

08MIAR-Aprendizaje por refuerzo

Sesión 4 – Algoritmos base: Policy Gradient

Curso Abril 23/24



Universidad
Internacional
de Valencia

De:



Planeta Formación y Universidades

Índice

Definición Policy Gradient

Deep Policy Gradient

Proceso de aprendizaje

Algoritmo: REINFORCE

Algoritmo: Vanilla Policy Gradient

Conclusiones

Bibliografía recomendada

Índice

Definición Policy Gradient

Deep Policy Gradient

Proceso de aprendizaje

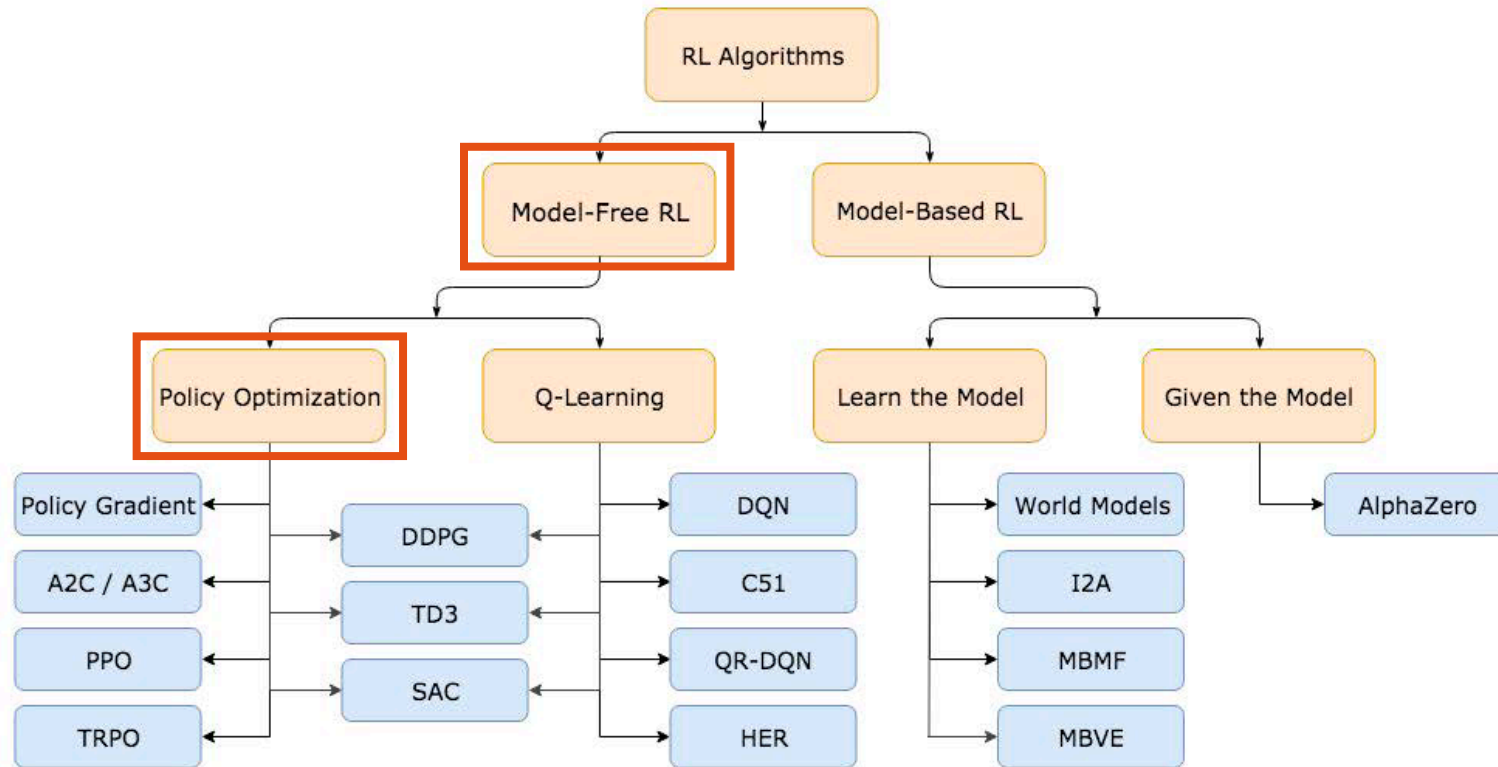
Algoritmo: REINFORCE

Algoritmo: Vanilla Policy Gradient

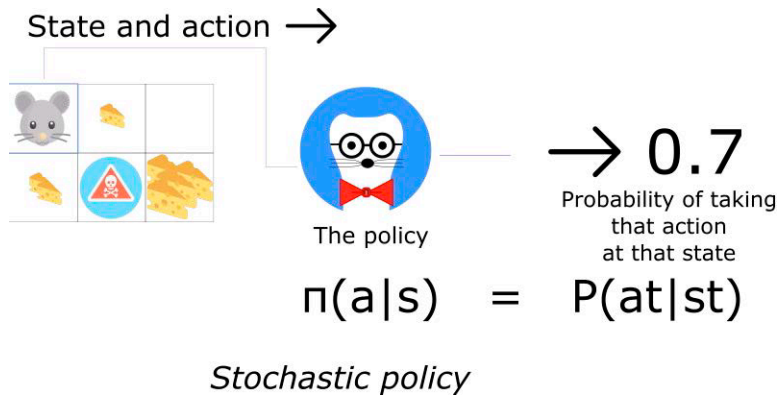
Conclusiones

Bibliografía recomendada

¿Dónde estamos?



Motivación



https://cdn-images-1.medium.com/max/1600/1*YCABimP7x1wZZZKqz2CoyQ.png

- **Policy, π .** Estrategia que sigue nuestro agente. Probabilidad de realizar una acción en un estado.

$$p(a|s) = \pi(s): \pi(s) \in [0, 1]$$

- **Q-learning.** Se sigue una **política voraz** respecto a la función Q, la cual debe estimarse previamente.

