OpenAI Gym radionica

Vladimir Milenković, Filip Vesović

Matematička gimnazija

NEDELJA$^4_{\text{INFORMATIKE}}$

28. mart 2018.

Šta je OpenAI?

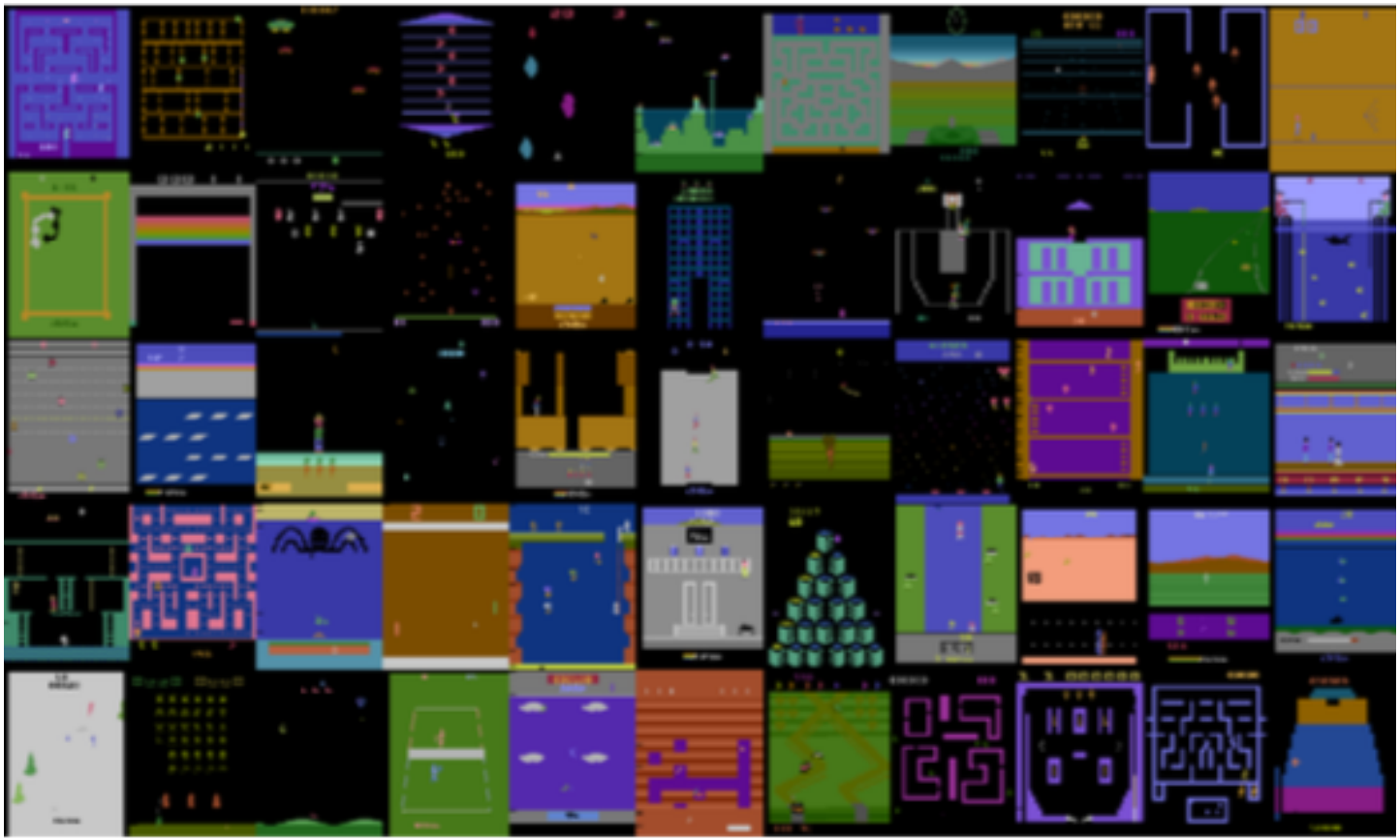
- Pre ove radionice, Petar vam je pričao o Reinforcement learningu.
- OpenAI je neprofitabilna istraživačka kompanija, kojoj je cilj da pravi sigurnu opštu veštačku inteligenciju (AGI), i da obezbedi da se ta ista veštačka inteligencija raširi što je više moguće.
- Kompanija je osnovana krajem 2015. godine u San Francisku.

