmax 가 될 때까지 종료

알고리즘 S3 비동기 이점 Actor-Critic - 각 Actor-Learner 스레드에 대한 의사 코드.

// 전역 공유 매개변수 벡터 θ 및 θv 및 전역 공유 카운터 T = 0 가정 // 스레드별 매개변수 벡터 가정 θ 스레드 스
텝 카운터 초기화 t ← 1 반복 그라디언트 재설정: dθ ← 0 및 dθv ← 0.

스레드별 매개변수 동기화 θ tstart = t 상태 가져오기 = θ 및 θ = θv
st repeat 정책 에 따라 수행 π(at|st; θ 보상 rt 및 새
상태 수신 st +1 t ← t + 1 T ← T + 1 터미널 st 또는
t tstart == tmax 0 for terminal st for non-
terminal st// Bootstrap from last state)

R =
V (st, θ0 v)
for i ∈ {t 1, . . . , tstart} do R ←
ri + γR 누적 기울기 wrt θ
: dθ ← dθ + ∇θ0 log π(ai|si; θ)(R V (si; θ 0 안에))
dθ를 사용하여 θ의 비동기 업데이트 dθv ← dθv + ∂ (R V (si; θ) v))2 / ∂ θ0 v v:
수행 및 dθv 를 사용하여 θv의 비동기 업데
이트 수행 을 위해 θ end wrt 그라디언트를 누적합니다 . T > Tmax 까지

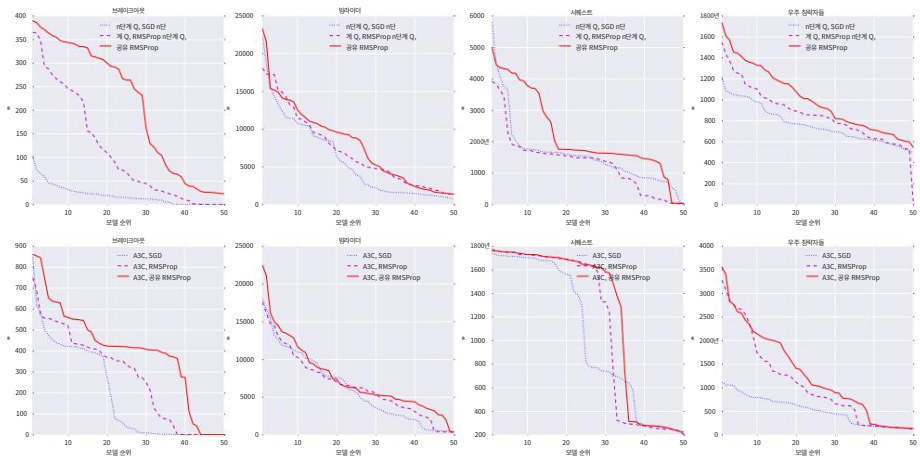

그림 S5. 4개의 다른 Atari 게임(Break out, Beamrider, Seaquest 및 Space Invaders)에서 2개의 다른 알고리즘(Async n-step Q 및 Async Advantage Actor-Critic)을 사용하여 테스트한 3가지 최적화 방법(Momentum SGD, RMSProp, Shared RMSProp)의 비교. 각 곡선은 50개 이상의 무작위 초기화 및 학습률에 대한 검색을 포함하는 내림차순으로 정렬된 50개의 실험에 대한 최종 점수를 보여줍니다. 상단 행은 Async n-step Q 알고리즘을 사용한 결과를 보여주고 하단 행은 Async Advantage Actor-Critic을 사용한 결과를 보여줍니다. 각 개별 그래프는 4가지 게임 중 하나와 3가지 다른 최적화 방법에 대한 결과를 보여줍니다. Shared RMSProp은 공유 없는 Momentum SGD 및 RMSProp보다 다양한 학습률 및 임의 초기화에 대해 더 강력한 경향이 있습니다.

