- ¿Cuál es la función de coste o la función ppal en la que se basa Actor Critic?

Se comenta en el tema de policy gradients que la función utilizada (para todos los algoritmos dentro de este grupo) es Cross-Entropy.

- De las 2 funciones de coste que tenía PPO. ¿cuál hemos estudiado?¿por qué es importante el clip?

La función estudiada en clase es la Clipped surrogate objective, para que la policy se actualice con suavidad

Es importante el clip, para que la actualización de las policies esté dentro de unos rangos controlados, de manera que la actualización sea más “suave”. Si sobrepasase estos rangos, se tomarían los valores límites para llevar a cabo la actualización. Esta estrategia nos asegura que no actualizamos las policies en exceso, evitando afectar negativamente a la convergencia del proceso de optimización.

- ¿Por qué CLIPPED?

PPO comparativa de distribución de probabilidades de la versión anterior de la policy y la actual. Para que la actualización de la POLICY sea suave, si el cambio es muy brusco eclipsaría al resto de acciones.

- ¿Qué es Q?

Q - estimación de la recompensa a futuro a partir del estado y la acción tomada

- ¿Qué es V?

V - recompensa a futuro, solo dependiendo del estado

- ¿Hasta qué futuro hay que calcular las recompensas en DQN?

Se calculan hasta el final del episodio o intervalo

- ¿Por qué tenemos T en policy gradient?

T es necesario porque en on-policy necesitamos una trayectoria finita

- ¿Cuándo para Bellmann?

Bellman para al final del episodio (recordad el condicional, si hay más estados o no)

- ¿Qué es la función de ventaja y qué mide?

La función de ventaja es A=Q(s,a)-V(s) y mide qué ganamos por elegir la acción 'a'

- ¿Qué caracteriza DQN en cuanto a la actualización de los pesos?

DQN actualiza los pesos a partir de TODA la memoria de transiciones, porque es OFF-POLICY

- ¿Qué introducimos para la toma de decisiones a partir de A2C/A3C?

A2C y A3C surgen de seguir evolucionando el algoritmo de Actor-Critic, enfocados en el factor de relevancia de la acción seleccionada, y utilizando en lugar del Value la función de Ventaja.

- ¿Cuántas arquitecturas tiene DDPG?

DDPG tiene 4 arquitecturas, 2 actores y 2 critics, que son un AC para el target y otro AC para el modelo.

- ¿Qué algoritmos para entornos discretos/continuos?

- Algoritmos Off-policy solo discreto (DQN)
- Algoritmos On-policy: Discreto y Continuo (PG, AC, A2C/A3C, PPO)
- DDPG (algoritmo mixto): solo continuo

- ¿Qué heurística usamos en model based?

Las opciones más utilizadas son la búsqueda en árbol o algoritmos basados en población (genéticos). Todos ellos basados a su vez en simulaciones de Montecarlo para poder analizar los distintos caminos disponibles. Pej: Monte Carlo Tree Search (MCTS)

- Característica que definía los problemas de model based

Que conocemos el modelo del entorno y la dinámica del estado. Podemos crear un tree search a partir del estado actual porque sabemos cómo funciona el entorno.

- ¿Qué caracteriza el aprendizaje del model based?

El aprendizaje model based se caracteriza por el conocimiento del modelo, lo que nos suele permitir conocer las acciones que nos llevan de un estado a otro, hacia adelante, hacia atrás…

- ¿Por qué necesitamos una trayectoria finita de experiencia de transición?

Porque estamos en on policy, es una serie de steps, es donde vamos acumulando toda la experiencia de casa step, todas las transiciones para luego llevar a cabo esa optimización

- ¿Qué son las discounted rewards, y cómo funcionan?

Es el valor esperado de la suma de las recompensas a futuro. Se calcula teniendo en cuenta las recompensas de la trayectoria almacenada. Cada recompensa está afectada por el discount factor para estimar la importancia de los valores futuros.

Es posible calcularlo porque trabajamos con un conjunto finito de elementos en la trayectoria, y por lo tanto, somos capaces de calcular las recompensas esperadas en cada instante de tiempo.

- ¿Qué es el discount factor (utilizado p.ej. en DQN), y cómo funciona?

Este factor define la importancia que le damos a las recompensas futuras en nuestra estimación.

A veces una recompensa futura no está del todo relacionada con la acción actual. Por ejemplo, salir de fiesta un domingo 3 meses antes de un examen difícilmente afecta a los resultados. Tomar la decisión de si salir o no en base al examen no tiene sentido.

