LG AI 해커톤
블록 장난감 제조 공정
최적화 AI 경진대회

팀명 : 춘

팀 소개

명지대학교 데이터사이언스 연구실

박민영 석사과정

안영근 석사과정

한국외국어대학교 최적화 연구실

강문정 석사과정

박현우 석사과정

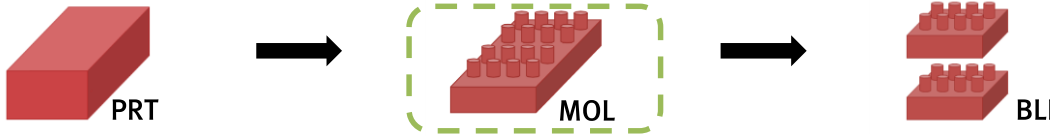
목차

1. 데이터 전처리 & EDA

DACON
DATA TO VALUE

- 어떤 종류의 MOL을 어느 시점에 얼만큼 만드는 것이 효율적일까?

MOL을 필요한 시점에 필요한 만큼만 생산하는 것이 효율적!



1. Order Recalculate

① 기초 재고 MOL을 BLK로 환산

② 총 계산된 기초 재고 BLK를 가장 최근 order부터 대체

⇒ 불필요한 생산을 줄일 수 있음

2. Calculate the NEED MOL

① 계산된 order 날짜를 기준으로 2일 전 날짜를 MOL 필요 생산 날짜로 정함 (∵MOL생산 48시간 소모)

② 날짜별 필요한 MOL 개수로 환산

3. Calculate the NEED MOL per hour

① 하루에 생산하는 MOL 1~4의 총 합이 200이 넘으면 필요한 MOL 누적 개수 증가시킴

② 시간별 필요한 MOL 개수로 환산

1.1 기초 재고 환산 예시

MOL_1	MOL_2	MOL_3	MOL_4	BLK_1	BLK_2	BLK_3	BLK_4
1086	0	0	0	61158	87279	0	0

MOL_1	MOL_2	MOL_3	MOL_4	BLK_1	BLK_2	BLK_3	BLK_4
0	0	0	0	528796	87279	0	0

1.2 최근 order 대체 예시

time	BLK_2 수요	BLK_2 재고
2020.4.14	0	87279
2020.4.15	18018	69261
2020.4.16	20437	48824
2020.4.17	12059	36765
2020.4.18	12059	24706
2020.4.19	12059	12647
2020.4.20	12059	588
2020.4.21	12059	0

필요 BLK_2
0
0
0
0
0
0
0
11471

2. 날짜별 필요한 MOL 개수 환산 예시

time	필요 BLK_2
2020.4.21	11471
2020.4.22	12059
2020.4.23	12059
2020.4.24	2953

time	필요 MOL_2
2020.4.19	26
2020.4.20	28
2020.4.21	28
2020.4.22	7

각 날짜별 최소한 생산했어야 하는 MOL 개수

time	필요 MOL_2 누적
2020.4.19	26
2020.4.20	54
2020.4.21	82
2020.4.22	89

3.1 필요한 MOL 개수 증가 예시

time	필요 MOL_1	필요 MOL_2	필요 MOL_3	필요 MOL_4	총합
2020.6.11	0	0	76	152	228

해당 날짜 3일 전 필요 누적 값을 바꿈

time	필요 MOL_3 누적	필요 MOL_4 누적
2020.6.11	1189	836

time	필요 MOL_3 누적	필요 MOL_4 누적
2020.6.6	1189	836

3.2 시간별 필요한 MOL 개수 환산 예시

time	필요 MOL_2 누적
2020.4.19	26

time	필요 MOL_2 누적
2020.4.18 19:00:00 ~ 2020.4.19 18:00:00	26

LG Science park
LG

1. 데이터 전처리 & EDA

- 효율적인 강화학습을 위한 데이터 활용

BRICK_MASK (BM) 날짜별 필요한 MOL 개수를 활용

날짜별 필요한 MOL 개수 예시

time	필요 MOL_1	필요 MOL_2	필요 MOL_3	필요 MOL_4
2020.4.19	0	26	0	0

시간별 BRICK_MASK 예시

time	BM_1	BM_2	BM_3	BM_4
2020.4.17 00:00:00 ~ 23:00:00	0	1	0	0

1. MOL을 생산하기 위해서는 미리 CHECK를 해야 하므로 필요 MOL 생산 2일 전에 해당 MOL에 대한 CHECK를 하도록 유도하기 위해 사용
2. 날짜별 필요한 MOL 개수가 0보다 크면 1 그렇지 않으면 0으로 설정
3. 강화학습에서 기존 MASK(공정 관련 제약)와 곱하여 사용

