# A REDUCTION OF IMITATION LEARNING AND STRUCTURED PREDICTION TO NO-REGRET ONLINE LEARNING (DAGGER)

임건호

2024.06.21

Reference Link

# 논문이 풀고자 하는 문제 (제목을 이해해보자)

Imitation Learning when expert policy exists!

- Reduction: Imitation Learning 문제를 근사하여 다른 문제로 바꾸는 것
- Regret: 매 순간순간 최선의 선택을 하는 것
  (cf. 헛된 일(Exploration)을 하면 후회 함)
- Online Learning: 모델이 전체 데이터를 보지 않고, 순차적으로 데이터를 받아들이는 학습 방법

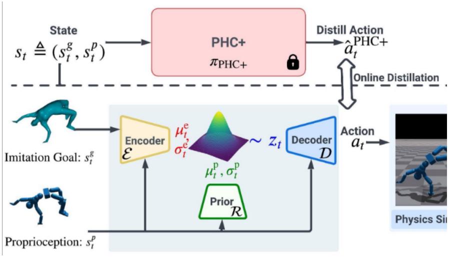
$$\frac{1}{N}\sum_{i=1}^{N}\ell_i(\pi_i) - \min_{\pi\in\Pi}\frac{1}{N}\sum_{i=1}^{N}\ell_i(\pi) \tag{1}$$

Regret 정의

# 그래서, 어디에 쓸모가 있을까?

## 네트워크 구조를 바꾸면서 이전 네트워크의 knowledge를 유지하고 싶을 때

- Catch & Carry (영상)
- Progressive RL (영상)
- PULSE, Neural Categorical Prior(NCP)
- Offline RL이 왜 잘 안되는지 이해할 수 있음




PULSE 학습 과정

# NAÏVE APPROACH OF IMITATION LEARNING

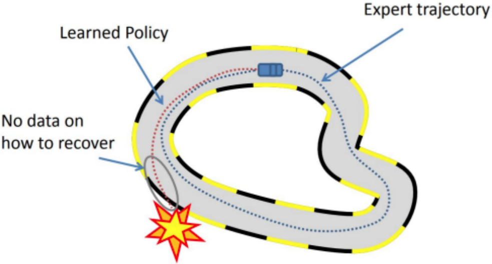
## Def. (Supervised Learning)

*$\pi$와 $\pi^*$이 동일한 state($s \sim d_{\pi^*}$)에서 활동한다고 가정하면,*

$$\hat{\pi}_{sup} = \arg\min_{\pi \in \Pi} \mathbb{E}_{s \sim d_{\pi^*}} [\ell(s, \pi)] \tag{2}$$

학습된 $\hat{\pi}$는 $\pi^*$와 학습 오차로 인해 state의 분포가 다르고,

벗어난 경로를 회복하는 action을 보지 못하여 trajectory 발산



## NOTATIONS

- $T$ the task horizon
- $d_\pi^t$ $t$ 시점의 state의 분포
- $d_\pi = \frac{1}{T}\sum_{t=1}^{T} d_\pi^t$ states의 평균
- $C(s, a)$ cost
- $C_\pi(s) = \mathbb{E}_{a\sim\pi(s)}[C(s, a)]$ C의 정의에서 $\pi$ 고정
- $C$ is bounded in $[0, 1]$
- $J(\pi) = \sum_{t=1}^{T} \mathbb{E}_{s\sim d_\pi^t}[C_\pi(s)] = T\mathbb{E}_{s\sim d_\pi}[C_\pi(s)]$
  $\pi$에 의한 state에 대한 전체 cost
- $\ell$ surrogate loss, C와 같을수도, 다를수도 있음
  $\Rightarrow \hat{\pi}_{sup}$은 $\pi^*$의 state에 대해 $\ell$을 최소화하는 $\pi$

