1. Introduction

인간은 제한된 감각으로 지각할 수 있는 것에 기초하여 세계에 대한 정신적 모델을 개발한다.

아무도 세계, 정부, 나라를 상상하지 않는다.

개념과 그 사이의 관계만을 선택하고, 그것들을 사용하여 실제 시스템을 나타낸다.

- 시스템 역학의 아버지 Jay Wright Forrester -

일상 생활을 통해 흐르는 방대한 양의 정보를 처리하기 위해 우리의 뇌는 이 정보의 공간적 측면과 시간적 측명 모두에 대한 추상적 표현을 학습한다.

우리는 장면을 관찰하고 그에 대한 추상적 설명을 기억할 수 있다.

(증거) 우리가 주어진 순간에 지각하는 것은, 미래에 대한 뇌의 예측에 의해 좌우되어서.

일반적으로 미래를 예측 하는게 아니라, 주어진 미래 감각 데이터를 예측하는 것 이다.

예측 모델에 따라 과정을 계획 할 필요없이, 본능적 행동 가능

야구 예)

야구 배트를 휘두르는 방법을 결정하는게, 시각적 신호가 뇌에 도달하는 시간보다 짧다.

시속 100마일의 빠른 공을 칠 수 있는 이유는, 볼이 언제 어디로 날아갈지 본능적으로 예측하는 능력 때문이다.

Model-free RL 방법은 종종 매개변수가 거의 없는 작은 신경만 만 사용한다.

RL 알고리즘은 크레딧 할당 문제로 인해 병목현상이 발생함.

그래서 기존 RL 알고리즘들은 대규모 모델의 수백만 개의 가중치를 학습하기 어려움.

Large RNN → back propagation 으로 학습 한다.

이 작업에서 우리는 agent를 큰 세계 모델과, 작은 컨트롤러 모델로 나누어 RL 작업을 처리하기 위해 큰 신경망1을 훈련하는 방법을 살펴볼 것 이다?

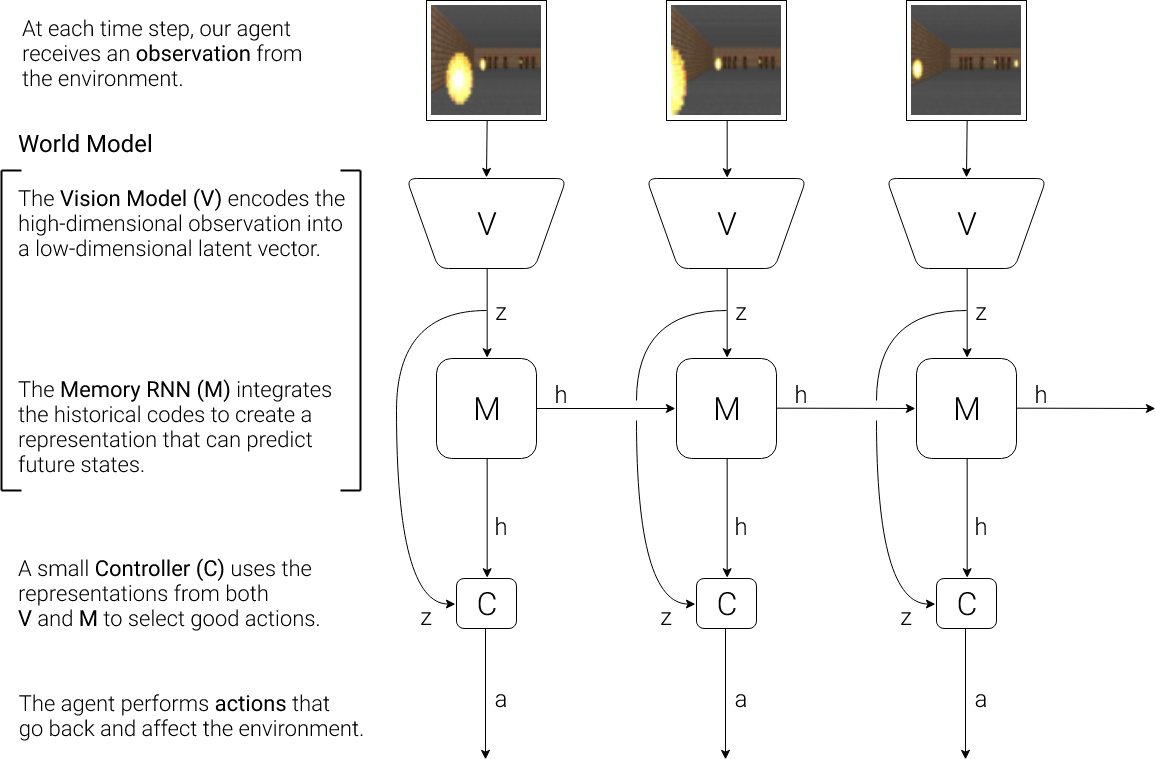
우리는 먼저 큰 신경망을 훈련하어 비지도 방식으로 에이전트 세계의 모델을 학습한 다음, 이 세계 모델을 사용하여 작업을 수행하는 방법을 배우기 위해 더 작은 컨틀롤러 모델을 훈련한다.