DIFF 시간별 필요한 (생산했어야 하는) MOL 누적 개수를 활용

DIFF = 현재 시점까지 실제로 생산한 해당 MOL 총합 – 현재 시점까지 필요한 해당 MOL 누적 개수

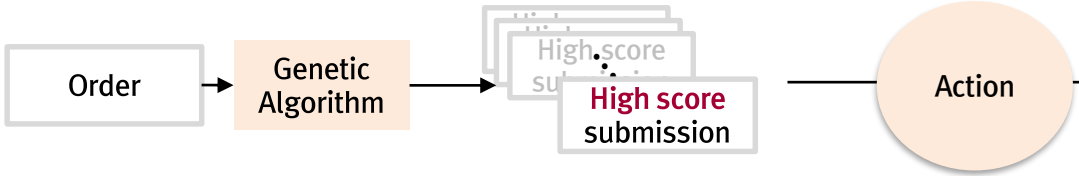
→ DIFF가 + 값이면 해당 MOL은 필요한 만큼은 생산됐으므로 더 이상 생산하지 않는 것이 좋음
→ DIFF가 – 값이면 해당 MOL이 필요한 만큼 생산되지 않았으므로 생산하는 것이 좋음

위의 정보를 이용하여 MOL 생산 개수를 결정할 때 MOL_DIFF가 + 값이면 생산 개수를 0으로, 월별 시간당 생산 가능한 양보다 작으면 (4월 : 5.8579개, 5월 : 5.8668개, 6월 : 5.8757개) 생산 가능한 양으로 설정
그 외의 경우에는 신경망에서 나온 아웃풋을 사용하여 필요한 만큼만 생산하도록 유도

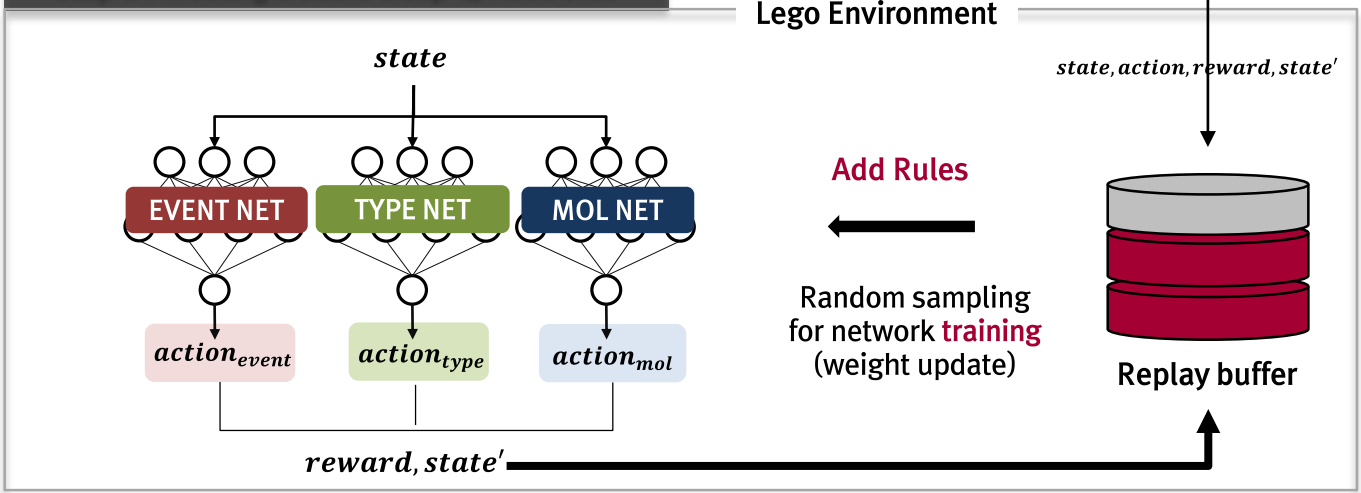
2. 모델구축 : Action Transfer Reinforcement Learning