OpenAI

Šta je OpenAI Gym?

- OpenAI Gym je interfejs koji nam omogućava da na jednostavan način komuniciramo sa širokim spektrom okruženja namenjenih za učenje sa pojačavanjem.
- Možemo se fokusirati samo na algoritme mašinskog učenja, dok će ostali deo posla Gym odraditi za nas

Šta je OpenAI Gym?



Šta je OpenAI Gym?

Gym sadrži mnoge okruženja:

- Atari igrice
- Klasični problemi
- Algoritamski problemi
- Robotički problemi

DEMO

Primer koda


```
import gym
env = gym.make("Taxi-v1")
observation = env.reset()
for _ in range(1000):
  env.render()
  action = env.action_space.sample() # your agent here (this takes random actions)
  observation, reward, done, info = env.step(action)
```

Slika: Primer koda

Objašnjenje koda

- **gym.make('Ime environmenta')** Učitava novi environment
- **env.reset()** Uraditi na početku, kao i svaki put kada želimo da počnemo ispočetka
- **env.render()** Prikazuje trenutno stanje na ekranu
- **env.step(action)** Funkcija koja odigrava potez *action*, i kao rezultat vraća četvorku:
  - *observation* - tuple koji vraća sve potrebne parametre o trenutnom stanju
  - *reward* - nagrada u upravo odigranom potezu
  - *done* - da li je igra već završena; ukoliko jeste, treba resetovati
  - *info* - neke debug informacije, nećemo koristiti

Šta je Keras?

- Keras je python biblioteka koja nam omogućava pravljenje neuralnih mreža.
- Keras sakriva implementaciju mreže na nižim nivoima, i omoguća nam da samo navođenjem strukture, kao i nekih hiperparametara mreže zapravo kreiramo celu mrežu
- Neuralne mreže se u Keras-u prave sloj po sloj; svaki sledeći sloj pravimo pozivanjem neke funkcije koja za argument ima prošli

Funkcije koje ćemo koristiti

- **novi sloj = Dense(broj čvorova, activation='ime aktivacione funkcije')(prošli sloj)**
  - kreira običan, gusto povezan, sloj neuralne mreže.
- Aktivacione funkcije: ['relu', 'linear', 'tanh', 'softmax', 'sigmoid', ...]
- **model = Model(inputs = [lista input layera], outputs = [lista output layera])**
  - pravi model na osnovu datih parametara

Šta je TensorFlow?

- TensorFlow je open source Python biblioteka, specijalizovana za izvršavanje kompleksnih računskih operacija.
- Osnovni model računanja TensorFlow-a je usmeren graf, u kome grane predstavljaju kretanja podataka, a funkcije koje se primenjuju na te podatke su čvorovi grafa.
- Taj grafovski model čini TensorFlow pogodnim pri radu sa neuralnim mrežama.

Korisne funkcije TensorFlow-a

Primenjuje se tako što specificiramo data flow (napravimo gorepomenuti graf), i onda puštamo sve podatke koje treba da obradimo kroz taj graf.

- ***tf.placeholder(type, shape)*** - Kreira mesto za ulaz podataka koje uvek mora biti 'feedovano' podacima.
- **matematičke funkcije - *tf.log*, *tf.square*, *tf.sqrt*, ..** - primenjuje matematičke funkcije na svaki element tenzora
- ***tf.stop_gradient*** - Zaustavlja propagiranje gradijenta i vraća tenzor onakav kakav trenutno jeste

Korisne funkcije TensorFlow-a cont'd

- **funkcije redukcije - *tf.reduce_sum*, *tf.reduce_max*, *tf.reduce_min*, *tf.reduce_mean*** - računa odgovarajuće funkcije po dimenzijama tenzora