Small controller 사용하면, 학습 알고리즘이 lager world model 을 통해 용량과 표현력을 희생하지 않으면서, 작은 검색 공간에서 credit 할당 문제에 집중할 수 있다.

과거 정보를 기반으로 미래코드?를 예측하는 메모리 구성 요소가 있다.

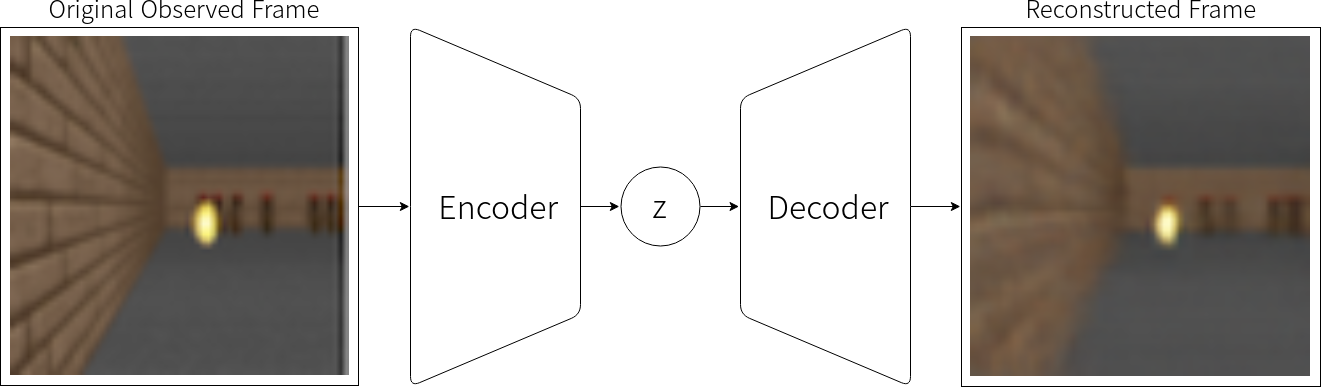
에이전트는 비전 및 메모리 구성요소에 의해 생성된 표현에만 기반하여 취해야 할 조치를 결정하는 의사 결정 구성 요소를 가진다.

2. Agent Model

Vision V, Memory M, Controller C

2.1 VAE (V) Model

보통 Video에서 고차원의 2D image frame 이 input 된다. V는 입력 frame을 추상적이고 압축된 표현을 하는것을 학습한다.

Simple Variational Autoencoder (VAE)

V model이 각 image frame을 small latent vector z로 변환해준다.

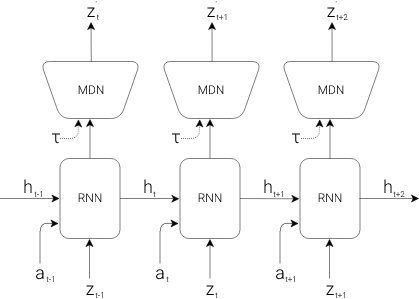
시간 지남에 따른걸 보고 싶다.

**2.2 MDN-RNN (M) Model**

M 모델은 미래를 예측하는게 목표다.

M모델은 V모델이 만들것으로 예상되는 future z vectors의 예측 모델 역할을 한다.

많은 복잡한 환경이 확률론적 성격을 띠기 때문에 우리는 z의 결정론적 예측 대신 확률 밀도 함수 p(z)를 출력하도록 RNN을 훈련시킨다.



Mixture Density Network 출력 레이어가 있는 RNN

MDN은 다음 잠재 벡터의 예측을 샘플링 하는데 사용되는 가우스 분포의 혼합 매개변수를 출력.