$$\pi(s) = \operatorname{argmax}_a (q^\pi(s, a))$$

- **Policy gradient (PG).** Tiene como objetivo modelar directamente una **distribución de probabilidad policy**, $\pi(s)$.

Motivación

PG vs. Q-learning

- PG permite obtener directamente **policies estocásticas** - e.g., $\pi(s) = [0, 0.2, 0.6, 0.2]$ -, mientras que Q-learning ofrece únicamente policies deterministas - e.g., $\pi(s) = [0, 0, 1, 0]$.
- PG permite modelar un espacio con **acciones continuas**. Funciona especialmente bien en espacios de acciones muy grandes.
- Q-learning busca obtener la mayor estimación de la función Q, para mejora de forma indirecta la policy. **PG actúa directamente sobre la policy**, lo cual mejora la **eficiencia y convergencia**.

Índice

Definición Policy Gradient

Deep Policy Gradient

Proceso de aprendizaje

Algoritmo: REINFORCE

Algoritmo: Vanilla Policy Gradient

Conclusiones

Bibliografía recomendada

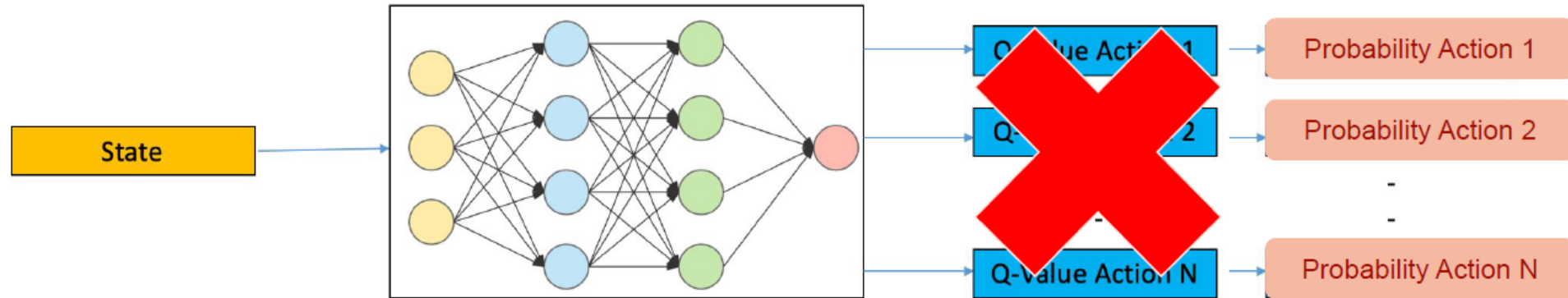
Deep Policy Gradient (DPG)