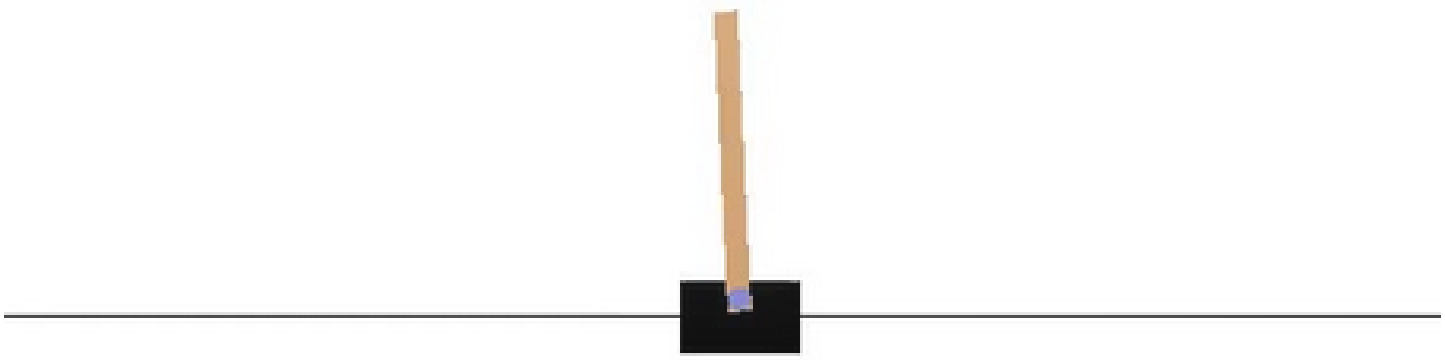
```
x = tf.constant([[1, 1, 1], [1, 1, 1]])
tf.reduce_sum(x)  # 6
tf.reduce_sum(x, 0)  # [2, 2, 2]
tf.reduce_sum(x, 1)  # [3, 3]
tf.reduce_sum(x, 1, keepdims=True)  # [[3], [3]]
tf.reduce_sum(x, [0, 1])  # 6
```

Slika: Primer reduce_sum

O problemima

- Rešavaćemo tri fizička problema:
  - CartPole-v0
  - MountainCar-v0
  - Acrobot-v1
- Za svaki od njih ćemo opisati šta se zapravo radi, moguća stanja, moguće akcije (poteze), kako funkcionišu nagrade, kao i terminirajuća stanja.