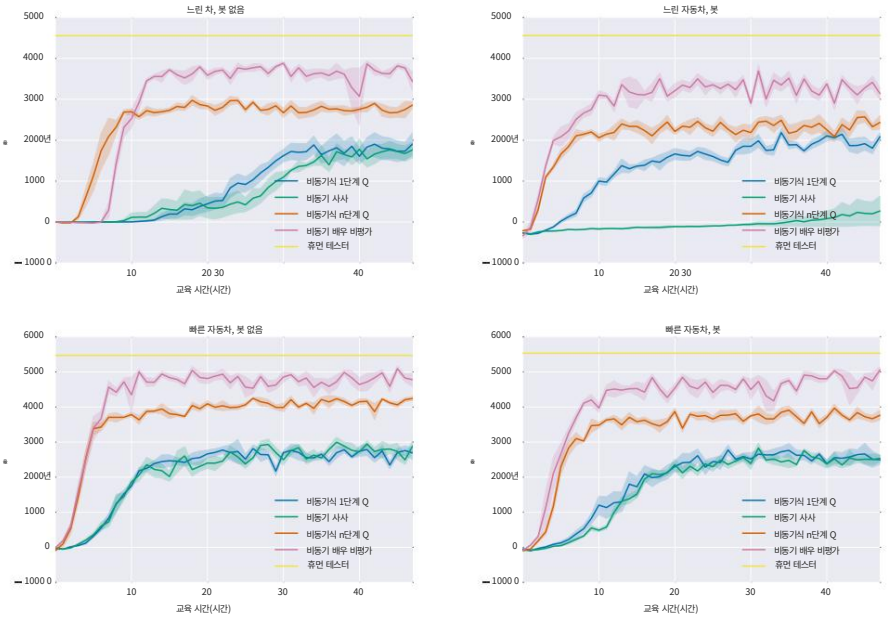

그림 S6. TORCS 자동차 경주 시뮬레이터의 알고리즘 비교. 자동차 속도와 상대방의 유무에 대한 네 가지 다른 구성이 표시됩니다. 각 플롯에서 4가지 알고리즘(1단계 Q, 1단계 Sarsa, n단계 Q 및 Advantage Actor-Critic)은 벽시계 시간의 점수 대 훈련 시간에서 비교됩니다. 다단계 알고리즘은 4개 수준 모두에서 1단계 알고리즘보다 훨씬 빠르게 더 나은 정책을 달성합니다. 곡선은 LogUniform(10 4 , 10 2) 에서 샘플링된 학습률 과 고정된 다른 모든 하이퍼파라미터를 사용하여 50개의 실험에서 5개의 최상의 실행에 대한 평균을 보여줍니다.