- Asunciones en **DQN**: mejorando la estimación de la q , *indirectamente* mejoraremos la policy. Muchas ocasiones (espacio de acciones muy grande) dicha convergencia no sucede.
- DPG: optimizamos la *policy* – distribución de probabilidad de acciones - directamente, $\pi(s)$.
- Uso de una **DNN**, $f_{\theta}(\cdot)$, como función **aproximadora** de $\pi^*(s)$.

$$\pi^*(s) = (f_{\theta}(s))$$

- Debido a que la mayoría de simulaciones trabajan con la pantalla directamente, el tipo de red neuronal que usaremos serán **redes convolucionales**.

Conceptos importantes



Deep Policy Gradient

Índice

Definición Policy Gradient

Deep Policy Gradient

Proceso de aprendizaje

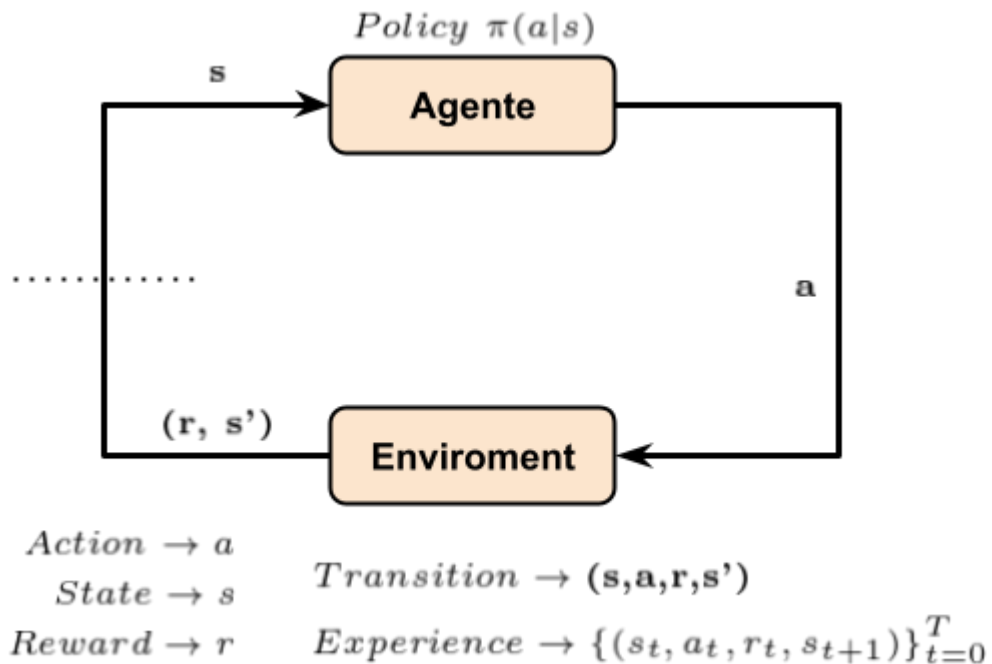
Algoritmo: REINFORCE

Algoritmo: Vanilla Policy Gradient

Conclusiones

Bibliografía recomendada

Trayectorias entorno-agente en PG



- En **PG**, trabajamos **on-policy**. Es decir, la memoria únicamente contiene **transiciones de una policy dada**. En este caso, se le llama **Trayectoria (T)**.
- Se define una **policy**, $\pi(a|s)$, parametrizada por una DNN, que toma como entrada el estado pre-procesado, y devuelve una **distribución de probabilidad sobre las acciones, con activación softmax**. Por ejemplo $\pi(a|s^*) = [0.2, 0.2, 0.6, 0]$.

