Ejemplo Sencillo: Un MDP de Corredor Simple con Diferencias On-policy y Off-policy

Este documento proporciona un ejemplo concreto que ilustra conceptos clave de Reinforcement Learning (RL), destacando la diferencia entre aprendizaje on-policy y off-policy.

---

1. Planteamiento del Problema

```
[L] -- [M] -- [R] -- [T]
```

Representación ASCII del corredor: L (Izquierda), M (Medio, inicio), R (Derecha), y T (Meta terminal).

Un agente se encuentra en un corredor de 3 celdas: Izquierda (L), Medio (M), Derecha (R). El objetivo es llegar al extremo derecho (T) partiendo desde M. Cada paso de tiempo incurre en un costo (recompensa negativa) hasta que se alcanza la meta.

- **Estados**: $\mathcal{S} = {L, M, R, T}$, donde T es el estado terminal o meta.
- **Acciones**: $\mathcal{A} = {\text{Left}, \text{Right}}$ (Izquierda, Derecha).
- **Estado inicial**: $S_0 = M$.

---

2. Formulación del MDP

Definimos el MDP $(\mathcal{S},\mathcal{A},P,R,\gamma)$:

1. **Probabilidades de transición**: Deterministas - moverse en la dirección indicada o quedarse en el mismo lugar si se llega a un límite.

PROF

2. **Función de recompensa**:
   $
   R(s,a,s') = \begin{cases}
   -1 & \text{para cualquier transición que no lleve a }T,\
   0 & \text{para transiciones hacia }T.
   \end{cases}
   $

3. **Factor de descuento**: $\gamma = 0.9$.

---

3. Comparación On-Policy vs Off-Policy

Para ilustrar la diferencia, consideremos dos políticas diferentes:

Política óptima (π)

Una política determinista que siempre elige Right:
$\pi(a|s) = \begin{cases} 1 & \text{si } a = \text{Right} \ 0 & \text{si } a = \text{Left} \end{cases}$ para todo estado $s \in {L,M,R}$.

Política exploratoria (μ)

Una política estocástica que explora ocasionalmente:
$\mu(a|s) = \begin{cases} 0.8 & \text{si } a = \text{Right} \ 0.2 & \text{si } a = \text{Left} \end{cases}$ para todo estado $s \in {L,M,R}$.

---

4. Diferencia Conceptual: On-Policy vs Off-Policy

Aprendizaje On-Policy

- **Definición**: Aprendemos sobre la misma política que estamos siguiendo.
- **Ejemplo**: Si seguimos μ (política exploratoria), aprendemos los valores de μ.
- **Característica**: Lo que aprendo es exactamente sobre lo que hago.
- **Función de valor**: $V^{\mu}(s)$ - valor del estado s bajo la política μ.
- **Función de valor-acción**: $Q^{\mu}(s,a)$ - valor de tomar la acción a en el estado s y luego seguir μ.

Aprendizaje Off-Policy

- **Definición**: Aprendemos sobre una política mientras seguimos otra diferente.
- **Ejemplo**: Seguimos μ (exploramos), pero aprendemos los valores de π (política óptima).
- **Característica**: Puedo explorar con una política, pero aprender sobre otra.
- **Función de valor**: $V^{\pi}(s)$ - valor del estado s bajo la política π (que no estamos siguiendo).
- **Función de valor-acción**: $Q^{\pi}(s,a)$ - valor de tomar la acción a en el estado s y luego seguir π.

PROF

5. Ecuaciones de Bellman para Ambas Políticas

Ecuaciones de Bellman para π

Dado que π siempre elige Right:

$V^{\pi}(s) = \sum_{a} \pi(a|s) \sum_{s'} P(s'|s,a)[R(s,a,s') + \gamma V^{\pi}(s')]$

Que se simplifica a:

$V^{\pi}(s) = \sum_{s'} P(s'|s,\text{Right})[R(s,\text{Right},s') + \gamma V^{\pi}(s')]$

Ecuaciones de Bellman para μ

Como μ puede elegir ambas acciones:

$V^{\mu}(s) = \sum_{a} \mu(a|s) \sum_{s'} P(s'|s,a)[R(s,a,s') + \gamma V^{\mu}(s')]$

Que se expande a:

$V^{\mu}(s) = 0.8 \sum_{s'} P(s'|s,\text{Right})[R(s,\text{Right},s') + \gamma V^{\mu}(s')] + 0.2 \sum_{s'} P(s'|s,\text{Left})[R(s,\text{Left},s') + \gamma V^{\mu}(s')]$