Se utiliza para penalizar las recompensas a futuros retornos. Nos importa más lo que pase ahora o en el momento más cercano que en el futuro más lejano. Normalmente se utiliza gamma, como valor cercano a uno (por ejemplo, 0.99) para penalizar más cuanto más te alejes del estado actual. (ya que 0.99^n < 0.99 con n natural no negativo.

- **¿Qué diferencia hay entre un algoritmo ON-POLICY y un algoritmo OFF-POLICY?**

Los algoritmos ON-POLICY hacen uso de los datos recolectados solo con la última versión de la policy aprendida hasta el momento, para luego actualizarla con la nueva información. Una vez la policy es actualizada, la información se desecha para comenzar una nueva trayectoria.

En cambio, los algoritmos OFF-POLICY mantienen una memoria con experiencia recolectada con distintas versiones de la policy del agente durante toda la interacción con el entorno. Durante el proceso de entrenamiento, en el momento de actualizar la policy, escogen un conjunto aleatorio de transiciones para llevar a cabo la actualización

- TradeOff Policy Optimization vs Q-Learning:

La principal fortaleza de los métodos de Policy Optimization es que se basan en principios, en el sentido de que se optimiza directamente para lo que desea. Esto tiende a hacerlos estables y fiables, además de tener mayor capacidad de convergencia.

Por el contrario, los métodos de Q-learning solo optimizan indirectamente el rendimiento del agente, entrenando Q para satisfacer una ecuación de autoconsistencia. Hay muchos modos de falla para este tipo de aprendizaje, por lo que tiende a ser menos estable. Sin embargo, los métodos de Q-learning obtienen la ventaja de ser sustancialmente más eficientes en cuanto a muestras cuando funcionan, porque pueden reutilizar los datos de manera más efectiva que las técnicas de optimización de políticas.

Policy Optimization y Q-Learning no son incompatibles, y existe una variedad de algoritmos que viven entre los dos extremos. (DDPG, SAC, …)

- Principal aportación de Deep Q-Networks:

Deep Q-Networks es un algoritmo de RL basado en Q-learning, su principal aportación es la combinación de este algoritmo con técnicas de deep learning.

Aprendizaje basado en la ecuación de Bellman

Necesario introducir para asegurar la convergencia una red target con pesos congelados para comparar en la función de costes, y secuencia de frames para mantener contexto de la tendencia de la secuencia.

- Principal característica de Policy Gradients:

Trabajamos con un conjunto finito de elementos en la trayectoria.

Aprendizaje basado en la función de Cross Entropy.

Optimizamos directamente la estrategia para maximizar la recompensa esperada, de ahí que se busca el gradiente de la policy para maximizar ese valor. Después de actualizar los pesos del modelo, la experiencia se resetea.

Más preparado para un espacio de acciones muy grande o continuo. Mejores propiedades desde el punto de vista de la convergencia.

Exploración mediante distribución aleatoria de probabilidad.

Función de coste surge de aplicar “Log Derivative Trick” (ratio entre la actualización de la policy y la probabilidad de la acción tomada) para facilitar los cálculos y no impactar negativamente al proceso de optimización cuando

una acción tiene probabilidad alta, se centra en ir optimizando la propia policy aplicando gradient ascent sobre la probabilidad de la acción seleccionada y su recompensa obtenida.

- Principal característica de AC:

Como punto de mejora de PG, introducimos el Value como factor de relevancia (como de bueno es el estado, sin necesidad de tomar una acción) El modelo del agente se divide en dos componentes, el Actor (devuelve la distribución de probabilidades asociada a las acciones) y el Critic (estima el Value en el estado en el que se encuentre).

Al hacer sampling de los datos, necesitamos estimar también el Value en cada estado almacenado.

Función de coste para actor (similar a PG, pero en vez de la recompensa utilizamos el Value estimado), y para el Critic (error cometido entre el Value recolectado y el Value estimado)

- Principal característica de A2C/A3C:

Algoritmo multiproceso.

A2C (actualización modelo síncrona), esperar a que todos acaben para actualizar-

AC3 (actualización modelo asíncrona), se “pisan” los modelos, permite ejecutarse en CPU.