- 모델의 전체 프로세스

Step 1 : Training Genetic Algorithm

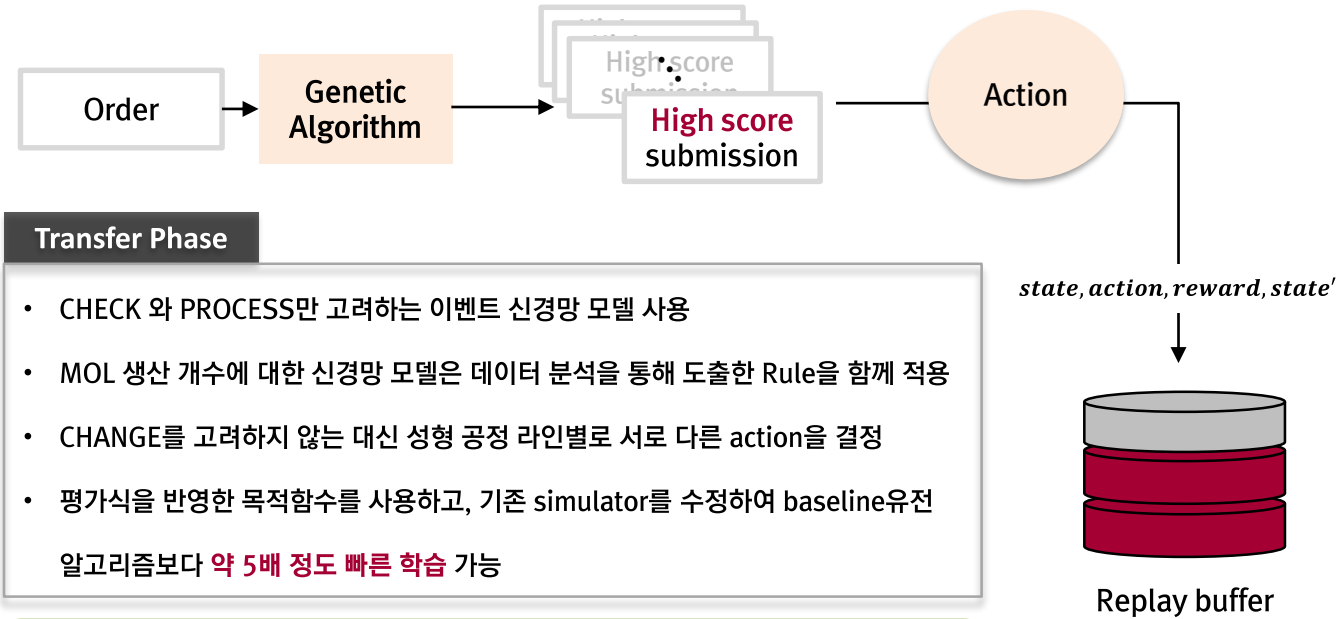


Step 2 : Training Double Deep Q-Networks



2. 모델구축 : Action Transfer Reinforcement Learning

Step 1 : Training Genetic Algorithm



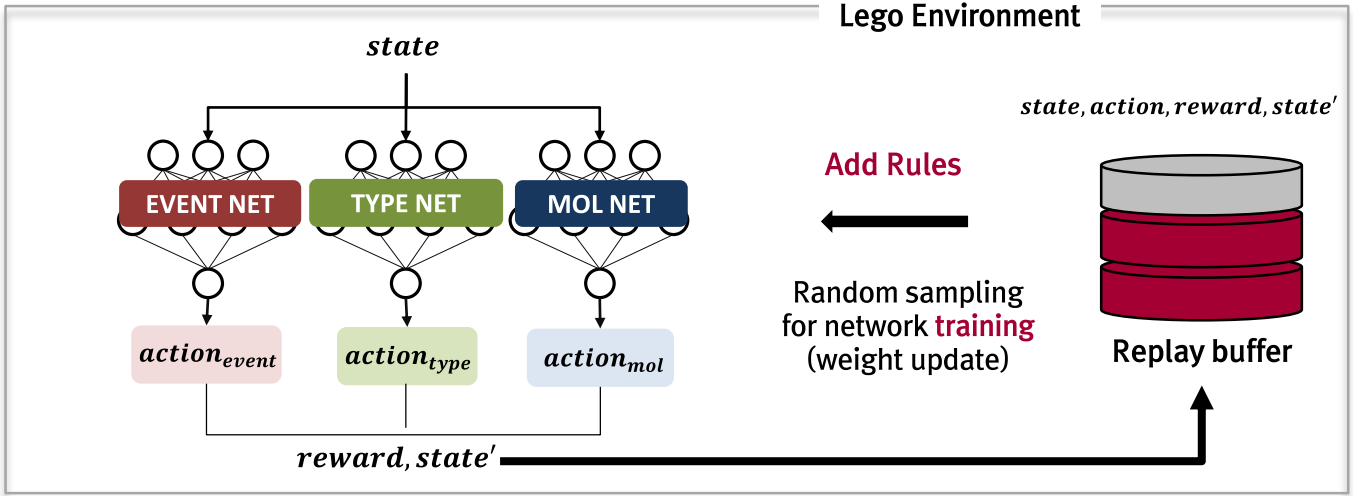
Transfer Phase

- CHECK 와 PROCESS만 고려하는 이벤트 신경망 모델 사용
- MOL 생산 개수에 대한 신경망 모델은 데이터 분석을 통해 도출한 Rule을 함께 적용
- CHANGE를 고려하지 않는 대신 성형 공정 라인별로 서로 다른 action을 결정
- 평가식을 반영한 목적함수를 사용하고, 기존 simulator를 수정하여 baseline유전 알고리즘보다 **약 5배 정도 빠른 학습** 가능

유전알고리즘 학습을 통해 얻은 85점 이상의 **high score submission action을** 토대로 강화 학습 요소를 buffer에 넣어 **action을 전이 시킴**

2. 모델구축 : Action Transfer Reinforcement Learning

Step 2 : Training Double Deep Q-networks



RL Phase

- 이벤트, 종류, 생산 개수를 결정하는 3가지 신경망 모델 사용
- 강화 학습 state는 order(124) + process(4) + time(1) + line_type(2) 총 131차원 vector 값을 사용
- 각 신경망 action에 의해 계산되는 reward가 다르며 환경에 영향을 줌