---

6. Comparación de Valores

Valores bajo la política óptima π

Si siempre vamos a la derecha (siguiendo π):

- Desde M, llegamos a T en dos pasos: M → R → T
- $V^{\pi}(M) = -1.9$
- $V^{\pi}(R) = -1.0$
- $Q^{\pi}(M,\text{Right}) = -1.9$
- $Q^{\pi}(M,\text{Left}) = -2.71$ (porque después del primer paso, seguiríamos π)

Valores bajo la política exploratoria μ

Si seguimos μ (80% derecha, 20% izquierda):

- A veces tomaremos desvíos: M → L → M → R → T
- $V^{\mu}(M) \approx -2.3$ (peor que $V^{\pi}(M)$ porque a veces tomamos acciones subóptimas)
- $V^{\mu}(R) \approx -1.2$ (peor que $V^{\pi}(R)$ por la misma razón)
- $Q^{\mu}(M,\text{Right}) \approx -2.1$
- $Q^{\mu}(M,\text{Left}) \approx -3.0$

---

7. Ejemplos Detallados de Actualizaciones On-Policy y Off-Policy

Para ilustrar concretamente la diferencia entre métodos on-policy y off-policy, vamos a ver paso a paso cómo funcionarían SARSA (on-policy) y Q-learning (off-policy) en nuestro ejemplo del corredor.

PROF

Ejemplo On-Policy: SARSA

SARSA es un algoritmo on-policy que actualiza valores Q basados en las acciones realmente tomadas según la política actual.

Fórmula de actualización SARSA:
$Q^{\mu}(s_t, a_t) \leftarrow Q^{\mu}(s_t, a_t) + \alpha [r_{t+1} + \gamma Q^{\mu}(s_{t+1}, a_{t+1}) - Q^{\mu}(s_t, a_t)]$

Ejemplo concreto:
Supongamos que seguimos la política exploratoria μ y estamos en el estado M. Inicializamos todos los valores Q^{μ} en -1.0.

1. **Estado actual**: $s_t = M$
2. **Elegimos acción** según μ: $a_t = \text{Right}$ (con probabilidad 0.8)

3. **Observamos**: $r_{t+1} = -1$, $s_{t+1} = R$
4. **Elegimos siguiente acción** según μ: $a_{t+1} = \text{Right}$ (con probabilidad 0.8)
5. **Actualización SARSA** (con $\alpha = 0.1$):
   $Q^{\mu}(M, \text{Right}) \leftarrow -1.0 + 0.1 [-1 + 0.9 \times Q^{\mu}(R,\text{Right}) - Q^{\mu}(M, \text{Right})]$
   $Q^{\mu}(M, \text{Right}) \leftarrow -1.0 + 0.1 [-1 + 0.9 \times (-1.0) - (-1.0)]$
   $Q^{\mu}(M, \text{Right}) \leftarrow -1.0 + 0.1 [-1 - 0.9 + 1.0]$
   $Q^{\mu}(M, \text{Right}) \leftarrow -1.0 + 0.1 [-0.9]$
   $Q^{\mu}(M, \text{Right}) \leftarrow -1.0 - 0.09 = -1.09$

Después de muchas iteraciones, $Q^{\mu}(M, \text{Right})$ convergerá a $Q^{\mu}(M, \text{Right}) \approx -2.1$, el valor real bajo la política μ.

Importante: En SARSA, usamos la acción a_{t+1} que realmente tomaremos según μ, incluso si no es la óptima. Esto hace que SARSA aprenda el valor real de seguir μ, incluyendo sus movimientos exploratorios.

Ejemplo Off-Policy: Q-learning

Q-learning es un algoritmo off-policy que actualiza valores Q basándose en la acción óptima para el siguiente estado, independientemente de qué política estamos siguiendo.

Fórmula de actualización Q-learning:
$Q^(s_t, a_t) \leftarrow Q^(s_t, a_t) + \alpha [r_{t+1} + \gamma \max_a Q^(s_{t+1}, a) - Q^(s_t, a_t)]$

Ejemplo concreto:
Misma situación que antes, seguimos μ pero aprenderemos valores Q óptimos Q^* (que corresponden a la política óptima π).