Reemplazo del Value como factor de las acciones tomadas por el agente por la función de ventaja o funciones derivadas como la GAE (Generalized Advantage estimation)

- Principal característica de PPO y TRPO:

Tanto TRPO como PPO son algoritmos on policy, optimizando la policy del agente con la trayectoria que se ha obtenido a partir de la policy más actual. En este sentido, los métodos que pertenecen a la familia de trust region methods, abordan la actualización de la policy diferenciando entre la policy anterior y la policy nueva, calculando el ratio entre ambas, asegurando la convergencia.

Permiten multiproceso para tareas de recolección de datos.

En TRPO se utiliza como función de coste la matriz hessiana, lo cual por su carga computacional hace que el algoritmo no sea muy aplicable.

En cambio, PPO utiliza KL-divergence, o Clipped Surrogate objective, la cual ayuda a asegurar la convergencia y a que la actualización de la policy sea suave, manteniendo la actualización de la policy dentro de unos rangos controlados.

Exploración mediante distribución de probabilidad aleatoria ponderada, al igual que en AC.

- Principal característica de DDPG:

Es un algoritmo híbrido, ya que combina características de la familia Deep Q-Networks (funcion Q), con elementos de la familia Policy Gradients (policy), siguiendo un enfoque Actor-Critic:

- Actor: modelo basado en Policy Gradients capaz de ofrecer como salida valores en una escala continua
- Critic: modelo basado en Deep Q-networks para estimar los valores de recompensa esperados

Consta de 4 redes neuronales diferentes: Dos redes para predicción (actor y critic networks) y dos redes target (actor y critic target networks).

Permite también el multiproceso.

“Es como utilizar Q-Learning en espacios de acciones continuos.”

El proceso de exploración se lleva a cabo incluyendo una función de ruido (Ornstein-Uhlenbeck) en el momento de la selección de la acción, haciendo que las variaciones están marcadas por el estado anterior, y que sea suave.

Preguntas del Test del Manual interactivo:

1. **En inteligencia artificial, el aprendizaje por refuerzo pertenece al campo de:**
   Aprendizaje automático.

2. **Durante el proceso de aprendizaje, la interacción se produce entre:**
   El entorno y el agente

3. **Cada vez que el agente ejecuta una acción, el entorno responde con (puede haber varias respuestas correctas):**
   Una recompensa. Una nueva observación.

4. **El objetivo de la estrategia del agente es:**
   Llegar a la estrategia óptima que maximice la recompensa.

5. **Cuando el agente termina el proceso de exploración, comienza el proceso de:**
   Explotación.

6. **Indica qué afirmaciones son verdaderas.**
   On-policy y Off-policy son enfoques dentro de la clasificación basada en estrategia.

7. **Cuando trabajamos con un enfoque model-based, el modelo se refiere a:**
   El modelo que define las dinámicas del entorno y que el agente utiliza durante el proceso de aprendizaje.

8. **Respecto al algoritmo de Deep Q-Networks (puede haber varias respuestas correctas):**
   Sigue una estrategia Off-policy
   y su función de coste está basada en la ecuación de Bellman.

9. **En el algoritmo de Policy Gradients, el factor de relevancia que se utiliza en la función de coste es:**
   La recompensa.

10. **En el algoritmo de Actor-Critic (puede haber varias respuestas correctas):**
    El Actor se encarga de estimar la distribución de probabilidades de las acciones. y El Critic estima el Value del estado actual.

11. **En el algoritmo de A2C (puede haber varias respuestas correctas):**
    Es un algoritmo on-policy. Es un algoritmo multi-proceso. En la función de coste, usa como factor de relevancia la función de ventaja.

12. **PPO:**
    Es un algoritmo multi-proceso. Respecto a la función de coste, hay dos versiones: Clip y KL-divergence. Es una evolución del algoritmo de TRPO.

13. **Sobre el algoritmo de DDPG:**
    Al ser un algoritmo híbrido, se combinan arquitecturas de modelos basadas en DQN y Policy Gradients.

14. **En su definición base, una Transición está compuesta de:**
    Estado, Acción, Recompensa, Siguiente Estado.

15. **La arquitectura de Deep Learning más utilizada hoy en día en algoritmos de aprendizaje por refuerzo es:**
    Arquitectura Convolucional.

Preguntas Javier:

Aprovechar sobre el esquema y apuntar las características principales de cada algoritmo.

Conceptos de las fórmulas, a que algoritmos pertenecen y diferencias principales.