# NAÏVE APPROACH

## Theorem (Naïve는 Quadratic Loss)

*$\ell(s, \pi)$ loss of $\pi$ with respect to $\pi^*$*

*$\mathbb{E}_{s \sim d_{\pi^*}}[\ell(s, \pi)] = \epsilon$ (학습 과정에서 발생한 최대 오차)*

$$J(\pi) \le J(\pi^*) + T^2 \epsilon \tag{3}$$

*Horizon 길이의 제곱에 비례하는 오차가 발생 (tight bound)*

## Proof

Let, $\ell(s, \hat{\pi}) = I(\hat{\pi}(s) \neq \pi^*(s))$ $\hat{\pi}$의 실수(mistake)에 대한 0-1 오차

확률 $p_t$: $\pi$가 첫 $t$-step 동안 $\pi^*$에 대해 실수를 하지 않음

$d_t$: $\hat{\pi}$이 실수를 하지 않았을 때 state의 분포

## Proof (cont'd)

$d'_t$: $\hat{\pi}$이 적어도 한번 이상 실수 했지만, $\pi^*$를 따라갈 때 분포

$\pi$가 $\pi^*$을 따라가면, 실수를 하거나, 하지 않으므로,

정리하면, $d^t_{\pi^*} = p_t d_t + (1 - p_{t-1}) d'_t$

실수를 할때 cost의 상한은 1, 실수를 하지 않으면 $\mathbb{E}_{s \sim d^\pi_t}(C^\pi_t(s))$

$$\text{따라서,} \quad J(\pi) \leq \sum_{t=1}^{T} [p_{t-1} \mathbb{E}_{s \sim d^\pi_t}(C^\pi_t(s)) + (1 - p_{t-1})].$$

Let, $\epsilon_i = \mathbb{E}_{s \sim d^i_{\pi^*}}[\ell(s, \hat{\pi})]$ for $i = 1, 2, \ldots, T$

$\pi^*$의 state에 대한 $\hat{\pi}$의 i 시점에서 오차

($\ell$의 정의에 의해 $\hat{\pi}$을 따라갈 때 t 시점에서 실수할 확률과 같음)

# cont'd.

$e_t$ / $e'_t$: state $d_t$ / $d'_t$에서 $\pi$가 실수할 확률 ($\epsilon_i$ 와 다르다)

t 시점에 $\pi$는 실수를 하거나, 하지 않으므로,

$$\mathbb{E}_{s\sim d_t^\pi}(C_t^\pi(s)) \leq \mathbb{E}_{s\sim d_t^\pi}(C_t^*(s)) + \epsilon_t,$$

또한, $\epsilon_t = p_{t-1}e_t + (1 - p_{t-1})e'_t \quad \rightarrow \quad p_{t-1}e_t \leq \epsilon_t$

추가로, $p_t = (1 - e_t)p_{t-1}$

그런데, 앞선 $d_t^{\pi^*}$ 계산식에 의하면,

$J(\pi^*) = \sum_{t=1}^T [p_{t-1}\mathbb{E}_{s\sim d_t^\pi}(C_t^*(s)) + (1 - p_{t-1})\mathbb{E}_{s\sim d_t^\pi}(C_t^*(s))]$ 이고

$$\Longrightarrow \sum_{t=1}^{T} p_t - 1\mathbb{E}_{s\sim d_t^\pi}(C_t^*(s)) \leq J(\pi^*).$$

## Proof (cont'd)

정리하면:

$$J(\pi) \le \sum_{t=1}^{T} \left[ p_{t-1} \mathbb{E}_{s \sim d_t^{\pi}} (C_t^{\pi}(s)) + (1 - p_{t-1}) \right]$$
$$\le J(\pi^*) + \sum_{t=1}^{T} \sum_{i=1}^{t} \epsilon_i$$
$$\le J(\pi^*) + T \sum_{t=1}^{T} \epsilon_t = J(\pi^*) + T^2 \epsilon. \quad (4)$$

($\epsilon = \frac{1}{T} \sum_{i=1}^{T} \epsilon_i$: 학습 오차의 평균)

이 증명은 논문에 포함된 6개 증명 중 하나로서
가장 쉬운(!) 증명이다.