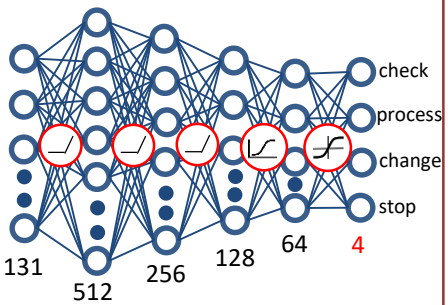
전이 학습을 통해 좋은 episode로 구성된 buffer를 이용하여 강화학습 진행

2. 모델구축 : Action Transfer Reinforcement Learning

DACON
DATA TO VALUE

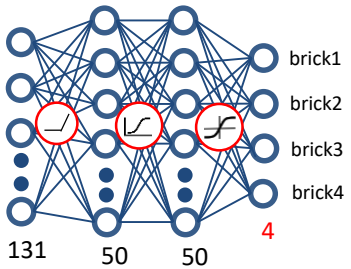
: ReLU : Sigmoid : Softmax

Event Network



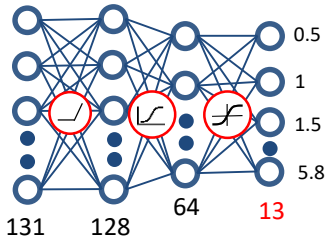
- 이벤트 결정 신경망은 총 4개의 hidden layer로 구성됨
- CHECK, PROCESS, CHANGE, STOP 중 하나의 **이벤트를 결정**
- Process의 q-value에 **가중치 α** 를 더해줌
- Output에 대한 reward는 Process면 +1 아니면 0

Brick Type Network



- 블록 타입 결정 신경망은 총 2개의 hidden layer로 구성됨
- Type1, Type2, Type3, Type4 중 하나의 **블록 타입 결정**
- 데이터 분석을 통해 도출한 **Brick Mask** 데이터를 활용하여 필요한 MOL 생산준비를 유도함
- Output에 대한 reward는 Brick Mask와 같으면 +1 아니면 0

MOL Amount Network



- MOL 생산량 결정 신경망은 총 2개의 hidden layer로 구성됨
- 0에서 5.5까지 0.5단위의 output 12개와 5.8을 합친 총 13개의 output 중 하나의 **생산량을 결정**
- 데이터 분석을 통해 도출한 **DIFF**를 활용하여 필요한 만큼만 MOL을 생산하도록 유도함
- Output에 대한 reward는 DIFF를 기준으로 만족하면 +1 아니면 0

