Definizione piano di trattamento radioterapico con l'applicazione del Deep Reinforcement Learning

March 22, 2023

Contents

1 Deep Reinforcement Learning

Il deep reinforcement learning è una sottocategoria dell'apprendimento automatico in cui si autilizzano tecniche di apprendimento per far sì che un agente interagisca con un ambiente al fine di massimizzare una ricompensa. L'agente è in grado di eseguire diverse azioni all'interno dell'ambiente e osservare le conseguenze di ogni azione, utilizzando queste informazioni per imparare quali azioni sono più efficaci ai fini della massimizzazione della ricompensa. Il Deep reinforcement Learning si distingue da altre forme di apprendimento automatico in quanto l'agente è in grado di apprendere sia dall'esperienza che dall'osservazione dell'ambiente, e ultilizza queste informazioni per prendere decisioni in modo autonomo.

2 Applicazione del Deep reinforcement Learning

L'algoritmo lavora dell'ambito sanitario, in particolare si vuole automatizzare la generazioni di piani di trattamento radioterapico (Dose, angoli di ingresso..) il più possibili simili a quelli stabiliti dal medico, piani che solitamente vengono ottenuti risolvendo un problema di ottimizzazione vincolato. Il Deep Reinforcement Learning viene utilizzato per la definizione dei parametri, ottenuti dalla rete neurale addestrata, i quali utilizzati come input nel problema di ottimizzazione vincolato garantiscono la realizzazione del piano di trattamento ottimale.

3 Ambiente del DRL

Il deep reinforcement learning ha come obiettivo l'addestramento di un agente il quale, tramite l'ottenimento di Feedback (positivi o negativi) dall'ambiente, apprende quali sono le azioni ottime da compiere per poter massimizzare un certo guadagno al termine degli episodi. Risulta quindi necessario definire l'ambiente, la funzione guadagno, gli stati del sistema che l'agente raggiunge e le azioni che l'agente deve intraprendere. Per questo specifico algoritmo ed in questo specifico ambito viene utilizzato **MatRad**, codice operSource fornito da Matlab che garantisce, posti specifici parametri, la risoluzione del problema di ottimizzazione vincolato.

4 MatRad

MatRad è un codice operSource fornito da MatLab che offre la possibilità, tramite codice compilabile ed eseguibile o tramite una semplice GUI la risoluzione di problemi di ottimizzazione nell'ambito sopra descritto. In particolare i parametri che può ricevere in input sono i seguenti:

Gantry Angle = Angolo del macchinario, può assumere valori compresi tra 0 e 359 gradi

Couch Angle = Angolo del letto robotizzato, può assumere valori compresi tra 0 e 359 gradi

Radiation Mode = Tipologia di particelle utilizzate **IsoCenter** = Punto in cui convergono i raggi

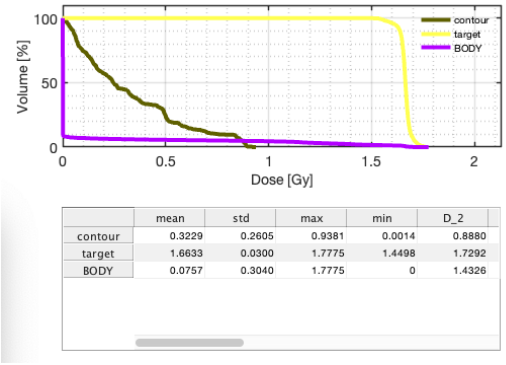
Number of fractions = Frazioni di dose, tale valore viene utilizzato anche per una

ricostruzione tridimensionale della dose.

E' inoltre possibile inserire constraints o vincoli al problema di ottimizzazione nelle diverse aree di interesse TARGET ed OAR. Un possibile vincolo può riguardare la dose nella zone target o tumore da trattare. Si vuole, ad esempio, che il 100 % della dose venga immesso nel 97 % della zona TARGET.
Gli output restituiti da MatRad oltre al piano ottimizzato sono il grafico DVH (Dose Volume Histogram) e la ricostruzione tridimensionale della dose. Il DVH in particolare viene utilizzato per la definizione dello stato del sistema raggiunto dall'agente intrapresa una certa azione e per la funzione guadagno, la quale, misura come varia, data una certa azione e raggiunto un certo stato del sistema il reward finale per l'agente addestrato.