# PREVIOUS WORKS

## FORWARD TRAINING

**Data:** $\pi_1^0, \ldots, \pi_T^0$ to query and execute $\pi^*$.

**for** $i = 1$ *to* $T$ **do**

  Sample $T$-step trajectories by following $\pi^{i-1}$;

  Get dataset $\mathcal{D} = \{(s_i, \pi^*(s_i))\}$ of states, actions taken by expert at step $i$;

  Train classifier $\pi_i^i = \arg\min_{\pi \in \Pi} \mathbb{E}_{s \sim \mathcal{D}}(\epsilon_\pi(s))$;

  $\pi_j^i = \pi_j^{i-1}$ for all $j \neq i$;

**end**

**return** $\pi_1^T, \ldots, \pi_T^T$;

# FORWARD TRAINING

i가 작을때는, $s\,\pi^*$ 위주로 학습하고,
i가 커질수록 $\pi^{i-1}$에 의해 학습된 데이터를 이용하여 학습

- $J(\pi) \leq J(\pi^*) + uT\epsilon$ (u는 대개 상수, T에 선형)
- $\pi$에 의한 오차를 $\pi^*$으로 복구하는 방법 학습
- T개의 classifier를 학습하므로, T가 큰 경우에 비효율적
- Motion VAE에서 Autoregressive하게 훈련하는 것은 $\pi_j^i = \pi_j^{i-1}$ 조건을 무시한 것으로 볼 수 있음

# STOCHASTIC MIXING ITERATIVE LEARNING (SMILE)

이전 policy를 stochastic하게 혼합하여 학습

**Data:** $\pi^0$ expert $\pi^*$로 초기화

**for** $i = 1$ *to* $N$ **do**

  Execute $\pi^{i-1}$ to get $\mathcal{D} = \{(s, \pi^*(s))\}$

  Train classifier $\hat{\pi}^i = \arg\min_{\pi \in \Pi} \mathbb{E}_{s \sim \mathcal{D}}(\epsilon_\pi(s))$

  $\pi^i = (1-\alpha)^i \pi^* + \alpha \sum_{j=1}^{i} (1-\alpha)^{i-j} \hat{\pi}^j$

**end**

Remove expert queries: $\tilde{\pi}^N = \frac{\pi^N - (1-\alpha)^N \pi^*}{1-(1-\alpha)^N}$ (정규화)

**return** $\tilde{\pi}^N$

- Normalize하여 결국 $\pi_0$ 제거
- T에 선형인 오차 bound
- 임의의 N을 사용할 수 있어 feasible한 알고리즘

# CAN WE DO BETTER?

# DAGGER ALGORITHM

$\hat{\pi}$가 실수할 수 있는 경로를 모두 합집합(Aggregate)한 데이터셋($\mathcal{D}$)을 이용하여 학습

**Data:** Initial dataset $\mathcal{D} \leftarrow \emptyset$.
**Data:** Initial policy $\hat{\pi}_1 \in \Pi$. (말그대로 임의의 policy)
**for** $i = 1$ **to** $N$ **do**
  Let $\pi_i = \beta_i \pi^* + (1 - \beta_i)\hat{\pi}_i$;
  Sample $T$-step trajectories using $\pi_i$;
  Get dataset $\mathcal{D}_i = \{(s, \pi^*(s))\}$ of visited states by $\pi_i$;
  Aggregate datasets: $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}_i$;
  Train classifier $\hat{\pi}_{i+1}$ on $\mathcal{D}$;
**end**
**Result:** Best $\hat{\pi}_i$ on validation