Función de coste de DQN: Ecuación de Bellman

Función de coste de Actor-Critic: Basada en el value, similar al Cat Cross Entropy

- Recompensa: Feedback básico que nos da el entorno, en cada instante de tiempo acorde a cada acción del agente el entorno nos devuelve una recompensa. Lo más bruto que podemos tener es la suma de recompensas en cada instante de tiempo

- Función Q: Función que no es solo de los DQN, es la recompensa esperada a futuro para un estado tomando la acción “a” que el agente ha decidido en ese estado. Es decir, a partir de ese estado y esa acción hazme esa estimación (esperanza) a futuro de cuál es esa recompensa. Ese a futuro en la Q es en base a un estado y una acción y en el Value es solo en el estado.

Si estamos en DQN la recompensa esperada a futuro, tenemos Bellman y es recurrente, vamos utilizando Bellman hasta el final del episodio.

En la familia de Policy gradient, tenemos la T mayúscula, periodo de tiempo finito

Por qué teníamos la T: (Porque tenemos que calcular el discount reward). Porque necesitamos una trayectoria finita de experiencia, porque todo es con la versión (policy) actual y para actualizar el modelo necesitamos parar la ejecución con toda la experiencia hacer la optimización y desechar la experiencia. Trayectoria finita

- Discount reward: Para ese trozo de ejecución desde el principio hasta la T tenemos todo lo que ha ocurrido. (El discount reward “es como una función Q”). El discount reward podría ser como la función de Bellman pero en finito y conociendo los elementos de la ecuación de Bellman
- Se podría decir que el value es la media ponderada de los Q-values (informal)

Realmente Q y V es un concepto "ideal" y a través de R es como podemos estimar Q y V.

- ¿Cuándo para Bellman?: Cuando se llegar al final del episodio se devuelve como Bellman la recompensa para ese estado y ya no hay estado siguiente.

- Función de ventaja: Q-V Nos ayuda a elegir una acción en concreto ya que nos dice lo que ganamos con esa acción. Es como que el Value nos marca un Umbral si la acción está por arriba es prometedora, haz esa, si está por abajo deséchala.

- El minibatch en DQN se utiliza en la actualización de los pesos del agente. La del target se lleva a cabo cada x steps y es llevar los pesos del agente al target.

Peculiaridad del DQN y por eso el minibatch para actualizar los pesos: Un batch aleatorio sobre toda la memoria, es off-policy y por lo tanto puede ser de distintas versiones del modelo del agente

Train_interval: cada cuanto actualizo policy (el modelo del agente)

- Actor-Critic: que novedad incluyen con respecto a policy gradient: Introduce el Value, estimamos también el value como salida (el critic) y la probabilidad de las acciones se estiman por el actor.
- En PPO de las dos posibles funciones de coste cual hemos estudiado en la asignatura: Clipped surrogate. Lo del clip venía para que la actualización de las probabilidades sea suave

- En A3C cada agente va por libre… asincronía total
- Cuántas arquitecturas de modelo tiene el DDPG (para espacio de acciones continuas, solo continuo): 4, dos de actor-critic y dos de actor-critic target. Para valor continuo determinista, no es un valor probabilista. Se entrena teniendo en cuenta Bellman
- DQN solo espacios discretos, y Policy Gradient y toda la familia tanto continuos como discretos.

¿¿El uso de gamma para quitar protagonismo a la recompensa?? Más que quitar protagonismo es para ponderar ciertos valores en los que no tenemos tanta confianza, que son los valores futuros. Cuanto más a futuro es un valor el valor de gamma es menor y por lo tanto ponderamos en mayor medida ese valor de recompensa.

Duda de PG. EN los apuntes dice: nuestra función de coste calcula el ratio entre la actualización de la policy y la probabilidad de la acción tomada, para de esa manera no impactar negativamente al proceso de optimización cuando una acción tiene probabilidad alta:

- Al hacer este ratio le damos más importancia a acciones que no se visiten tan a menudo, porque si no cuando una acción tuviera una probabilidad alta se escogería demasiado esa y no nos permitiría visitar otras probabilidades. Es una forma de explotar la exploración y ayudamos a que las variaciones no sean extremas en un corto espacio de tiempo.

En model base: Montecarlo tree-search.

La característica que definía el model base es que conocemos el modelo del entorno. Conocemos la dinámica de cómo se mueven los estados, éramos capaces de saber los estados y las acciones que las relacionaban. Algunos retos del model base es tener en cuenta observaciones parciales, la incertidumbre, la aleatoriedad. La observación no completa es cuando no podemos aplicar directamente planificación, sino que aproximamos el modelo del entorno porque no lo tenemos directamente, intentamos inferir esas dinámicas.