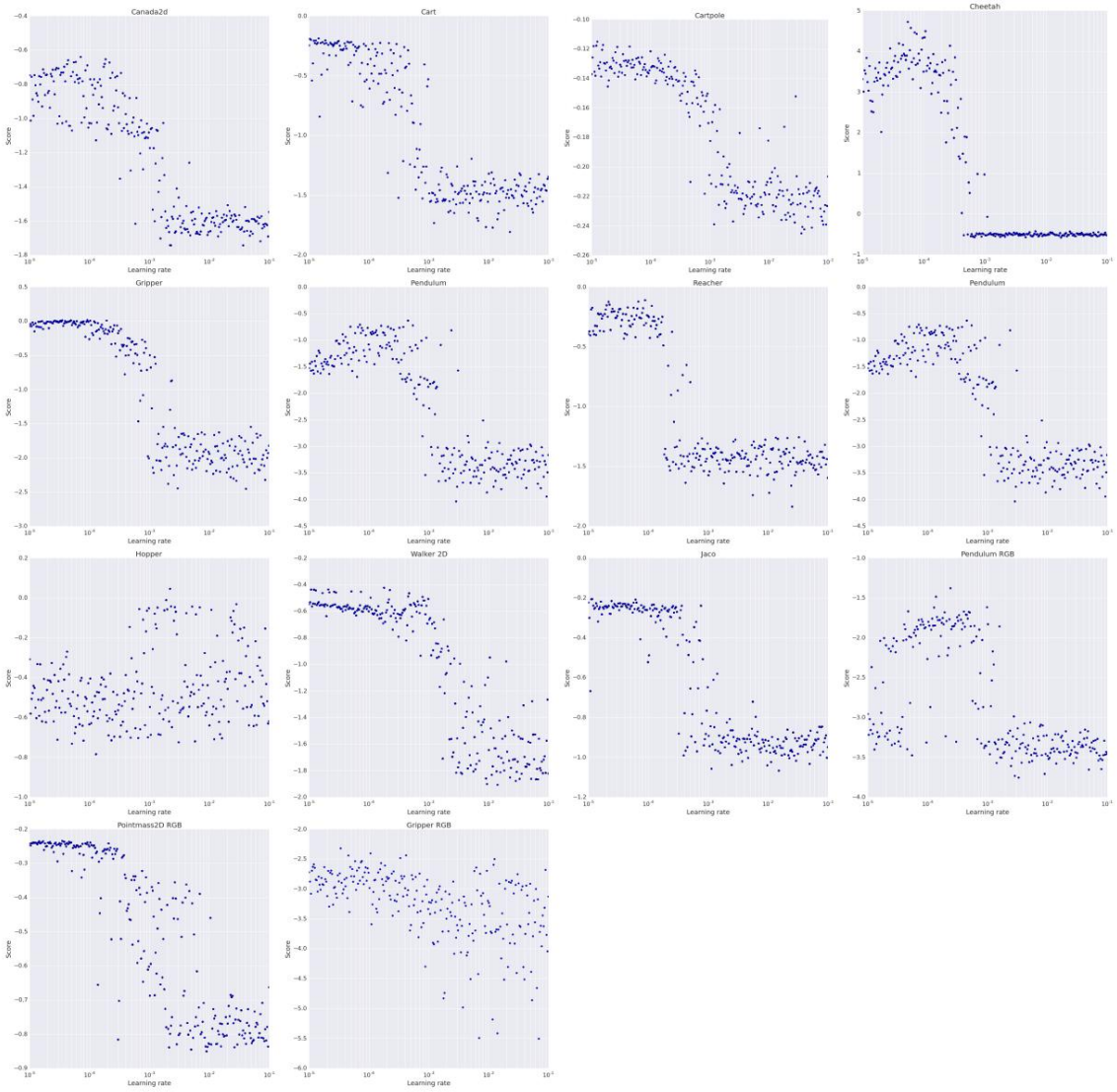

그림 S7. Mujoco 연속 작업 영역에 대한 성능. LogUniform(10 5 , 10 1) 에서 샘플링된 학습률에 대해 얻은 최고 점수의 산점도 입니다. 거의 모든 작업에 대해 작업에 대한 우수한 성능으로 이어지는 광범위한 학습률이 있습니다.