## DAGGER ALGORITHM

- 이전(*Forward, SMILe*)은 $\mathcal{D}$를 한번 사용하고 버렸지만, DAgger는 계속해서 사용
- 일반적으로 $\beta_i = p^{i-1}$로 설정 ($\pi_1$을 임의로 설정)
- No Regret (Asymptotic Optimal, Stable)
  $\Leftrightarrow$ Immediate/Expected Loss minimization

$$\frac{1}{N}\sum_{i=1}^{N} \ell_i(\pi_i) - \min_{\pi\in\Pi}\frac{1}{N}\sum_{i=1}^{N}\ell_i(\pi) \le \gamma_N \lim_{N\to\infty} 0 \qquad (5)$$

# DAGGER IS NO REGRET

**Theorem (Follow the Leader(FTL))**

$\pi_N = argmin \sum_{i=1}^{N-1} \ell_i(\pi)$

FTL is no regret algorithm (다른 paper에서 증명)
DAgger은 전체 데이터셋을 optimize하여 학습하므로,
FTL을 따르면서 학습하므로 No Regret 성질을 가짐

# DAGGER ALGORITHM은 EXPERT에 수렴 (PROOF)

Let $\epsilon_N = \min_{\pi \in \Pi} \frac{1}{N} \sum_{i=1}^{N} \mathbb{E}_{s \sim d_{\pi_i}} [\ell(s, \pi)]$
the true loss of the best policy in hindsight.
(돌이켜 봤을 때 가장 좋은 policy(student+expert mix)의 loss)

**Theorem (Student loss는 $\epsilon_N$에 수렴)**

*For* DAGGER, *if $N$ is $\tilde{O}(T)$,* $\exists \hat{\pi} \in \hat{\pi}_{1:N}$ *s.t.*
$\mathbb{E}_{s \sim d_{\hat{\pi}}} [\ell(s, \hat{\pi})] \leq \epsilon_N + O(1/T)$
*($\tilde{O}(T)$는 $\exists k$ s.t. $N = O(T \cdot \log^k(T))$, polylogarithmic in $T$)*

**Theorem (Total Cost역시 수렴)**

*if $N$ is $\tilde{O}(uT)$,* $\exists \hat{\pi} \in \hat{\pi}_{1:N}$ *s.t.*
$J(\hat{\pi}) \leq J(\pi^*) + uT\epsilon_N + O(1).$

# DAGGER ALGORITHM - FINITE SAMPLE (PROOF)



Trajectory를 모두 sample 할 수 없음 (finite sample, m)
$\Rightarrow \hat{\epsilon}_N$에 대해 앞선 부등식 증명 가능

앞선 부등식을 만족하는 $\hat{\pi}$은 확률적으로 존재

# DAGGER ALGORITHM - ONLINE LEARNING (PROOF)

DAGGER의 No Regret 성질로 more tight upper bound

$\Rightarrow \pi_{1:N}$ 중에 가장 좋은 policy와 비슷한 성능

**Theorem (State 분포의 차이는 Bounded)**

$||d_{\pi_i} - d_{\hat{\pi}_i}||_1 \leq 2T\beta_i$

**Theorem (DAgger Upper Bound)**

$\exists \hat{\pi} \in \hat{\pi}_{1:N}$ *s.t.*

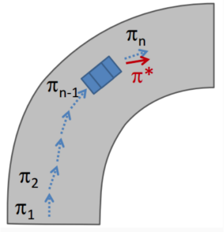
$\mathbb{E}_{s \sim d_{\hat{\pi}}}[\ell(s, \hat{\pi})] \leq \epsilon_N + \gamma_N + \frac{2\ell_{max}}{N}[n_\beta + T \sum_{i=n_\beta+1}^{N} \beta_i]$,

($\gamma_N$ = *average regret of* $\hat{\pi}_{1:N}$)