$$\pi(a|s) = f_{\theta}(s)$$

- Seleccionamos la acción **a a partir de un muestreo de $\pi(a|s)$** , lo cual permite la **exploración**.

$$a = \text{sample}(\pi(s))$$

Aprendizaje supervisado

Una función de optimización típica de otra rama de *deep learning*, el **aprendizaje supervisado**, es la **entropía cruzada**, utilizando las probabilidades predichas, y un *one-hot-encoding* de la distribución de probabilidad predicha para el conjunto de etiquetas futuras.

$$L = - \sum y_i \log (\hat{y}_i)$$

En **reinforcement learning**, no existe un ground truth para una acción dada, pero nos sirve para establecer ciertas **analogías**.

$$L = - \sum y_a \log (\pi(s)_a)$$

Policy gradients – Naive solution

Se define un entorno con una **policy**, $\pi_{\theta}(a|s)$, que produce una distribución de probabilidad sobre el conjunto de acciones en una interacción agente-entorno. Esta interacción se realiza durante una trayectoria, de duración T , y en cada iteración, t_i , se produce una transición $(s_{t_i}, a_{t_i}, r_{t_i}, s_{t_i+1})$.

Se quiere optimizar la policy para acercarla en la mayor medida a una solución óptima. Policy Gradients (PG) se centra en actualizar π_{θ} en dirección del gradiente de la misma. En el caso ideal, **si la acción seleccionada es la óptima, se quiere que su probabilidad sea lo más cercana a 1**, lo cual se conseguiría por descenso de gradiente.

$$\theta_{t_i+1} = \theta_{t_i} + \alpha \nabla \pi_{\theta_{t_i}}(a_{t_i}^* | s_{t_i}) \quad (i)$$

Sin embargo, **la acción óptima no es conocida a priori**. Inicializando en una **policy aleatoria**, si optimizamos sobre una acción genérica (ii), estamos **optimizando soluciones sub-óptimas**.

$$\theta_{t_i+1} = \theta_{t_i} + \alpha \nabla \pi_{\theta_{t_i}}(a_{t_i} | s_{t_i}) \quad (ii)$$

Policy gradients – Ponderando acciones

Una solución para **ponderar los gradientes** es dar más importancia a aquellas acciones que den una mejor policy, es utilizar la información que proporciona el entorno como *feedback*: las **recompensas esperadas a futuro** dada la trayectoria realizada. Para ello, retomamos el concepto de función **quality** $q_{\pi}(s, a)$:

$$q_{\pi}(s, a) = E_{\pi}[G_t | (s, a)]$$

Por lo tanto, la actualización de la policy en un step dado quedaría:

$$\theta_{t_i+1} = \theta_{t_i} + \alpha q_{\pi}(s_{t_i}, a_{t_i}) \nabla \pi_{\theta_{t_i}}(a_{t_i} | s_{t_i}) \quad (iii)$$

On-policy correction: actualizamos la policy en base a las acciones que toma la propia policy en una trayectoria. **Se debe compensar que acciones más probables serán llevadas a cabo más a menudo.** Por ello, se obtiene el *ratio* entre la actualización de la policy y la probabilidad de la acción tomada.

$$\theta_{t_i+1} = \theta_{t_i} + \alpha \frac{q_{\pi}(s_{t_i}, a_{t_i}) \nabla \pi_{\theta_{t_i}}(a_{t_i} | s_{t_i})}{\pi_{\theta_{t_i}}(a_{t_i} | s_{t_i})} \quad (iv)$$

¿Qué hay del logaritmo?

