APRENDIZAJE POR IMITACIÓN Y APRENDIZAJE REFORZADO BASADO EN MODELOS

EL7021: Seminario de robótica y sistemas autónomos

Francisco Leiva[2] Javier Ruiz-del-Solar[1,2]

[1]Departamento de Ingeniería Eléctrica, Universidad de Chile

[2]Advanced Mining Technology Center (AMTC), Universidad de Chile

Mayo, 2023

Aprendizaje por imitación

Imitation Learning

- El aprendizaje por imitación aborda el problema de aprender un comportamiento a partir de demostraciones.

- Típicamente se divide en dos ramas:
  - Behavioral Cloning (BC)
    - El objetivo es aprender una política a partir de demostraciones expertas (pares observación-acción).
  - Inverse Reinforcement Learning (IRL)
    - El objetivo es recuperar una función de recompensa a partir de demostraciones expertas.

- También hay *model-free* y *model-based* IL.

Behavioral Cloning

- Dada una base de datos de demostraciones expertas, $\mathcal{D} = \{(o_i, a_i)\}_{i=1}^{M}$, se busca aprender una política $\pi(a|o)$.

- Lo anterior se puede formular como un problema de aprendizaje supervisado, donde la política es obtenida solucionando un problemas de clasificación/regresión.

- Representando a la política mediante $\pi_\theta(a|o)$, se busca entonces resolver un problema de optimización a través del ajuste de los parámetros θ.

Behavioral Cloning

Algoritmo 1: Behavioral Cloning

Obtener una base de datos $\mathcal{D}$ con demostraciones expertas
Inicializar política π_θ con parámetros θ
Seleccionar una función objetivo ad-hoc $\mathcal{L}(\theta)$
Optimizar $\mathcal{L}(\theta)$ usando los datos contenidos en $\mathcal{D}$

Ejemplos de funciones objetivo:

- Política determinista, acciones continuas

$$\mathcal{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N}\left(\pi_\theta(o_i) - a_i\right)^2$$

- Política estocástica, acciones discretas

$$\mathcal{L}(\theta) = -\frac{1}{N}\sum_{i=1}^{N}\log(\pi_\theta(a = a_i|o_i))$$

El problema con Behavioral Cloning

- Obtener un dataset $\mathcal{D}$ que sea representativo puede ser muy complejo.
- Errores de estimación al desplegar la política generan un “error compuesto”, por diferencias entre datos de entrenamiento y prueba.
- Una forma de aliviar lo anterior es ir agregando datos (y etiquetarlos) conforme el agente los observa (esto se conoce como “dataset aggregation”).

Dataset Aggregation

Algoritmo 2: BC + DAgger

Obtener una base de datos $\mathcal{D}$ con demostraciones expertas
Inicializar política π_θ con parámetros θ
Seleccionar una función objetivo ad-hoc $\mathcal{L}(\theta)$
for *k=1, K* **do**
    Optimizar $\mathcal{L}(\theta)$ usando los datos contenidos en $\mathcal{D}$
    Desplegar π_θ y obtener D_π a través de un experto
    $D \leftarrow D \cup D_\pi$
end

Objetivo del aprendizaje reforzado

- Recordemos el objetivo del aprendizaje reforzado:

$$\underbrace{p_\pi(s_1, a_1, ..., s_T, a_T)}_{p_\pi(\tau)} = p(s_1) \prod_{t=1}^{T} \pi(a_t|s_t) p(s_{t+1}|s_t, a_t)$$

$$J_{\text{RL}}(\pi) = \mathbb{E}_{\tau \sim p_\pi(\tau)} \left[\sum_{t=1}^{T} \gamma^{t-1} r(s_t, a_t) \right]$$

- ¿Y si tuvieramos acceso a $p(s_{t+1}|s_t, a_t)$?

Recordatorio: Taxonomía de los algoritmos de RL

Model-based vs Model-free

Hacen o no uso de un modelo del ambiente.

Recordatorio: Taxonomía de los algoritmos de RL

Model-based vs Model-free

Hacen o no uso de un modelo del ambiente.

Value-based

Aproximan $V^*(s)$ o $Q^*(s, a)$ para derivar una política.

Recordatorio: Taxonomía de los algoritmos de RL

Model-based vs Model-free

Hacen o no uso de un modelo del ambiente.

Value-based

Aproximan $V^*(s)$ o $Q^*(s, a)$ para derivar una política.

Policy gradient

Buscan $\pi(a|s)$ a través de la optimización directa de $J_{\text{RL}}(\pi)$.

Recordatorio: Taxonomía de los algoritmos de RL

Model-based vs Model-free

Hacen o no uso de un modelo del ambiente.