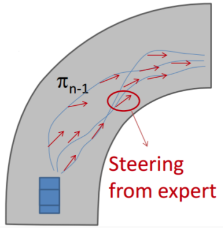
$N \rightarrow \infty$일때, 두번째, 세번째 항은 0으로 수렴

Finite Sample에서도 $\hat{\pi}$는 $\epsilon_N$에 수렴

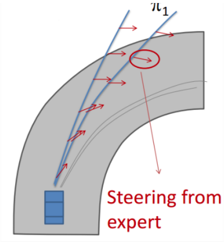
# VISUALIZE



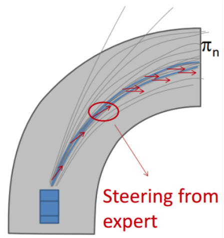

(a) Forward Training




(b) SMILe




(c) Initial DAgger



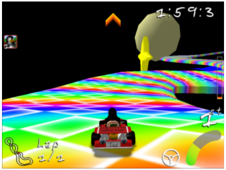

(d) Last DAgger

# RESULT

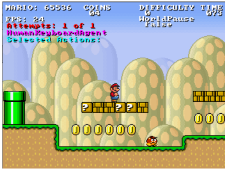
# TASKS

Imitation Learning 문제와 라벨링 문제

- Super Tux Kart: (Image) $\rightarrow$ (Joystick)
- 슈퍼 마리오: (Image) $\rightarrow$ (4 방향)
- Handwriting 인식: (Image) $\rightarrow$ (Class)



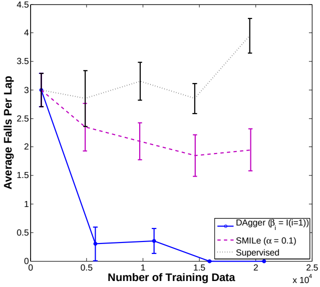
(a) Super Tux Kart



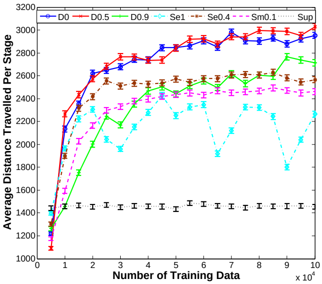

(b) Super Mario

# RESULTS

DAgger(파란색)가 다른 방법보다 더 나은 성능을 보임




(a) Super Tux Kart
(Falls Per Lap)




(b) Super Mario
(Travelled Stage)

# AGGREVATE (RL VERSION)

Current Cost 뿐만 아니라 Future Cost까지 고려한 학습 (Cost-to-go)

**Data:** Initialize $\mathcal{D} \leftarrow \varnothing$, $\hat{\pi}_1$ to any policy in $\Pi$.

**for** $i = 1$ **to** $N$ **do**

Let $\pi_i = \beta_i \pi^* + (1 - \beta_i)\hat{\pi}_i$ **for** $j = 1$ **to** $m$ **do**

Sample $t \in \{1, 2, \ldots, T\}$;

Start new trajectory from initial state distribution;

Execute $\pi_i$ up to time $t - 1$;

Exploration action $a_t$ ;

Execute expert from $t + 1$ to $T$;

Estimate of cost-to-go $\hat{Q}$ from $t$;

**end**

Dataset $\mathcal{D}_i = \{(s, t, a, \hat{Q})\}$ (이후 Dagger과 동일) ;

**end**

**return** best $\hat{\pi}_i$ on validation.

# BY THE WAY, HOW WE HANDLED THE PROBLEM OF IMITATION LEARNING?

## GAIL
## (GENERATIVE ADVERSARIAL IMITATION LEARNING)

AMP, ASE, PHC... 등이 사용하는 방법으로서,
GAN의 reward를 통해 expert trajectory로 guide

문제

1. RL을 사용하여야 함
2. GAN의 mode-collapse로 인해 diversity ↓

$\Longrightarrow$ GAIL은 dataset만 가지고 있을 때,
$\pi^*$를 생성하는 문제에 대한 방법론임
$\therefore$ expert를 가지고 있을 때에는 굳이 사용할 필요 없음