Aprendizaje supervisado: $L = - \sum y_i \log (\hat{y}_i)$

RL, *Policy Gradient*: Podemos realizar el *log-derivative trick*, para obtener una expresión logarítmica de la actualización de los pesos.

$$\nabla \ln f(x) = \frac{\nabla f(x)}{f(x)} \rightarrow \frac{q_{\pi}(s_{t_i}, a_{t_i}) \nabla \pi_{\theta_{t_i}}(a_{t_i} | s_{t_i})}{\pi_{\theta_{t_i}}(a_{t_i} | s_{t_i})} = q_{\pi}(s_{t_i}, a_{t_i}) \nabla_{\theta_{t_i}} \log \pi_{\theta_{t_i}}(a_{t_i} | s_{t_i})$$

Por tanto, el criterio de optimización de ***vanilla policy gradient*** consiste en:

$$\theta_{t_{i+1}} = \theta_{t_i} + \alpha q_{\pi}(s_{t_i}, a_{t_i}) \nabla_{\theta_{t_i}} \log \pi_{\theta_{t_i}}(a_{t_i} | s_{t_i}) \quad (v)$$

Suavizando recompensas

$$q_{\pi}(s, a) = [1.0, 1.1, 1.0, 1.0]$$

$$q_{\pi}(s, a) = [1.0, 0.0, 0.0, 0.0]$$

¿A qué acción se le debe dar mayor importancia al actualizar?

- Para suavizar el problema con el uso de la recompensa como factor, se proponen reemplazos de la función de calidad.

Policy gradient methods maximize the expected total reward by repeatedly estimating the gradient $g := \nabla_{\theta} \mathbb{E} [\sum_{t=0}^{\infty} r_t]$. There are several different related expressions for the policy gradient, which have the form

$$g = \mathbb{E} \left[\sum_{t=0}^{\infty} \Psi_t \nabla_{\theta} \log \pi_{\theta}(a_t | s_t) \right], \quad (1)$$

where Ψ_t may be one of the following:

- | | |
|--|---|
| 1. $\sum_{t=0}^{\infty} r_t$: total reward of the trajectory. | 4. $Q^{\pi}(s_t, a_t)$: state-action value function. |
| 2. $\sum_{t'=t}^{\infty} r_{t'}$: reward following action a_t . | 5. $A^{\pi}(s_t, a_t)$: advantage function. |
| 3. $\sum_{t'=t}^{\infty} r_{t'} - b(s_t)$: baselined version of previous formula. | 6. $r_t + V^{\pi}(s_{t+1}) - V^{\pi}(s_t)$: TD residual. |

The latter formulas use the definitions

$$V^{\pi}(s_t) := \mathbb{E}_{\substack{s_{t+1:\infty}, \\ a_{t+1:\infty}}} \left[\sum_{l=0}^{\infty} r_{t+l} \right] \quad Q^{\pi}(s_t, a_t) := \mathbb{E}_{\substack{s_{t+1:\infty}, \\ a_{t+1:\infty}}} \left[\sum_{l=0}^{\infty} r_{t+l} \right] \quad (2)$$

$$A^{\pi}(s_t, a_t) := Q^{\pi}(s_t, a_t) - V^{\pi}(s_t), \quad (\text{Advantage function}). \quad (3)$$

Estrategias de entrenamiento

Durante el proceso de aprendizaje debemos tener algunos conceptos y situaciones presentes, para entender qué está ocurriendo:

- Una de las primeras decisiones que debemos tomar es “¿**Cuántos *steps* vamos a usar para ir modificando la estrategia?**”. El proceso de aprendizaje se puede ver más o menos impactado dependiendo del número de iteraciones que realicemos para ir almacenando nuestra experiencia.
- Además, al trabajar con la trayectoria, **las recompensas obtenidas se procesarán siguiendo un enfoque conocido como *discounted rewards***. Al tener una trayectoria finita, utilizaremos las recompensas en sentido inverso para ir estimando la recompensa esperada a futuro (G) en los siguientes estados y de esta forma poder ponderar las acciones de manera adecuada.
- Por otro lado, **usar la recompensa como factor de las probabilidades de las acciones produce una varianza en los datos muy grande**. Tened en cuenta que con esta definición la probabilidad de una acción en un estado puede cambiar dependiendo de si la recompensa cambia también. Esto **dificulta el aprendizaje** ya que **no se encuentra una correlación entre estado y probabilidad de acción fácilmente**. Esta situación se puede dar en muchos escenarios, sobre todo en las simulaciones basadas en videojuegos.

