Progetto SOAR

ESCAPE-ROOM

Pietro Sangermano e Michele Metta

Problema

Il nostro agente è bloccato in una stanza dalla quale deve cercare di uscire sfruttando gli oggetti disponibili in modo da rompere l’unica finestra presente al suo interno. La finestra è blindata ma ha un punto debole sul bordo ed è posizionata più in alto rispetto al nostro agente.

Environment e strategie

Il nostro agente ha a disposizione nella stanza 5 oggetti disposti in varie parte di essa: una molla, un bastone, una roccia e 2 tronchi d'albero (non spostabili).

Le fasi principali seguite dall'agente sono le seguenti:

1. L'agente ha la possibilità di combinare diversi oggetti per ottenere un qualcosa che può danneggiare la finestra più gravemente.
2. Può posizionarsi in diversi punti della stanza in modo da ottenere un miglior risultato lanciando l'oggetto verso la finestra.
3. Infine, può scegliere quale parte della finestra quest'ultima colpire.

- Ognuna di queste scelte è seguita da una fase di reward nella quale ogni scelta fatta dall'agente verrà ricompensata, in modo tale da permettere all'agente, durante i diversi cicli di esecuzione di poter apprendere: 1) quale sia la combinazione di oggetti migliore, 2) in quale posizione sarebbe meglio posizionarsi per colpire la finestra e 3) quale parte della finestra colpire, premiando la scelta giusta e screditando quella «sbagliata».

Comportamento dell' agente

1. Lo stato iniziale dell'agente ci dice che non ha nessun oggetto ed è in una certa posizione nella stanza.
2. L'agente osserverà la stanza per rendersi conto della posizione degli oggetti nella stessa e pianificherà il da farsi seguendo queste fasi:

    1. Sceglie la combinazione di strumenti da utilizzare (una volta preso uno strumento decide se prenderne un altro o meno)
    2. Sceglie da quale posizione lanciare gli oggetti (sui tronchi, nord, sud, est, ovest)
    3. Sceglie quale parte della finestra colpire (border o center)
    4. Si muove nella posizione nella quale si trova l'oggetto (o gli oggetti) scelto/i al punto 1.
    5. A questo punto l'agente lancia contro la finestra

- Dopo il lancio alla finestra verrà calcolato il danno fatto ad essa rispetto alle azioni intraprese dall'agente (infatti i reward pesano ogni azioni elencata prima e in base a ciò che viene fatto si calcola l'output di danno del lancio).
- Se la finestra si rompe l'agente potrà scappare terminando l'esecuzione, altrimenti gli oggetti che aveva in mano verranno posizionati in una posizione random della stanza e verrà inizializzata una nuova esecuzione dove l'agente ripartirà dalla posizione finale del ciclo precedente, ricominciando la sua routine di ricerca, assemblaggio, positioning e lancio.

Sistema di Reward

3 sistemi di reward :

- Reward sulla combinazione di oggetti
  - Spring-stick = 1
  - Rocks-stick = -1
  - Rocks-spring = -1
  - Stick = -1.1
  - Rocks = -1.1
  - Spring = -1.1
- Reward sulla posizione della stanza da cui lanciare gli oggetti combinati:
  - tronchi = 1; centro = -0.5; nord = -0.5; sud = -2; est = -1; ovest = -1
- Reward sulla posizione della finestra verso cui effettuare il lancio:
  - Bordo = 1
  - Centro = -1

Sistema di Reward

I sistemi di reward scelti guidano l'apprendimento dell'agente in modo tale che le sue scelte, a seguito di diversi tentativi, cadano su alcune "combinazioni vincenti":