# SUPERVISED LEARNING

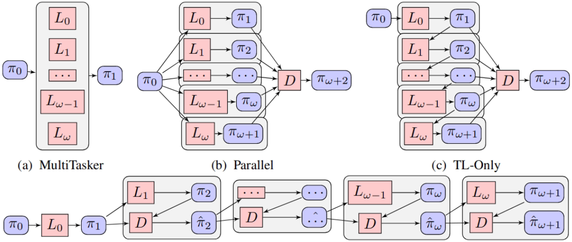
MotionVAE, ControlVAE 등이 사용하는 방법으로서,
autoregressive하게 네트워크의 출력을 입력으로 주어
그 결과가 dataset을 따라가게 gradient 부여

$\Longrightarrow$ DAgger은 $\pi^*$를 가지고 있을 때
Supervised Learning을 잘 하기 위한 방법
(이렇게 할 일이 있을까?)

# EXPERT를 가지고 있을 때에는?

## Distillation이라는 용어를 사용한 작업들

| | 목적 |
|---|---|
| **Catch&Carry** | Policy input: marker $\rightarrow$ Image |
| **Progressive** | Adapt Terrain |
| **NCP** | Posterior(future frame) $\rightarrow$ Prior(no future) |
| **PULSE** | Policy structure: MoE $\rightarrow$ VAE |


(a) MultiTasker (b) Parallel (c) TL-Only
(d) PLAiD

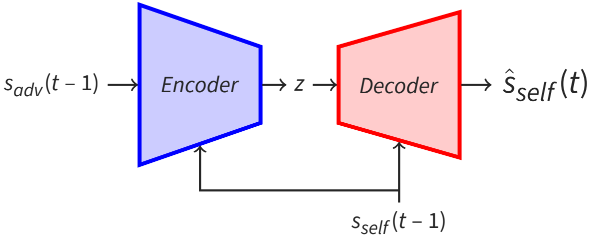
Progressive RL 학습 Curriculum (D: distillation, L: learning)

# 나의 연구에 어떻게 적용할 수 있을까?

목적: 두 캐릭터에 대해 캡쳐된 모션을 사용하여

1. N 캐릭터가 상호작용 하는 모션 생성 혹은
2. 상대 캐릭터 행동에 적절한 반응을 하는 모션 생성

현재 구상하는 구조




하나의 네트워크로 학습하면, 결과가 좋지 못하다.

Posterior collapse + 동작 6개 학습하는데 8시간 소요

# ABULATION STUDY ON PULSE

PULSE에서는 $\pi_{PHC+}$를 distill 하여 $\pi_{PULSE}$를 학습

Distill하지 않고, naïve하게 학습한 결과

| | AMASS-Train* | | | | | AMASS-Test* | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Distill | Succ ↑ | $E_{\text{g-mpjpe}}$ ↓ | $E_{\text{mpjpe}}$ ↓ | $E_{\text{acc}}$ ↓ | $E_{\text{vel}}$ ↓ | Succ ↑ | $E_{\text{g-mpjpe}}$ ↓ | $E_{\text{mpjpe}}$ ↓ | $E_{\text{acc}}$ ↓ | $E_{\text{vel}}$ ↓ |
| ✗ | 72.0% | 76.7 | 52.8 | 3.5 | 8.0 | 32.6% | 98.4 | 79.4 | 9.9 | 16.2 |
| ✓ | 99.8 % | 39.2 | 35.0 | 3.1 | 5.2 | 97.1% | 54.1 | 43.5 | 7.0 | 10.3 |

(논문 주장) Latent와 Recon이 동시에 학습이 잘 안됨

⇒ 나의 연구에도 시사하는 바가 있음,

Knowledge를 잘 전달하기 위해서는 어떻게 해야할까?

# THANK YOU