1. **Estado actual**: $s_t = M$
2. **Elegimos acción** según μ: $a_t = \text{Right}$ (con probabilidad 0.8)
3. **Observamos**: $r_{t+1} = -1$, $s_{t+1} = R$
4. **Actualización Q-learning** (con $\alpha = 0.1$):
   $Q^(M, \text{Right}) \leftarrow Q^(M, \text{Right}) + \alpha [r_{t+1} + \gamma \max_a Q^(R, a) - Q^(M, \text{Right})]$
   $Q^(M, \text{Right}) \leftarrow -1.0 + 0.1 [-1 + 0.9 \times \max(Q^(R,\text{Left}), Q^(R,\text{Right})) - (-1.0)]$
   $Q^(M, \text{Right}) \leftarrow -1.0 + 0.1 [-1 + 0.9 \times \max(-1.0, -1.0) - (-1.0)]$
   $Q^(M, \text{Right}) \leftarrow -1.0 + 0.1 [-1 - 0.9 + 1.0]$
   $Q^(M, \text{Right}) \leftarrow -1.0 + 0.1 [-0.9]$
   $Q^*(M, \text{Right}) \leftarrow -1.0 - 0.09 = -1.09$

Después de muchas iteraciones, $Q^(M, \text{Right})$ convergerá a $Q^(M, \text{Right}) = Q^{\pi}(M, \text{Right}) = -1.9$, el valor óptimo.

Diferencia clave: En Q-learning, usamos $\max_a Q^(s_{t+1}, a)$ para la actualización, no la acción que realmente tomaremos según μ. Esto permite que Q-learning aprenda los valores Q óptimos $Q^$ (que corresponden a la política óptima π) independientemente de qué política estamos siguiendo.

PROF

8. El Desafío Off-Policy: ¿Cómo aprender π mientras seguimos μ?

Problema: Si seguimos μ (explorando), ¿cómo podemos estimar los valores de π?

Este es el desafío central del aprendizaje off-policy: necesitamos una manera de "corregir" las experiencias obtenidas con μ para estimar los valores bajo π.

Solución 1: Q-learning

El Q-learning es un método off-policy que aprende directamente los valores Q^* óptimos:

$Q^_{t+1}(s_t, a_t) \leftarrow Q^t(s_t, a_t) + \alpha [r{t+1} + \gamma \max_a Q^t(s{t+1}, a) - Q^_t(s_t, a_t)]$

Con Q-learning:

- Exploramos usando μ
- Actualizamos Q hacia los valores óptimos $Q^$ usando $\max_a Q^$
- No necesitamos conocer explícitamente π

Solución 2: Importance Sampling

Si queremos estimar directamente V^{π} mientras seguimos μ:

1. **Concepto**: Reponderar las experiencias según la relación entre π y μ
2. **Idea básica**: Las trayectorias que son más probables bajo π que bajo μ deben tener más peso

El peso de importance sampling para una trayectoria $\tau = (s_0, a_0, s_1, a_1, ..., s_T)$ es:

$\rho_{\tau} = \prod_{t=0}^{T-1} \frac{\pi(a_t|s_t)}{\mu(a_t|s_t)}$

Por ejemplo, para la trayectoria M → R → T:

PROF

- Probabilidad bajo π: $\pi(\text{Right}|M) \times \pi(\text{Right}|R) = 1.0 \times 1.0 = 1.0$
- Probabilidad bajo μ: $\mu(\text{Right}|M) \times \mu(\text{Right}|R) = 0.8 \times 0.8 = 0.64$
- Ratio = $\frac{1.0}{0.64} = 1.5625$

Este ratio nos permite ajustar los retornos observados siguiendo μ para estimar los retornos esperados bajo π.

Visualizacion

9. Intuición: ¿Cuándo Usar On-Policy vs Off-Policy?

Entender cuándo usar cada enfoque es crucial para aplicar RL de manera efectiva:

Ventajas del Aprendizaje On-Policy (SARSA)

1. **Más seguro para aprender en entornos reales**:

- Como aprende sobre lo que realmente hace, considera los riesgos de exploración
- Ejemplo: Un robot que aprende a caminar evitará acciones que puedan provocar caídas frecuentes

2. **Más estable durante el entrenamiento**:
    - Menor varianza en las actualizaciones
    - Convergencia más suave pero potencialmente más lenta
3. **Mejor cuando no existe una política óptima clara**:
    - En algunos problemas con aleatoriedad o adversarios, no hay una política "perfecta"
    - On-policy aprende a optimizar la política que realmente estamos usando
4. **Ideal para problemas donde exploración y explotación están balanceadas**:
    - En nuestro ejemplo del corredor: SARSA aprendería que, bajo μ, $Q^{\mu}(M, \text{Right}) \approx -2.1$, reconociendo que a veces tomaremos Left

Ventajas del Aprendizaje Off-Policy (Q-learning)