Problem 1: CartPole-v0 Prikaz



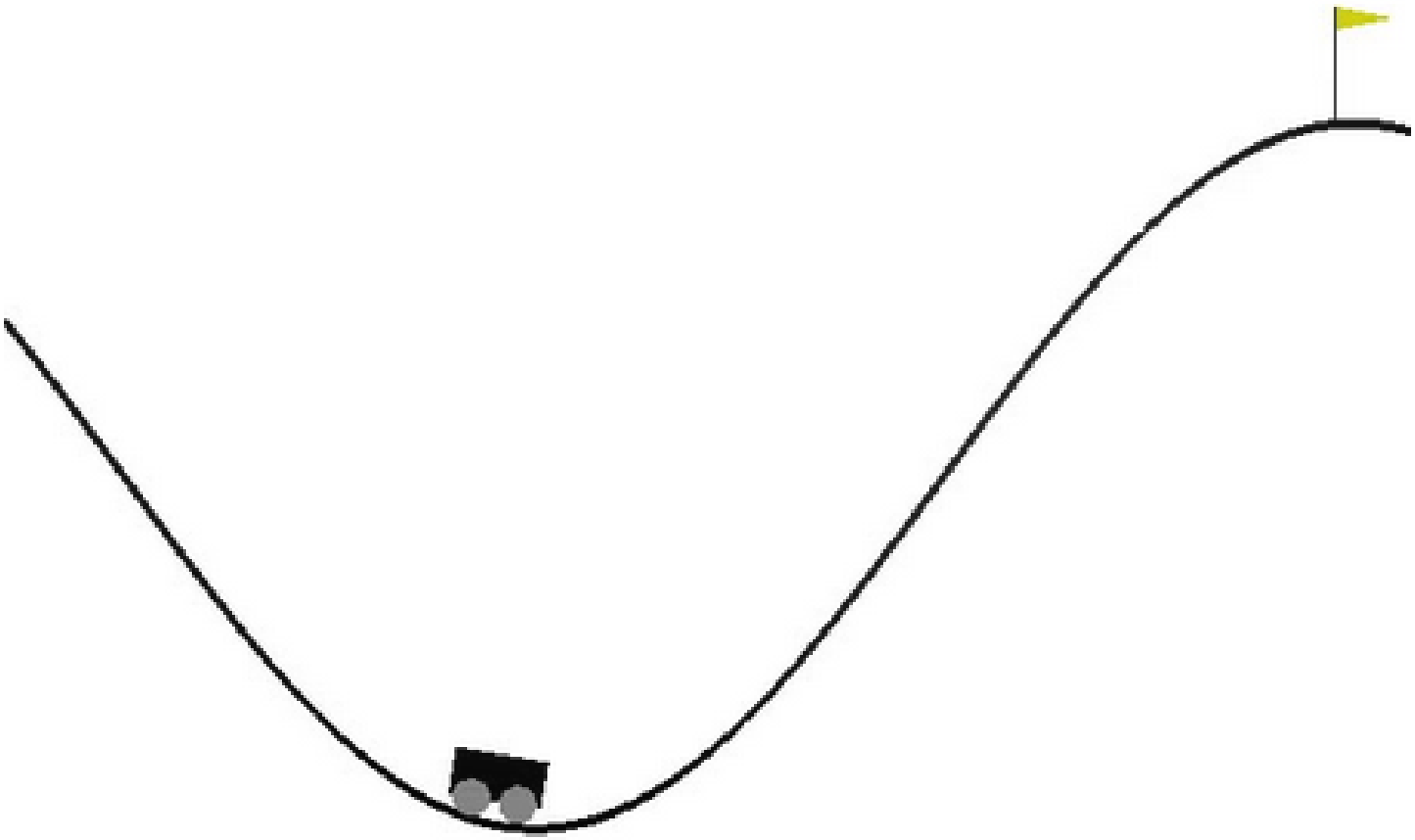
Problem 1: CartPole-v0

- **Opis zadatka**: Štap je pričvšrćen zglobom za kolica, koja se kreću po traci bez trenja. U početku, štap je u vertikalnom položaju, i želimo da pomeranjem kolica levo-desno i ubrzavnjam/usporavanjem kolica postignemo to da štap nikada ne padne.
- **Stanje**: Naše trenutno stanje možemo opisati pomoću 4 parametra:
  - Pozicija kola: realan broj u intervalu (-2.4, 2.4)
  - Ugao pod kojim je nagnut štap u intervalu približnom (-42°, 42°)
  - Brzine vrha štapa i kola: realni brojevi

Problem 1: CartPole-v0 cont'd

- **Potezi**: Dozvoljena su samo dva poteza: gurni levo i gurni desno
- **Nagrada**: Nagrada je jedan za svaki korak tokom kog simulacija i dalje traje
- **Krajnje stanje**: Naša simulacija završava ukoliko je:
  - Ugao štapa veći od 12 po apsolutnoj vrednosti
  - Centar kola došao to kraja ekrana
  - Već odigrano 200 koraka
- Program će biti ocenjen kao tačan ukoliko u proseku traje duže od 195 koraka posle 100 pokušaja

Problem 2: MountainCar-v0 prikaz



Problem 2: MountainCar-v0

- **Opis zadatka**: Imamo automobil koji nije dovoljno jak da se može popeti na brdo krećući se samo u smeru vrha brda. Potrebno je dovesti auto na vrh.
- **Stanje**: Naše trenutno stanje možemo opisati pomoću 2 parametra:
  - Pozicija automobila: realan broj u intervalu (-1.2, 0.6)
  - Trenutna brzina automobila: realan broj u intervalu (-0.07, 0.07)
- **Potezi**: Dozvoljena su tri poteza: gurni levo, ne radi ništa, i gurni desno