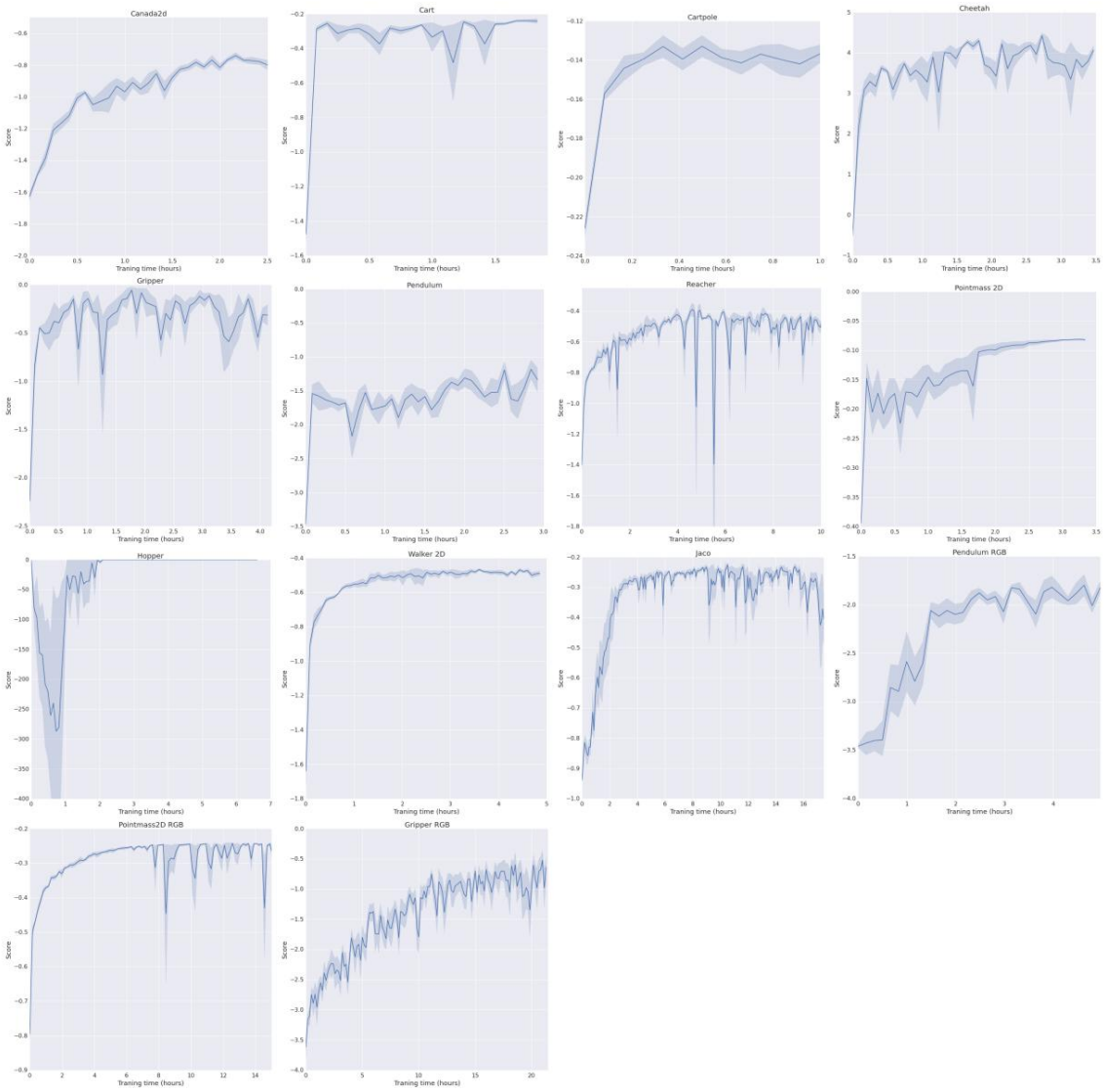

그림 S8. Mujoco 도메인에 대한 에피소드당 점수 대 벽시계 시간 플롯. 각 플롯은 상위 5개 실험에 대한 오차 막대를 보여줍니다.

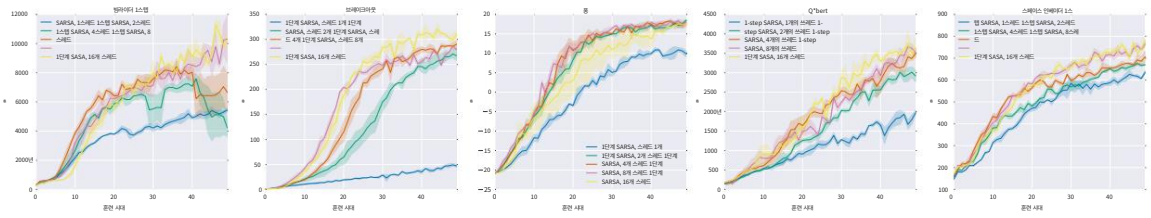

그림 S9. 5개의 Atari 게임에서 다양한 수의 배우-학습자 1단계 Sarsa의 데이터 효율성 비교. x축은 에포크가 400만 프레임(모든 스레드에서)에 해당하는 훈련 에포크의 총 수를 보여줍니다.
y축은 평균 점수를 보여줍니다. 각 곡선은 50개 이상의 무작위 학습률에서 검색한 최고 성능 에이전트 3명의 평균을 보여줍니다. Sarsa는 병렬 작업자 수가 증가하여 데이터 효율성이 향상되었음을 보여줍니다.

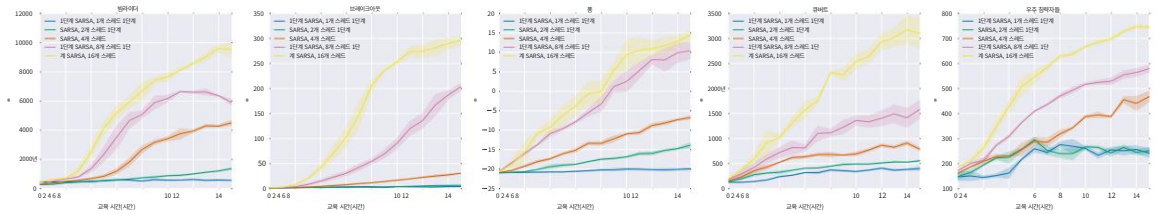

그림 S10. 5개의 Atari 게임에서 모든 1단계 Sarsa에 대한 다양한 수의 배우-학습자의 교육 속도 비교.
x축은 훈련 시간을 시간 단위로 표시하고 y축은 평균 점수를 표시합니다. 각 곡선은 50개 이상의 무작위 학습률에서 검색한 최고 성능 에이전트 3명의 평균을 보여줍니다. Sarsa는 더 많은 병렬 액터 학습자를 사용하여 상당한 속도 향상을 보여줍니다.