Value-based

Aproximan $V^*(s)$ o $Q^*(s, a)$ para derivar una política.

Policy gradient

Buscan $\pi(a|s)$ a través de la optimización directa de $J_{\text{RL}}(\pi)$.

Actor-Critic

Aproximan conjuntamente $V^*(s)$ o $Q^*(s, a)$ y una política $\pi(a|s)$.

Aprendizaje reforzado basado en modelo

¿Cómo emplear un modelo?

- Múltiples opciones:
  - Planificación.
  - Aprendizaje de políticas.
  - ...

- Aprendizaje reforzado basado en modelo:
  - Aborda el problema del aprendizaje reforzado haciendo uso de un modelo del ambiente.
  - El modelo puede ser dado, o aprendido.

Panificación

- El problema de planificación consiste en encontrar una secuencia de acciones que maximicen el retorno del agente al interactuar con el ambiente.

- Ejemplos:
  - *Random shooting*:
    - Se muestrean N secuencias de acciones $A^{(1)}, ..., A^{(N)}$ a partir de una cierta distribución.
    - Se selecciona aquella secuencia que maximice la recompensa total que sería obtenida al ejecutarla.
  - *Linear Quadratic Regulator* (LQR).
  - *Monte Carlo Tree Search* (MCTS).

MBRL + Panificación

Empleando un modelo para planificar, es posible llegar a un algoritmo simple:

Algoritmo 3: MBRL (ejemplo 1)

Inicializar modelo p_θ con parámetros θ
Inicializar *buffer* $\mathcal{D}$
Correr política $\pi_{\text{base}}(a|s)$ para obtener tuplas (s_t, a_t, s_{t+1}), y guardarlas en $\mathcal{D}$
Ajustar modelo p_θ usando los datos contenidos en $\mathcal{D}$
Emplear modelo para planificar (y así, seleccionar acciones)

- ¿Problemas con este algoritmo?

MBRL + Panificación

Para aliviar el problema de distribuciones diferentes, se puede hacer uso de *dataset aggregation* (DAgger).

Algoritmo 4: MBRL (ejemplo 2)

Inicializar modelo p_θ con parámetros θ
Inicializar *buffer* $\mathcal{D}$
Correr política $\pi_{\text{base}}(a|s)$ para obtener tuplas (s_t, a_t, s_{t+1}), y guardarlas en $\mathcal{D}$
Ajustar modelo p_θ usando los datos contenidos en $\mathcal{D}$
for *i=1, N* **do**
    Emplear modelo para planificar (y así, seleccionar acciones)
    Ejecutar acciones y guardar las tuplas (s_t, a_t, s_{t+1}) resultantes en $\mathcal{D}$
    if $i \,\%k == 0$ **then**
        Ajustar modelo p_θ usando los datos contenidos en $\mathcal{D}$
    end
end

- ¿Problemas con este algoritmo?

MBRL + Panificación

Algoritmo 5: MBRL (ejemplo 3)

Inicializar modelo p_θ con parámetros θ
Inicializar *buffer* $\mathcal{D}$
Correr política $\pi_{\text{base}}(a|s)$ para obtener tuplas (s_t, a_t, s_{t+1}), y guardarlas en $\mathcal{D}$
Ajustar modelo p_θ usando los datos contenidos en $\mathcal{D}$
for *i=1, N* **do**
    Emplear modelo para planificar (y así, seleccionar acciones)
    Solo ejecutar la primera acción planificada (MPC)
    Guardar la tupla (s_t, a_t, s_{t+1}) resultante en $\mathcal{D}$
    if $i \,\%k == 0$ **then**
        Ajustar modelo p_θ usando los datos contenidos en $\mathcal{D}$
    end
end

Aprendizaje de políticas

- Otras formas en las que es posible emplear un modelo incluyen:
  - Para generar *rollouts* en algoritmos *model-free*.
  - Para “destilar” políticas globales a partir de políticas locales.

Algoritmos tipo “Dyna”

Algoritmo 6: MBRL (Dyna)

Inicializar modelo p_θ con parámetros θ
Inicializar *buffer* $\mathcal{D}$
Correr política $\pi_{\text{base}}(a|s)$ para obtener tuplas (s_t, a_t, s_{t+1}), y guardarlas en $\mathcal{D}$
Ajustar modelo p_θ usando los datos contenidos en $\mathcal{D}$
for *i=1, N* **do**
    Muestrear s_t de $\mathcal{D}$
    Elegir acción a_s (de $\mathcal{D}$, o según π, o de otra forma)
    Predecir s_{t+1} según p_θ
    Usar transición generada para entrenar usando algún algoritmo *model-free*
    (Opcionalmente se pueden generar más transiciones empleando el modelo)
end