LG Science park LG

2. 모델구축 : Action Transfer Reinforcement Learning

DACON
DATA TO VALUE

- Action Transfer Reinforcement Learning이 나오기까지 시도해본 다양한 알고리즘 및 구조

Knowledge based	Algorithm	Model architecture	Considerations
기존 데이터를 활용하여 모델의 성능을 높임	유전알고리즘 (Genetic Algorithm)	Single deep neural network	State : MOL 생산량, 재고 상태, 수요, 기계 타입, 시간, 공정 상태 등 고려
좋은 score의 데이터만 buffer에 입력 후 DQN의 network training	Double-Deep Q-Network (DDQN)	Recurrent neural network (RNN)	Reward : action 전후 score 비교, 수요 부족분, 수요 초과분, 제약 만족 상태 등 고려
모든 시점의 데이터를 buffer에 입력	Transfer learning	Two deep neural network (line A, line B)	Action : 이벤트 결정 시 CHANGE와 STOP을 추가하여 시도, 다양한 MOL 생산량을 고려
이벤트 결정 시 데이터를 buffer에 입력	Proximal Policy Optimization (PPO)		

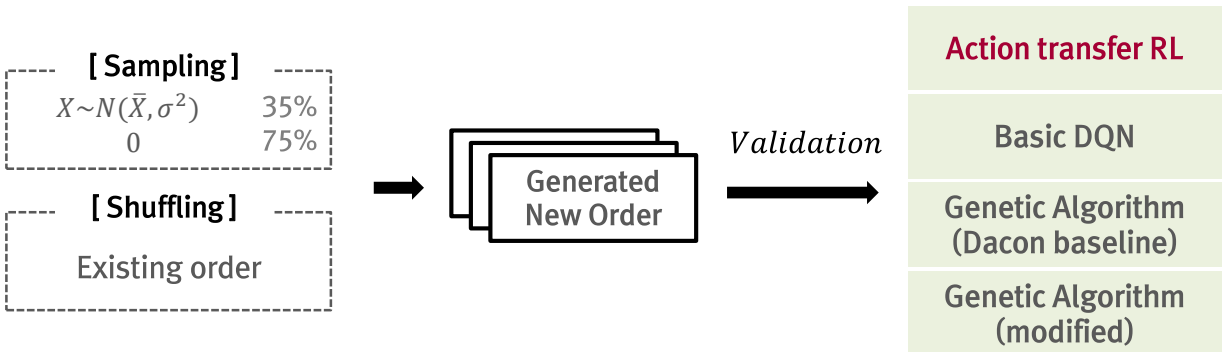
3. 모델 검증

- 모델 검증을 위해 생성한 새로운 order data

1. 4월 15일 이후 성형 공정 투입 가능 개수 (하루 약 240개) 와 BLK 당 order 발생 빈도 (약 35 %) 및 공정의 수용 능력을 고려하여 $240 * 453 \div 4 * 0.804 = 27180$을 평균으로 하는 $X \sim N(27180, 15000)$ 분포를 따르는 order 수량을 설정하여 새로운 order data를 생성

2. 기존 order의 수량 그대로 random shuffling하여 새로운 order data를 생성

- 위와 같이 새로운 order data를 생성하고 10회에 걸쳐 4가지 모델에 대한 성능 평가를 수행



3. 모델 검증 결과 및 적용가능성

DACON
DATA TO VALUE

Model validation

- Action transfer RL의 검증 결과 **학습시간**이나 **score**면에서 다른 알고리즘 보다 좋은 성능을 보임
- 전이학습을 통해 초기에 좋은 에피소드를 가지고 훈련을 할 수 있고 80점이 넘는 성능을 보임
- 데이터 분석을 통해 **도출된 rule을 적용할 경우 92점이 넘는 뛰어난 성능을 보임**
- 기존 order 뿐만 아니라 새로운 order에 대해서도 평균적으로 90점 정도의 우수한 성능을 보임

Applicability

- **다양한 order**에 대해서도 **robust한 공정설계 가능**
- 제안하는 모델은 실제 데이터를 기반으로 학습했기 때문에 학습했던 데이터와 같은 구조의 데이터를 input으로 넣어준다면 **현업에서도 바로 적용 가능**

[Results of experiment about new order]

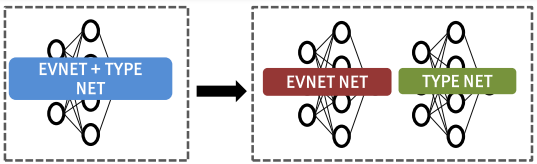
	Action transfer RL	Basic DQN	Genetic (Dacon)	Genetic (modified)
Training time(sec)	1.18	1.20	61.25	14.15
Best score	90.70	63.86	84.80	90.86