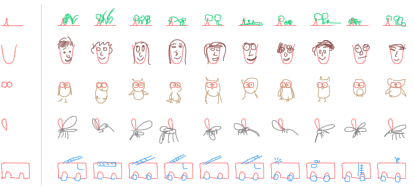
가우스 분포의혼합으로 p(z)를 근사하고, RNN을 훈련하여 사용 가능한 현재 및 과거 정보가 주어지면 다음 잠재 백터 z(t+1)의 확률 분포를 출력한다.

a(t) == action

h(t) == hidden state



이 방법은 과거에는 손글씨 생성, 스케치와 같은 sequence 생성 문제에 사용됐었음.



**2.3 Controller (C) Model**

C모델은 rollout 동안 에이전트의 예상 누적 보상을 최대화 하기 위해 취해야 할 조치 과정을 결정하는 역할을 함.

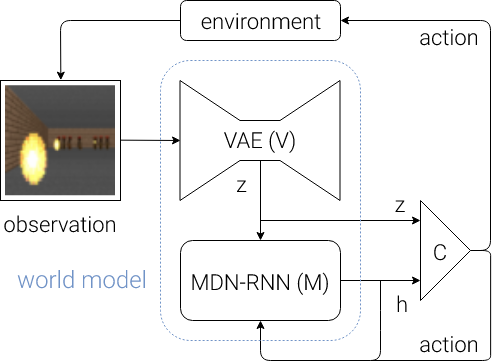
의도적으로 C를 가능한 한 간단하고 작게 만들고, V와 M과는 별도로 훈련하여

에이전트의 복잡성의 대부분이 세계모델(V, M)에 있도록 합니다.

C는 각 time step마다 z(t)와 h(t)를 action a(t)에 직접 매핑하는 간단한 단일 선형 layer 모델이다.

.

**2.4 Putting V,M and C Together]**



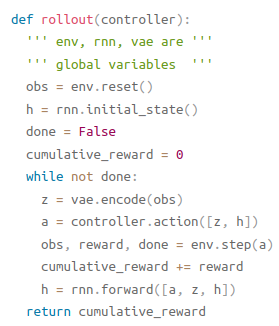
각 time step 마다 z(t)를 생성하기 위해 V에 의해 먼저 처리된다.

C에 대한 input은 각 time step 에서 M의 hidden state 와 연결된 잠재 벡터 이다.

그 다음 C는 motor control에 의한 action vector인 a(t)를 출력하고 환경에 영향을 미친다.

그 다음 M은 현재 z(t) 및 a(t)를 입력으로 사용하여, time t+1 에서 사용할 h(t+1)을 생성하도록 자신의 은닉 상태를 업데이트 한다.

슈도 코드



Open AI Gym 환경

텍스트이(가) 표시된 사진

자동 생성된 설명

주어진 controller C 에서 이 함수를 실행하면 rollout 동안 누적 보상(Cumulative\_reward)이 반환 됨

C의 매개변수를 최적화 하기 위해 Covariance-Matrix Adaptation Evolution Strategy(CMA-ES)를 최적화 알고리즘으로 선택

== black box optimization

3. Car Racing Experiment

트랙은 trial 마다 랜덤 생성 되고, 최소 시간에 최대한 많은 타일 방문하는 것에 대해 reward 를 받는다.

에이전트는 세가지 연속동작(좌/우, 스티어링, 가속 및 브레이크)을 제어한다.

V모델을 훈련하기 위해 먼저 환경의 10000개의 무작위 롤아웃(최초 구축 후 확산) 데이터 세트를 수집

먼저 무작위로 행동하여 환경을 여러 번 탐색하고 수행된 무작위 행동(a\_t)과 환경에서 결과적인 관찰을 기록.

이 데이터 세트를 활용하여 각 프레임의 잠재 공간을 학습하기 위해 V를 훈련한다.

주어진 프레임과 z로부터 디코더에 의해 생성된 프레임의 재구성된 버전가느이 차이를 최소화 하여 각 프레임을 저차원 잠재 벡터로 인코딩 하도록 VAE를 훈련한다.

이제 훈련된 V모델을 사용하여 M모델을 훈련하는 시간에 각 프레임을 사전 처리할 수 있다.