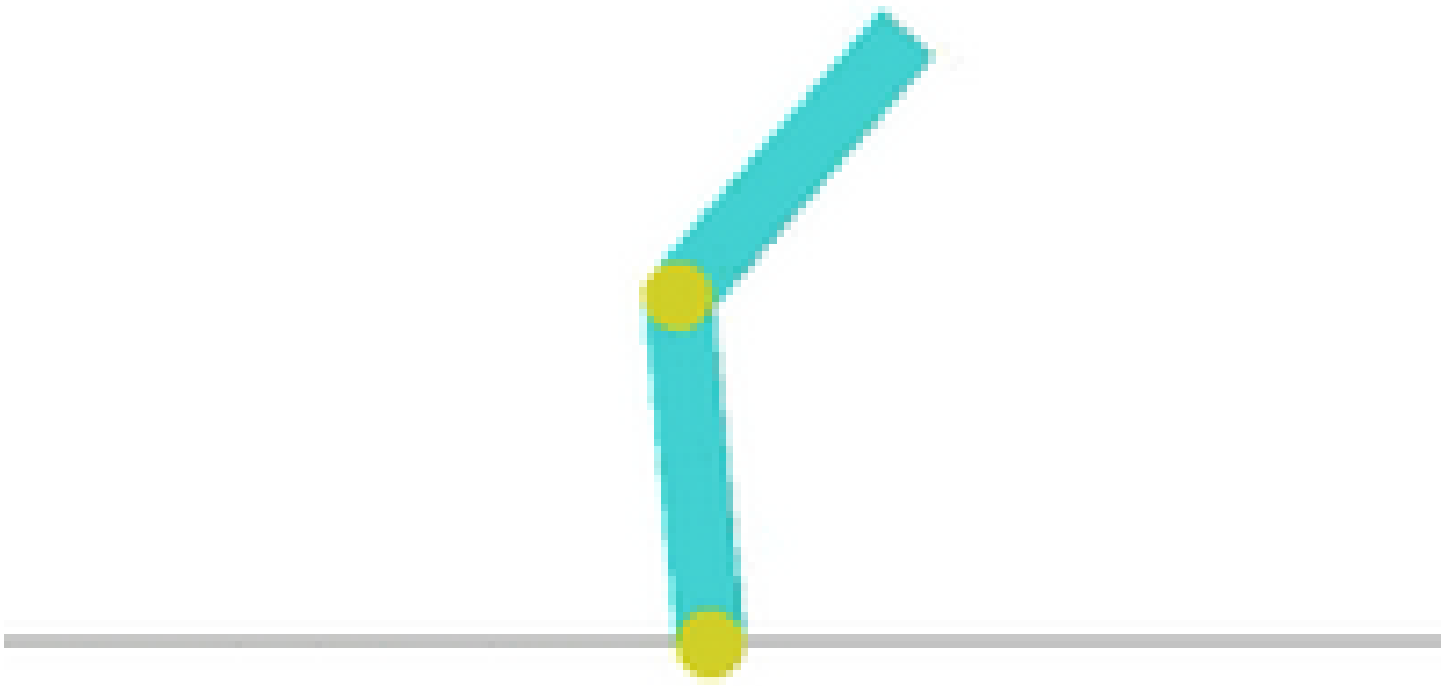
Problem 2: MountainCar-v0 cont'd

- **Nagrada**: Nagrada je -1 za svaki korak tokom kog se automobil još nije popeo na cilj
- **Krajnje stanje**: Proces se završava kada automobil dostigne koordinatu 0.5, ili kada se završi dvesta poteza.
- Program će biti ocenjen kao tačan ukoliko je prosečna nagrada nakon 100 izvršavanja bolja od -110.

Problem 2: MountainCar-v0 cont'd

- Da li postoji neki problem vezan za okruženje MountainCar-v0?
- Koji su načini da taj problem rešimo?

Problem 3: Acrobot-v1 prikaz



Problem 3: Acrobot-v1

- **Opis zadatka**: Imamo dva štapa koja vise, gde je vrh prvog štapa fiksiran, i oni su povezani zglobom. Cilj nam je da zanjišemo donji štap što je moguće više.
- **Stanje**: Naše trenutno stanje možemo opisati pomoću dva ugla (datih kao sinus i kosinus tih uglova), i ugaone brzine oba štapa. Paziti na to da je drugi ugao dat u referentom sistemu prvog štapa, ukoliko je drugi ugao 0, to znači da su štapovi kolinearni.
- **Potezi**: Potezi koji možemo raditi su: ne raditi ništa, i delovanje ugaonim momentom od +1 ili -1 u tački spoja dva štapa.

Problem 3: Acrobot-v1 cont'd

- **Nagrada**: Nagrada je -1 za svaki trenutak kada još uvek nismo prešli preko granice.
- **Krajnje stanje**: Završava se kada kraj donjeg štapa pređe preko određene y koordinate; kad se zanjiše dovoljno.

Vaš zadatak

- Imate implementiran skelet A3C-a, koji ima implementiranu komunikaciju sa OpenAI Gym-om, podršku za multithreading, kao i još neke pomoćne i skeletne funkcije
- Vaš današnji zadatak je:
  - Implementacija mreže za računanje polise, kao i vrednost funkcije V.
  - Implementacija funkcije gubitka
  - Implementacija odabira akcije

  Prvo probati na CartPole-u, kada proradi na njemu, optimizovati kako bi radilo i na ostala dva problema.

SREĆNO!

https://goo.gl/k6qBM6