4. 독창성

Action Transfer Learning

- 서로 다른 신경망을 가지고 있는 두 개의 알고리즘으로 **기존에 없던 새로운 전이 학습을 시도함**
- 유전알고리즘을 통해 나온 submission의 action을 토대로 강화 학습 요소를 buffer에 추가함
- 이를 통해 초기에 좋은 episode를 찾기 위한 **exploration의 시간을 단축할 수 있음**
- 랜덤으로 초기화한 모델보다 안정적으로 훈련이 진행됨을 확인함

Multi task system



- 기존의 Dacon baseline으로 주어진 유전알고리즘 신경망 중에서 **이벤트 결정 신경망을 두개로 나누어 변경함**
- 같은 환경을 공유하는 독립적인 task를 가진 agent가 각각의 action을 학습할 수 있고 **task specific**한 특징을 가짐

Knowledge based architecture

- Heuristic solution을 찾기 위해 탐색해야 하는 **해공간을 효과적으로 줄일 수 있음**
- 강화 학습 훈련을 추가적으로 보조해주는 역할을 함

4. 확장성

DACON
DATA TO VALUE

Action transfer learning

- 신경망 구조가 동일하지 않아도 action의 결과를 공유할 수 있다면 유전알고리즘 외의 **어떤 알고리즘으로도 전이 학습이 가능함**
- 기존에 **보유중인 데이터가 있다면 이를 활용**하여 미리 전이 학습을 시킬 수도 있음
- 블록 공정 외에도 다른 공정 상황에 맞게 모델을 적용하면 확장이 가능할 것으로 기대함

Multi task system

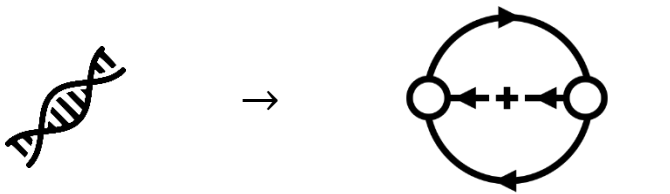
- Task의 특성 별로 신경망을 나누었기 때문에 **추가적인 task**에 대해서도 비교적 쉽게 확장 가능함
- 여러 종류의 타입이 있는 경우 기존 베이스라인 모델보다 효율적으로 추가 확장이 가능함

Knowledge based architecture

- 강화 학습의 경우 초기에 좋은 에피소드를 찾 훈련시키는 것이 굉장히 어렵기 때문에 **데이터 분석을 통해 해공간을 줄여** 좀 더 빠르고 정확한 결과값을 내도록 유도할 수 있음
- 공정에 대한 이해를 모델에 적용하면 새로운 환경에서도 빠른 학습이 가능할 것으로 기대함

5. 결론

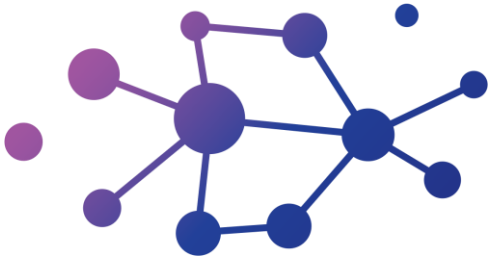
- 블록 장난감 제조 공정을 최적화하기 위한 AI 모델로 **Action Transfer Reinforcement Learning 알고리즘 개발**
- Action Transfer Reinforcement Learning은 **multi task를 처리**할 수 있는 독립적인 task를 가진 네트워크로 구성되어 더욱 효율적인 학습이 가능
- 서로 다른 신경망 구조를 가진 알고리즘이라도 같은 action을 공유할 수 있다면 전이 학습이 가능
- 정량적인 점수를 올리는 측면에서는 **기존 데이터를 활용한 rule**을 같이 적용해주는 것이 효과적
- Simulator 코드를 변경하여 실행 속도를 단축시킴으로써 빠른 학습 성능을 보임



Genetic Algorithm → Reinforcement Learning

6. 기타

- **실험 환경 및 hyper parameter (experiment setting)**
  - Mac OS Catalina on Mac Pro Late 2013 with 3.5 GHz 6-Core Intel Xeon E5, 32GB RAM
  - Genetic Algorithm : Python 3.7
  - Action Transfer RL : Double-Deep-Q-Network (DDQN) with Pytorch, Python 3.7
  - **Hyper parameter (Action Transfer Reinforcement Learning)**
    - Learning rate : 0.05, Optimizer : Adam Optimizer , Batch size : 256
    - Gamma : 0.9, Buffer size : 10000, epsilon decay : 1000



THANK YOU