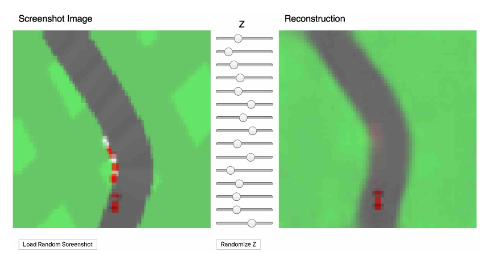
이 사전 처리된 데이터와 기록된 무작위 행동을 사용하여 MDN-RNN은 이제  를 가우시안 혼합으로 모델링하도록 훈련할 수 있다.

이 실험에서 world model(V & M)은 환경의 실제 보상 신호에 대한 지식이 없습니다.

이 모델은 관찰된 이미지 프레임의 시퀀스를 압축하고 예측하는 것이다.

Controller 모델만이 환경의 보상 정보에 접근할 수 있다.

선형제어 모델 내부에 매개변수가 867개에 불과하기 때문에 CMA-ES와 같은 진화 알고리즘이 이 최적화 작업에 적합하다.



VAE 를 사용하여 z\_t프레임을 재구성하여, 롤 아웃 중에 에이전트가 실제로 보는 정보의 품질을 시각화 할 수 있다.

3.2 Procedure

Car Racing

1. 무작위 정책에서 10000개의 롤아웃을 수집

2. 프레임을  로 인코딩 하도록 VAE(V)를 훈련

3. MDN-RNN(M)을 훈련 

4. 컨트롤러 C 정의 

5. CMA-ES로 누적보상을 최대화 하도록 기대하는 W\_c 와 b\_c를 푼다.

3.3 Experiment Results

V Model Only

관찰을 잘 나타내면 에이전트가 운전하도록 훈련시키는건 어려운 작업이 아니다..

이러한 이유로 C가 V에만 엑세스 할 수 있지만, M에는 엑세스 할 수 없도록 하여 에이전트를 테스트하고 싶으므로,  로 컨트롤러를 정의한다.

날카로운 부분에서 이탈을 자주하는데, 632+- 251로 점수가 나왔다. 100번 랜덤 시도 끝에,

정책 네트워크에 은닉층 추가시 788 +- 141 로 개선되었지만, 문제를 해결하지는 못함

Full World Model (V and M)

V 모델은 한 순간의 표현만을 포착하여 많은 예측력을 가지지 않는다.

대조적으로 M은 한가지 일을 하도록 훈련받았다.

z\_t +1을 예측하는 것,

M의 z\_t+1 예측은 RNN 의 hidden state h\_t at time t 에서 생성 되므로,

이 벡터는 에이전트에 제공할 수 있는 학습된 기능 집합에 대한 좋은 후보이다.

z\_t 와 h\_t와 컨트롤러 C를 결합하면, 현재 관찰과 미래에 무엇을 기대하는지를 잘 표현할 수 있다.

에이전트가 z\_t 와 h\_t양쪽에 엑세스 할 수 있도록 허용하면, 에이전트의 운전 능력이 크게 향상된다.

운전이 더 안정적이고, 에이전트가 날카로운 모서리를 효과적으로 공격할 수 있다.

게다가 운전중 빠른 반사 운전 결정을 내릴때는, 에이전트가 미리 계획하고 미래의 가상 시나리오를 실행할 필요가 없다는걸 알 수 있었다.

미래의 확률 분포에 대한 정보가 포함되어 있으므로, 에이전트를 본능적으로 RNN을 쿼리하여 액션 결정을 안내할 수 있었다.

노련한 Formula one 드라이버나 야구선수 처럼 에이전트는 순간의 열기 속에서 탐색할 때와, 장소를 본능적으로 예측 할 수 있다.

텍스트이(가) 표시된 사진

자동 생성된 설명

기존 deep rl 방법은 입력에 몇 개의 최근 프레임을 쌓는 것 외에도, 에지 감지를 사용하는 것 과 같은 각 프레임의 사전처리 작업이 필요 했었다.

하지만 우리의 world model은 original rgb 픽셀 이미지의 스트림을 취하고, 공간-시간적 표현을 직접 학습한다.

우리가 아는 한, 우리의 방법은 이 작업을 해결하기 위해 보고 된 첫번째 솔루션이었다.

3.4 Car Racing Dreams

world model은 미래를 모델링 할 수 있기 대문에, 가상의 자동차 경주 시나리오를 자체적으로 만들수도 있다.

우리는 현재상태 sample z\_t+1이 주어진 경우, z\_t+1의 확률분포를 생성하고 이 샘플을 실제 관측값으로 사용하도록 할 수 있다.