1. Best Reward per la strategia di combinazione : tool1(rametto + molla) con reward +1
2. Best Reward positioning migliore da cui tirare: on_trunks (Salire sui tronchi) con reward +1
3. Best Reward parte della finestra da colpire: border (Mirare ai bordi della finestra) con reward +1

```
agent*rl*strategy*fire*on*CENTER  1.000000 -3.456334
agent*rl*strategy*fire*on*BORDER  6.000000 4.515170
agent*rl*strategy-on-NORD  0.000000 0
agent*rl*strategy-on-EST  0.000000 0
agent*rl*strategy-on-OVEST  0.000000 0
agent*rl*strategy-on-SUD  1.000000 -0.032900
agent*rl*strategy-on-CENTER  0.000000 0
agent*rl*strategy-on_trunks  6.000000 2.527495
agent*rl*strategy-tool6*SPRING  4.000000 0.595581
agent*rl*strategy-tool5*ROCKS  0.000000 0
agent*rl*strategy-tool4*STICK  1.000000 0.630306
agent*rl*strategy-tool3*ROCKS-SPRING  0.000000 0
agent*rl*strategy-tool2*ROCKS-STICK  0.000000 0
agent*rl*strategy-tool1*SPRING-STICK  2.000000 2.712928
```

```
agent*rl*strategy*fire*on*CENTER  0.000000 0
agent*rl*strategy*fire*on*BORDER  4.000000 8.687305
agent*rl*strategy-on-NORD  0.000000 0
agent*rl*strategy-on-EST  1.000000 2.350364
agent*rl*strategy-on-OVEST  0.000000 0
agent*rl*strategy-on-SUD  0.000000 0
agent*rl*strategy-on-CENTER  0.000000 0
agent*rl*strategy-on_trunks  3.000000 5.851646
agent*rl*strategy-tool6*SPRING  0.000000 0
agent*rl*strategy-tool5*ROCKS  0.000000 0
agent*rl*strategy-tool4*STICK  0.000000 0
agent*rl*strategy-tool3*ROCKS-SPRING  0.000000 0
agent*rl*strategy-tool2*ROCKS-STICK  0.000000 0
agent*rl*strategy-tool1*SPRING-STICK  4.000000 3.831227
```

```
agent*rl*strategy*fire*on*CENTER  8.000000 0.748940
agent*rl*strategy*fire*on*BORDER  3.000000 7.198511
agent*rl*strategy-on-NORD  1.000000 0.267648
agent*rl*strategy-on-EST  1.000000 0.261214
agent*rl*strategy-on-OVEST  0.000000 0
agent*rl*strategy-on-SUD  0.000000 0
agent*rl*strategy-on-CENTER  0.000000 0
agent*rl*strategy-on_trunks  9.000000 3.811028
agent*rl*strategy-tool6*SPRING  0.000000 0
agent*rl*strategy-tool5*ROCKS  0.000000 0
agent*rl*strategy-tool4*STICK  0.000000 0
agent*rl*strategy-tool3*ROCKS-SPRING  0.000000 0
agent*rl*strategy-tool2*ROCKS-STICK  0.000000 0
agent*rl*strategy-tool1*SPRING-STICK  11.000000 3.121705
```

Test effetuati

Abbiamo effettuato 4 test:

1. Test 1 : con indifferent-selection epsilon-greedy (deviation al 10%)
2. Test 2 : con con indifferent-selection epsilon (deviation al 20%)
3. Test 3 : con indifferent-selection boltzmann (temperatura a 25)
4. Test 4: con indifferent-selection boltzmann (temperatura a 1)

Risultati

- Dopo 10 tentativi, con il 10% di deviation, l'agente impiega in media 11 round per completare il task
- Dopo 10 tentativi, con il 20% di deviation, l'agente impiega in media 6 round per completare il task
- Dopo 10 tentativi, con valore 25 di temperatura (default) e il decision making impostato su Boltzmann, l'agente impiega in media 19 round per completare il task.
- Il Risultato migliore raggiunto (in media) è stato quello applicando come decision making "Boltzmann" e usando TEMPERATURA = 1

✓ Con T=1 il metodo di selezione di Boltzmann (basato sulla distribuzione termica di Boltzmann) eseguirà una scelta più deterministica perché andrà ad assegnare una probabilità maggiore alle azioni che hanno valori di reward più alti e quindi queste ultime verranno selezionate molto più facilmente

Decision Making	N° Tentativi	Parametri: e (epsilon-greedy) t (Boltzmann)	N° rules fired (media)	N° cicli di elaborazione (media)	N° rounds (medio)
epsilon-greedy	10	0.1	557	133	11
epsilon-greedy	10	0.2	288	115	6
Boltzmann	10	25	856	545	19
Boltzmann	10	1	152	90	3

Conclusioni (1) – decision making peggiori

- Con il decision making Epsilon-greedy (con deviation = 0.10) abbiamo notato che fa **maggiore difficoltà nel riuscire ad individuare il miglior punto nel quale colpire la finestra**, in particolare se l'agente nei primi rounds sceglie la strategia con reward più basso ("center") difficilmente sceglierà la strategia alternativa più premiante ("border")

- Con il decision making Boltzmann (con T=25) abbiamo notato che l'agente fa maggiore difficoltà nel riuscire a capire **sia qual è il miglior punto nel quale colpire la finestra**, **sia qual è la miglior posizione dalla quale lanciare gli oggetti** e **sia qual è la miglior combinazione di oggetti** (probabilmente questo è dovuto al fatto che il valore T essendo troppo alto, induce una causalità troppo alta nella scelta che l'agente deve prendere in ogni fase)

Conclusioni (2) – decision making migliori

- Il decision making Epsilon-greedy (con deviation = 0.20) anche se inizialmente sceglie "center", riesce in pochi rounds a capire grazie ai rinforzi ricevuti che la scelta migliore per colpire la finestra è "border"
- Il decision making Boltzmann (con T = 1) invece, nella maggior parte dei test effettuati è sempre riuscito fin da subito a capire quali fossero le migliori scelte da selezionare in ogni fase (sicuramente questo comportamento è ottenuto grazie alle probabilità maggiori assegnate alle scelte migliori). Comunque anche in casi "meno fortunati", dove le scelte iniziali comunque non sono ottime, è in grado in pochi rounds di permettere all'agente di poter preferire le scelte migliori

Grazie per l’attenzione