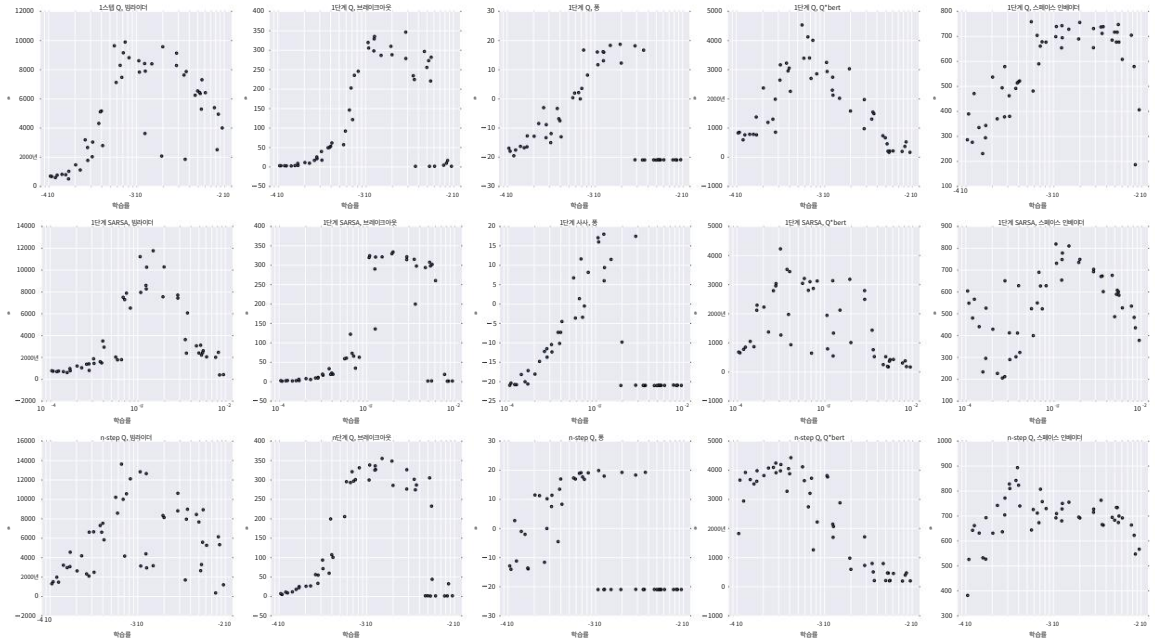

그림 S11. 50가지 다른 학습률과 무작위 초기화에 대해 5가지 게임(Beamrider, Breakout, Pong, Q*bert, Space Invaders)에서 1단계 Q, 1단계 Sarsa 및 n단계 Q로 얻은 점수의 산점도. 모든 알고리즘은 학습률 선택에 대해 어느 정도 견고성을 나타냅니다.