훈련된 C를 M이 생성한 환각 환경에 다시 넣을 수 있다.

예시 영상)

텍스트이(가) 표시된 사진

자동 생성된 설명

4. VizDoom Experiment

우리는 현실환경 내부에서 학습된 정책이, 환각(꿈) 환경 내부에서 어느정도 기능한다는 것을 보았다.

이것은 질문을 던진다.

에이전트가 자신의 꿈 내부에서 학습하도록 훈련하고 이 정책을 실제 환경으로 다시 이전 할 수 있는지? (알파고 마냥 스스로 바둑 두면서 학습한다는듯?)

viz-doom환경을 모방하도록 훈련된 world model에 의해 생성된 환각 내부의 에이전트를 훈련시켰다.



몬스터가 쏘는 불덩어리를 피해야함. 이 환경에서는 명시적인 보상이 없어서, 자연 선택을 모방하기 위해 누적 보상은 에이전트가 roll-out동안 살아남기 위해 관리하는 시간 단계 수로 정의할 수 있다.

환경의 각 롤아웃은 최대 2100개의 타임 스텝(60초) 동안 실행되며 100개의 연속 roll-out에 대한 평균 생존 시간이 750개의 타임 스텝(20초)보다 크면 작업이 해결 된 것으로 간주함.

4.2 Procedure

몇가지 주요 차이점 제외하면, Car racing 작업과 거의 동일.

car racing에 M은 next z\_t를 모델링 하도록 훈련됨.

여기서는 우리의 에이전트를 훈련할 수 있는 world model을 만들고 싶기 때문에, 여기서 우리의 M모델은 에이전트가 다음 프레임에서 죽을지 여부도 예측함.

이 시뮬레이션에서 환각 과정 동안 실제 픽셀 프레임을 인코딩 하기위해 V모델이 필요 없음.

따라서 에이전트는 잠재적인 공간 환경에서만 완전히 훈련할 것 인데, 이것은 우리가 보게 될 많은 장점이 있다.

이 가상환경은 실제 환경과 동일한 인터페이스를 가지므로, 에이전트가 가상 환경에서 만족스러운 정책을 학습한 후 이 정책을 실제 환경에 쉽게 배포하여 정책이 얼마나 잘 전달 되는지 확인할 수 있다.

1. 랜덤 정책에 의한 10000개의 rollouts을 수집한다

2. encoder로 각 프레임을 latent vector 로 V를 학습 시킨다. 그리고 V를 사용해 수집된 이미지를 latent space representation으로 변환한다.

3. M을 학습시킨다. 

4. C를 정의한다. 

5. CMA-ES를 사용해, 예상 생존 시간을 최대화 하는 W\_c를 를 구한다.

6. 실제환경에서 (5)에서 학습된 정책 사용?

텍스트이(가) 표시된 사진

자동 생성된 설명

4.3 Training Inside of the Dream

약간의 훈련 후에 우리의 controller는 꿈의 환경을 탐색하고, M이 생성한 괴물이 발사하는 치명적인 불덩이에서 탈출하는 방법을 배웠다.

우리 에이전트는 각 스텝별 가상 환경에서 ~900점수 달성

RNN기반 world model은 인간 프로그래머가 디자인한 완전한 게임 환경을 모방하도록 훈련되었다.

무작위 에피소드에서 수집된 원시 이미지 데이터에서만 학습하여, 게임 논리, 적의 행동, 물리학 및 3D그래픽 렌더링과 같은 게임의 필수 측면을 시뮬레이션 하는 방법을 배운다.

M모델은 게임 상태의 내부 표현을 조정하는 방법을 학습(에이전트가 어느 방향으로 너무 멀리가면 이동 차단 하거나, 몬스터가 발사하는 불덩이 추적해서, 의도한 방향으로 이동하거나, 불덩어리에 의해 사망했는지 등)

이때 가상환경에서는 불확실성을 추가할 수 있다. 즉 게임이 더 어려워짐

그래서 실제 환경에서 더 잘 작동함.

4.4 Transfer Policy to Actual Environment

실제 환경에서는 가상환경에서 나온 점수보다 더 좋은 점수를 기록함

V 모델이 각 프레임의 모든 세부사항을 올바르게 가져올순없지만(몬스터의 수 를 정확하게 파악), 에이전트는 여전히 학습된 정책을 사용하여 실제 환경에서 탐색할 수 있다.