1. **Mayor eficiencia en el uso de datos**:
    - Puede aprender de cualquier experiencia, incluso de datos recolectados por políticas antiguas
    - Permite reutilizar experiencias pasadas (experience replay)
2. **Puede aprender la política óptima mientras explora**:
    - Aprende Q^* directamente, sin importar qué política estamos siguiendo
    - En nuestro ejemplo: Q-learning aprenderá que $Q^*(M, \text{Right}) = Q^{\pi}(M, \text{Right}) = -1.9$ (el valor óptimo), ignorando las exploraciones de μ
3. **Permite separar exploración de aprendizaje**:
    - Podemos usar una política muy exploratoria sin comprometer el aprendizaje
    - Útil cuando la exploración puede ser costosa o peligrosa
4. **Necesario para aprendizaje desde observación**:
    - Permite aprender de demostraciones humanas o de otros agentes
    - No necesitamos seguir la misma política que queremos aprender

PROF

Consideraciones Prácticas

- **Complejidad computacional**: Off-policy suele requerir más cómputo
- **Estabilidad**: On-policy es generalmente más estable, off-policy puede diverger
- **Tipo de problema**:
    - Problemas seguros, donde la exploración no es costosa → Off-policy
    - Problemas de seguridad crítica → On-policy
    - Problemas donde los datos son escasos → Off-policy (reutiliza datos)

En nuestro ejemplo del corredor, la diferencia de rendimiento no es dramática ($V^{\pi}(M) = -1.9$ vs $V^{\mu}(M) \approx -2.3$), pero en problemas más complejos, elegir el enfoque correcto puede marcar una gran diferencia en rendimiento, estabilidad y seguridad.

---

10. Visualización Comparativa de Funciones de Valor

```
Valores de Estado V:
[L] --- [M] --- [R] --- [T]
-2.71  -1.9   -1.0    0     (V^π - Política Óptima)
-3.0   -2.3   -1.2    0     (V^μ - Política Exploratoria)
```

```
Valores Q para la política π:
Estado | Q^π(s,Left) | Q^π(s,Right)
-------|-------------|-------------
  L    |    -2.71    |    -2.71
  M    |    -2.71    |    -1.9
  R    |    -2.71    |    -1.0
  T    |      0      |      0
```

```
Valores Q para la política μ:
Estado | Q^μ(s,Left) | Q^μ(s,Right)
-------|-------------|-------------
  L    |    -3.0     |    -2.8
  M    |    -3.0     |    -2.1
  R    |    -2.5     |    -1.2
  T    |      0      |      0
```

PROF

11. Métodos Típicos

Tipo	Método	Característica	Actualización
On-Policy	SARSA	Aprende Q^{μ} para la política actual	$Q^{\mu}(s_t,a_t) \leftarrow Q^{\mu}(s_t,a_t) + \alpha[r_{t+1} + \gamma Q^{\mu}(s_{t+1},a_{t+1}) - Q^{\mu}(s_t,a_t)]$
	Actor-Critic	Actualiza política y función de valor en paralelo	Actualiza V^{μ} y μ simultáneamente
Off-Policy	Q-learning	Aprende valores Q^* óptimos	$Q^*(s_t,a_t) \leftarrow Q^*(s_t,a_t) + \alpha[r_{t+1} + \gamma \max_a Q^*(s_{t+1},a) - Q^*(s_t,a_t)]$

Tipo	Método	Característica	Actualización
	DQN	Versión de Q-learning con redes neuronales	Aprende Q^* mientras sigue μ (ε-greedy)

12. Ventajas y Desventajas

Aspecto	On-Policy	Off-Policy
Exploración	La política μ debe balancear exploración y explotación	Puede usar una política μ altamente exploratoria mientras se aprende π
Estabilidad	Más estable, menor varianza	Puede tener alta varianza (especialmente con importance sampling)
Eficiencia de datos	Menos eficiente (μ cambia, los datos viejos no aplican)	Más eficiente (puede reutilizar datos aunque π cambie)
Complejidad	Generalmente más simple	Suele ser más complejo debido a las correcciones necesarias

13. Conclusión

La distinción entre on-policy y off-policy es fundamental en RL:

- **On-policy**: "aprendo sobre lo que hago" (μ → V^{μ}, Q^{μ})
- **Off-policy**: "hago una cosa, aprendo sobre otra" (μ → V^{π}, Q^{π} o Q^*)

En nuestro ejemplo del corredor:

- On-policy: Seguimos μ y aprendemos V^{μ} y Q^{μ}
- Off-policy: Seguimos μ (exploramos) pero aprendemos V^{π}, Q^{π} o directamente Q^* (valores óptimos)

PROF

Esta distinción es crucial para entender algoritmos modernos de RL y el balance entre exploración y explotación.