게임	DQN 고릴라 570.2 813.5	더블	결투 우선 A3C FF, 1일 A3C FF A3C LSTM				
외계인	133.4 189.2 333211195.8	1033.4	1486.5	900.5 182.1 518.4 945.3			
아미다르	124.5 3324.7 697913.6	169.1	172.7	218.4 283.9 263.9 173.0			
폭행		6060.8	3994.8	7748.5 3746.1 5474.9 14497.9			
아스테릭스		16837.0	15840.0	31907.5 6723.0 22140.5 17244.5			
소행성		1193.2	2035.4	1654.0 3009.4 4474.5 5093.1			
아틀란티스	76108.0 629166.5 319688.0	445360.0 1129.3		593642.0 772392.0 911091.0 875822.0			
은행 강도	176.3 399.4	886.0		816.8 946.0 970.1 932.8			
배틀존	17560.0 19938.0	24740.0	31320.0	29100.0	11340.0 12950.0 20760.0		
빔 라이더	8672.4 3822.1	17417.2	14591.3	26172.7	13235.9	22707.9	24622.2
버저크		1011.1	910.6	1165.6	1433.4	817.9	862.2
볼링 권투	41.2 54.0	69.6	65.7	65.8	36.2	35.1	41.8
브레이크아	25.8 74.2	73.5	77.3	68.6	33.7	59.8	37.3
웃 지네 쵸퍼	303.9 313.0	368.9	411.6	371.6	551.6	681.9	766.8
지휘관 미친 등	3773.1 6296.9	3853.5	4881.0	3421.9	3306.5	3755.8	1997.0
반가 수비수 악마 공격 더	3046.0 3191.8	3495.0	3784.0	6604.0	4669.0	7021.0	10150.0
블 덩크 엔듀로 낚시 더비	50992.0 65451.0	113782.0	124566.0	131086.0	101624.0	112646.0	138518.0
고속도로 동상 고퍼 중력		27510.0	33996.0	21093.5	36242.5	56533.0	233021.5
영웅	12835.2 14880.1	69803.4	56322.8	73185.8	84997.5	113308.4	115201.9
	-21.6 -11.3	-0.3	-0.8	2.7	0.1	-0.1	0.1
	475.6 71.0	1216.6	2077.4	1884.4	-82.2	-82.5	-82.5
	-2.3 4.6	3.2	-4.1	9.2	13.6	18.8	22.6
	25.8 10.2	28.8	0.2	27.9	0.1	0.1	0.1
	157.4 426.6	1448.1	2332.4	2930.2	180.1	190.5	197.6
	2731.8 4373.0	15253.0	20051.4	57783.8	8442.8	10022.8	17106.8
	216.5 538.4	200.5	297.0	218.0	269.5	303.5	320.0
	12952.5 8963.4	14892.5	15207.9	20506.4	28765.8	32464.1	28889.5
아이스하키 제임	-3.8 -1.7	-2.5	-1.3	-1.0	-4.7	-2.8	-1.7
스 본드 캥거루 크	348.5 444.0	573.0	835.5	3511 .	351.5	541.0	613.0
룰 쿵푸 마스터	2696.0 1431.0	11204.0	10334.0		106.0	94.0	125.0
몬테주마의 복수	3864.0 6363.1	6796.1	8051.6		8066.6	5560.0	5911.4
Ms. Pacman Name	118755.0 20620.0	30206.1	24288.0		3046.0	28819.0	40835.0
This Game Phoenix Pit Fall	50.0 84.0		22.0		53.0	67.0	41.0
Pong Private Eye Q*Bert	5463. 9263.0		2250.6		594.4	653.7	850.7
River Raid			11185.1		5614.0		12093.7
			20410.5				74786.7
			-46.9				-135.7
	16.2 16.7		18.8				10.7
	298.2 2598.6		292.6				421.1
	4589.8 7089.8		14175.8				21307.5
	4065.3 5310.3	302066.2121266	4121266.4121266.19366.19366		.2818181818181818	2818181818181.	6591.9
로드 러너	9264.0 43079.8	43156.0	58549.0	56990.0	31769.0	34216.0	73949.0
로봇 탱크	58.5 61.8	59.1	62.0	55.4	2.3	32.8	2.6
시퀘스트	2793.9 10145.9	14498.0	37361.6	39096.7	2300.2	2355.4	1326.1
스키 타기		-11490.4	-11928.0	-10852.8	-13700.0	-10911.1	-14863.8
솔라리스		810.0 2628.7	5997.3	2238.2	1884.8	1956.0	1936.4
우주 침략자들	1449.7 1183.3	58365.0 90804.0	0.7	9063.0	2214.7 64393.0	15680.0	23846.0
스타 거너	34081.0 14919.2	4.4 8267.8 6608.0	18651.0	51959.0	-9.6 -9.7 -10.2	-6.3	164766.0
둘러 싸다		92.2 48.0	8747.7	-0.9	26.1 156.3	54525.4	-8.3
테니스	-2.3	19086.9 24759.2	23.1	-2.0	74705.7 19.0	23.0	-6.4
타임 파일럿	5640.0	110976.2 10431.0	7448.0	185852.6	3316	18.0	27202.0
투탕캄	32.4			33.6	172444.0	7270.8	144.2
위아래로	3311.3			29443.7	247270.8	24270.8	105728.7
투기	54.0			244.0			25.0
비디오 핀볼	20228.1			374886.9			470310.5
워의 마법사	246.0			7451.0			18082.0
야르스 리벤지				5965.10			5615.5
작슨	831.0 6159.4			9			23519.0

표 S3. 인간 시작 조건에 대한 원시 점수(에뮬레이터 시간 30분). (Nair et al., 2015). (Van Hasselt et al., 2015) 에서 가져온 Double DQN 점수 , (Wang et al., 2015)에서 Dueling 점수 및 Prioritized (Schaul et al., 2015) 에서 가져온 점수