4.5 Cheating the world Model

우리는 어린시절 게임 디자이너가 의도하지 않은 방식으로 비디오 게임을 악용한적이 있다. 무한한 생명 등을 발견하여, 악용해서 어려운 게임을 쉽게 완료할 수 있었음.

그 과정에서 디자이너가 의도한 대로 게임을 마스터하는데 필요한 기술을 배울 기회를 상실할 수 있음.(skip 되버려서)

예를 들어, 가상환경에서 몬스터가 불덩어리를 쏘려고 하면, 초능력을 가진 것처럼 불덩어리를 꺼버림

world model은 환경의 대략적인 확률 모델이기 대문에, 실제 환경을 지배하는 법칙을 안따름. 그래서 실제 환경에서는 존재하지 않는 악용이, 가상 환경에서는 존재할 수 있음.

그리고 에이전트를 위한 가상 꿈 환경을 생성하기 위해 M모델을 사용하고 있기 때문에 컨트롤러에 M의 모든 숨겨진 상태에 대한 엑세스 권한도 부여

본질적으로 우리 에이전트에게 플레이어가 보게 되는 게임 관찰뿐만 아니라,

게임 엔진의 모든 내부 상태, 메모리 에대한 엑세스 권한을 부여함

따라서 우리의 에이전트는 예상 누적보상을 최대화 하기 위해, 게임 엔진의 숨겨진 상태를 직접 조작하는 바법을 효율적으로 탐색할수 있음. 즉 우리의 가상 환경에서는 좋아 보이지만, 실제 환경에서는 실패할 정책을 찾게됨.

베이지안 모델을 사용하면 불확실성 추정치로 이 문제를 해결하는데 어느정도 도움이 되지만, 문제를 완전히 해결하지는 못함.

최근 연구는 먼저 학습된 정책으로 정책 네트워크를 초기화함으로써 모델기반 접근 방식을 기존의 모델 free RL 교욱과 결합하지만, 이후에 이 정책을 미세 조정하기 위해 모델-free 방법에 의존해야함. 실제환경에서

In Learning to think에서 RNN M이 항상 신뢰할 수 있는 예측 변수는 아니라는 것을 받아들일 수 있다. A(잠재적으로 진화 기반) RNN C는 원칙적으로 결함 있는 M을 무시하거나, 계층적 계획 등을 포함한 임의의 계산 목적을 위해 M의 특정 유용한 부분을 악용하는 방법을 학습할 수 있다.

우리의 현재 접근 방식은 RNN M을 사용하여 단계별로 미리 예측하고 계획하는 일부 이전 시스템에 여전히 더 가까움. 그러나 이 초기 작업과 달리 단순성과 일반성의 이점을 모두 갖춘 RNN과 결합된 기존 RL보다 C에 대한 진화를 사용

우리의 C 모델이 M모델의 결점ㅇ르 악용하는 것을 더 어렵게 만들기 위해, 단지 결정적인 미래를 예측하기 보다는 실제 환경에서 가능한 결과의 분포를 모델링 하는 MDN-RNN as Dynamics model을 사용하기로 결정했다.

실제 환경이 결정적일 지라도, MDN-RNN은 사실상 이를 확률적 환경으로 근사함.

이것의 장점은 모든 환경의 확률적 버전 내에서 C모델을 훈련할 수 있다는 장점이 있음

temperature 매개변수 t 타우 를 조절하여 M 모델의 임의성의 양을 제어 할 수 있으므로, 현실성과 악용 가능성 간의 균형을 제어할 수 있다.

예를 들어, t를 매우 낮은값 0.1로 설정하면 이 꿈의 환경에 있는 몬스터는 어떤 일이 있어도 불덩어리를 쏘지 못함.

가상환경에서는 2100점 만점을 받지만, 현실세계는 실패함

테이블이(가) 표시된 사진

자동 생성된 설명

M 모델의 t를 높이면 C모델이 적대적 정책을 찾기가 더 어려워 지지만, 너무 많이 올리면 가상환경이 에이전트가 배우기가 너무 어려워진다.

t가 1.15일 때 best지만, t를 1.3으로 높이면 점수는 낮아지지만, score 분산이 낮아지는 전략을 가진다.

5. Iterative Training Procedure

우리가 실험한 작업은 비교적 간단한 환경이므로, 임의 정책에서 수집된 데이터 세트를 사용하여, 합리적인 세계모델을 학습할 수 있었다.