5 Stato, Guadagno ed Azioni

Lo stato e il guadagno vengono definiti a partire dal DVH (Dose Volume Histogram) ottenuto in output lanciando in Python la simulazione di MatRad. In particolare l'output (in MatRad) che si ottiene è il seguente:



	mean	std	max	min	D_2
contour	0.3229	0.2605	0.9381	0.0014	0.8880
target	1.6633	0.0300	1.7775	1.4498	1.7292
BODY	0.0757	0.3040	1.7775	0	1.4326

Figure 1: Output del grafico DVH con le relative zone di interesse

Lo stato del sistema è rappresentato dall'immagine (file DICOM), il DVH, la dose e l'informazione su tutti i fasci precedenti (in forma vettoriale). L'obiettivo è quello di avvicinarsi quanto più possibile al DVH ottenuto dal piano ottimale fornito da MatRad stesso svolgendo ad ogni iterazione delle azioni. L'azione prevede l'aumento del numero di fasci e quindi della dose utilizzata nel trattamento stesso andando a variare l'angolo di ingresso del fascio. Ad ogni iterazione partendo da uno stato iniziale viene quindi aggiunto un fascio con un nuovo angolo (azione), tale azione permetterà il raggiungimento di un certo Reward che deve essere massimizzato. In particolare lo spazio delle azioni è così definito $A = \{0°, 2°, ...STOP\}$ con $STOP$ che indica l'azione finale. L'addestramento termina quando viene eseguita l'azione di STOP e quando il numero massimo di iterazioni viene raggiunto ovvero al raggiungimento dello stato terminale.

La funzione guadagno viene definita formalmente come :

$$R = -\alpha + \Delta_{performance}$$

6 Algoritmi di addestramento

6.1 PPO

Proximal Policy Optimization è un algoritmo di ottimizzazione della polocy utilizzato in ambito di apprendimento automatico per la risoluzione di problemi di apprendimento per rinforzo. Si tratta di una variante dell'algoritmo di ottimizzazione della politica di base (Policy Gradient). A differenaza di altri algoritmi di apprendimento per rinforzo, come il Deep Q-Learning, che si basano sulla costruzione di una funzione di valore Q, PPO cerca di ottimizzare direttamente la politica di un agente.

PPO utilizza una funzione di "vantaggio" per misurare l'efficacia di una determinata azione in uno stato specifico. In pratica, l'algoritmo cerca di massimizzare il vantaggio atteso della politica attuale, limitando al contempo la quantità di cambiamento apportata alla politica di ogni interazione. Ciò viene fatto utilizzando una formulazione di "clip" della politica, che impedisce al modello di allontanarsi troppo dalla politica ottimale precedente.

Inoltre, PPO utilizza una tecnica di "batch sampling" per selezionare solo un sottoinsieme di dati dal buffer replay per l'addestramento del modello. Ciò consente di ridurre il tempo di addestramento del modello e di evitare di essere troppo influenzati da dati di replay anomali o fuori modello.

Formalmente:

Funzione di vantaggio A di un'azione è una misura dell'efficacia di un'azione in un determinato stato. Viene calcolata come la differenza tra la funzione di valore di stato Q e la funzione di valore di stato V. Più precisamente:

$$A(s,a) = Q(s,a) - V(s)$$

dove s è uno stato specifico e a un'azione specifica. $Q(s,a)$ misura il valore atteso di fare l'azione a nello stato s, mentre $V(s)$ misura il valore atteso di essere nello stato s. Quindi $A(s,a)$ misura il vantaggio di fare l'azione a nello stato s rispetto al semplice fatto di essere nello stato s.
Il vantaggio attesso della politica attuale π è dato dalla somma del vantaggio atteso di ogni possibile stato-azione, ponderato dalla probabilità di ottenere quello stato-azione secondo la politica attuale:

$$\mathbb{E}_\pi[A(s,a)] = \sum_s P(s) \sum_a \pi(a|s) A(s,a)$$

dove $P(s)$ è la distribuzione di probabilità di stato (cioè la probabilità di ottenere lo

stato s), $\pi(a|s)$ è la politica (cioè, la probabilità di fare l'azione a nello stato s) e $A(s,a)$ è il vantaggio di fare l'azione a nello stato s come sopra definito.

L'obiettivo di PPO è quello di massimizzare questa quantità, ovvero di trovare la politica che massimizza il vantaggio atteso. Ciò viene fatto utilizzando una formulazione di "clip" della politica, che impedisce al modello di allontanarsi troppo dalla politica ottimale precedente. La "clip" della politica è data da:

$$L_{clip} = \gamma^* A(s,a) - clip(\gamma^* A(s,a), 1-\epsilon, 1+\epsilon) * A(s,a)$$

Dove γ è un fattore di sconto (che misura l'importanza del vantaggio attuale rispetto al vantaggio futuro), ϵ è un paramentro di "clip", e $clip(x, lower, upper)$ è una funzione che ritorna x se x è compreso tra $lower$ e $upper$, $lower$ se x è minore di $lower$ e $upper$ se x è maggiore di $upper$.

La funzione di "clip" serve a limitare la quantità di cambiamento apportata alla politica in ogni iterazione, impedendo al modello di allotanarsi troppo dalla policy ottimale precedente.

Infine la Loss Function di PPO viene definita come segue:

$$L = \alpha L_{clip} - \mathbb{E}[A(s,a)]$$

dove α è un parametro di trade-off che controlla l'importanza relativa della formulazione di "clip" della politica rispetto al vantaggio atteso della politica. α determina quindi quando il modello dovrebbe essere penalizzato se cerca di allontanarsi troppo dalla politica ottimale precedente.