Índice

Definición Policy Gradient

Deep Policy Gradient

Proceso de aprendizaje

Algoritmo: REINFORCE

Algoritmo: Vanilla Policy Gradient

Conclusiones

Bibliografía recomendada

REINFORCE

function REINFORCE

 Initialise θ arbitrarily

for each episode $\{s_1, a_1, r_2, \dots, s_{T-1}, a_{T-1}, r_T\} \sim \pi_\theta$ **do**

for $t = 1$ to $T - 1$ **do**

$\theta \leftarrow \theta + \alpha \nabla_\theta \log \pi_\theta(s_t, a_t) v_t$

end for

end for

return θ

end function

Índice

Definición Policy Gradient

Deep Policy Gradient

Proceso de aprendizaje

Algoritmo: REINFORCE

Algoritmo: Vanilla Policy Gradient

Conclusiones

Bibliografía recomendada

Vanilla Policy Gradient

Algorithm 1 “Vanilla” policy gradient algorithm

Initialize policy parameter θ , baseline b

for iteration=1, 2, ... **do**

Collect a set of trajectories by executing the current policy

At each timestep in each trajectory, compute

the *return* $R_t = \sum_{t'=t}^{T-1} \gamma^{t'-t} r_{t'}$, and

the *advantage estimate* $\hat{A}_t = R_t - b(s_t)$.

Baseline, reducir la varianza!

Re-fit the baseline, by minimizing $\|b(s_t) - R_t\|^2$, summed over all trajectories and timesteps.

Update the policy, using a policy gradient estimate \hat{g} , which is a sum of terms $\nabla_{\theta} \log \pi(a_t | s_t, \theta) \hat{A}_t$

end for

Índice

Definición Policy Gradient

Deep Policy Gradient

Proceso de aprendizaje

Algoritmo: REINFORCE

Algoritmo: Vanilla Policy Gradient

Conclusiones

Bibliografía recomendada

Conclusiones

- **Policy Gradient** es uno de los algoritmos base dentro del aprendizaje por refuerzo. Este algoritmo es el **origen de algunos de los algoritmos más potentes actualmente**. Es un algoritmo de tipo **on-policy**.
- En comparación con DQN, algunas de las características a destacar son una mayor capacidad de **convergencia**, apto para trabajar con **espacios de acciones grandes (y continuos)** y posibilidad de aprender **policies estocásticas**.
- La **función de coste** en Policy Gradient se centra en ir optimizando la propia policy aplicando ***gradient ascent*** sobre la **probabilidad de la acción seleccionada y su recompensa obtenida**.
- Al ser un algoritmo base, veremos en las siguientes sesiones de la asignatura las evoluciones que se han ido produciendo para estabilizar y mejorar el proceso de aprendizaje del agente.

Índice

Definición Policy Gradient

Deep Policy Gradient

Proceso de aprendizaje

Algoritmo: REINFORCE

Algoritmo: Vanilla Policy Gradient

Conclusiones

Bibliografía recomendada

Bibliografía recomendada

- “Reinforcement Learning: An Introduction”, Sutton y Barto:

<http://incompleteideas.net/book/bookdraft2017nov5.pdf>

(Capítulo 13, Policy Gradient Methods)

- An Intuitive explanation on Policy Gradients, Adrien Lucas, Towards data science / Medium

<https://towardsdatascience.com/an-intuitive-explanation-of-policy-gradient-part-1-reinforce-aa4392cbfd3c>



viu

Universidad
Internacional
de Valencia