하지만 환경이 더욱 정교해지면, 에이전트가 전략적으로 탐색하는 방법을 배운 후에만, world의 일부만 사용할 수 있다.

복잡한 환경에서는 반복적인 학습 환경이 필요하다.

1. 임의의 매개변수 M, C 초기화

2. 실제 환경으로 N 번 roll out하고, 모든 액선 a\_t 와 관찰 x\_t를 storage 에 저장한다.

3. M을 모델 로 훈련하고, C를 훈련하여 M 내부의 예상 reawards를 최적화 한다.

4. 완료되지 않은 작업들이 있다면 2로 돌아간다.

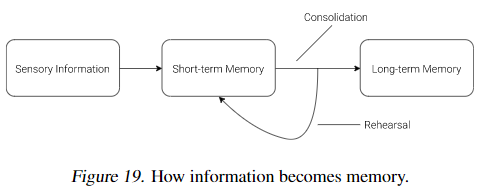
MDN-RNN이 제대로 수행되지 않으면, 에이전트가 익숙하지 않은 세계의 일부를 만났음을 의미한다.

일부러 이러한걸 위해, M의 훈련 손실 함수를 조정하고 재사용 할 수 있음.

실제 환경에서 M의 손실 함수의 부호를 뒤집어서, 일부로 익숙하지 않은 세계의 일부를 탐색하도록 권장함.

반복 훈련 절차는 M모델이 다음 관찰을 예측하고 완료될 뿐만 아니라, 다음 시간 단계에 대한 행동과 보상도 예측해야 함.

이것은 더 어려운 작업에 필요할 수 있다. 예를 들어, 에이전트가 주변을 걷기 위해 복잡한 운동기술을 배워야 하는 경우, 세계 모델은 이미 걷는 법을 배운 자체 C모델을 모방하는 법을 배운다. 걷기와 같은 어려운 운동 능력이 많은 용량을 가진 큰 world model에 흡수 된 후, 작은 C모델은 world model 에 이미 흡수된 운동 능력에 의존하고, 이미 가지고 있던 운동 능력을 사용하여 스스로 탐색하는 더 높은 수준의 기술을 배우는데 집중할 수 있다.

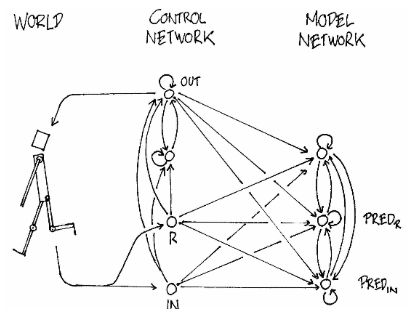


정보가 기억이 되는 방법

6. Related Work

PILCO(Deisenroth & Ras-mussen, 2011; Duvenaud, 2016; McAllister & Rasmussen,2016) 는 어려운 제어 문제를 해결하기 위해 설계된 확률 모델 기반 검색 정책 방법입니다.

PILCO는 환경에서 수집한 데이터를 사용하여 가우시안 프로세스 모델을 사용하여 시스템 역학을 학습한 다음, 이 모델을 사용하여 컨트롤러가 진자 휘두르기 또는 외발 자전거 타기와 같은 원하는 작업을 수행하도록 훈련하기 위해 많은 궤적을 샘플링 한다.



다른 최신 연구 (Gal et al.,2016; Depeweg et al., 2016)는 GP 대신 베이지안 신경망을 사용한다.

이 방법은 상태가 잘 알려져 있고, 잘 정의되고 관찰이 상대적으로 낮은 차원인 도전적인 제어 작업에서 좋은 결과를 보여줌.

~~ 최근 연구 관련 내용들이니까 한 번 읽어 보시면 좋을 듯.. (강화학습 흐름 파악 good)

7. Discussion

실제환경에서 에이전트를 훈련시키는 사이클을 낭비하지 않을 수 있다. 시뮬레이션된 환경에서 원하는 만큼 에이전트 훈련 가능.

비용 절약 가능.

게다가 헤를 사용하여, world model 시뮬레이션을 가속화 할 수 있음.

world model을 미분 가능한 재귀 계싼 그래프로 구현하는 이점은 역전파 알고리즘을 사용하여 목표 함수를 최대화하기 위해 정책을 미세 조정하여 world에서 agent를 훈련시킬 수 있다는 것을 의미함.