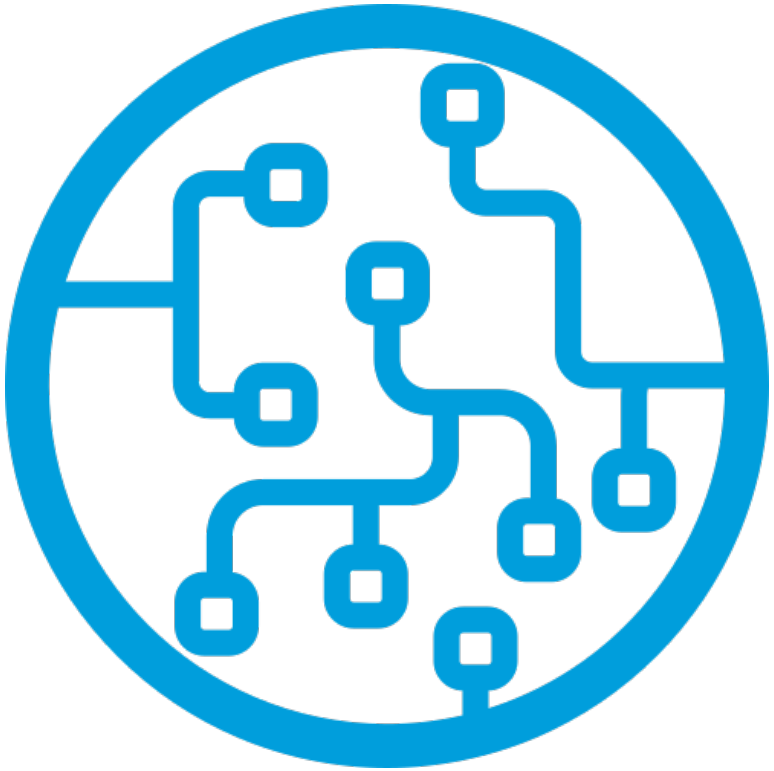
REINFORCEMENT LAERNING

Mattia Spazzoli

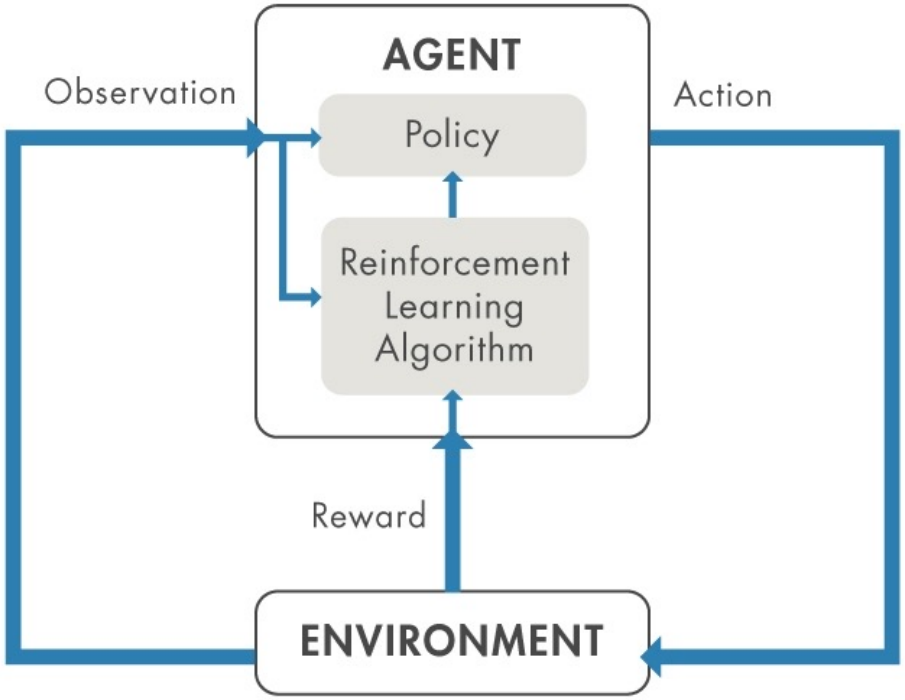
Definizione

Il **Reinforcement Learning** è una tecnica di apprendimento automatico che punta a realizzare agenti autonomi in grado di scegliere azioni da compiere per il conseguimento di determinati obiettivi tramite interazione con l'ambiente in cui sono immersi.

it.wikipedia.org



Modo di apprendimento



L'apprendimento avviene tramite ripetute iterazioni «trial-and-error» con un ambiente dinamico:

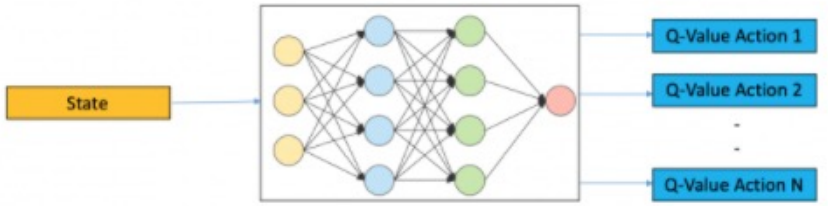
- L' Agente esegue un'azione sull'ambiente
- All'Agente viene restituito un valore di ricompensa che andrà ad aggiornare la politica di apprendimento
- L'Ambiente muta in base all'azione svolta e viene fatto visualizzare all'agente

Flusso di lavoro

1. Definire le attività che l'agente deve apprendere, eventualmente in obiettivi primari e secondari
2. Definire l'ambiente in cui l'agente opera, compresa l'interfaccia di comunicazione tra i due
3. Definire il sistema di ricompensa che l'agente usa per valutare la propria azione
4. Creare l'agente, configurandone l'algoritmo di apprendimento e definendone la politica
5. Addestramento della politica attraverso l'uso di dell'ambiente, della ricompensa e dell'algoritmo di addestramento
6. Valutazione delle performance dell'agente
7. Distribuzione della politica addestrata

RL con agente Deep Q-Learning

Q Learning



Deep Q Learning

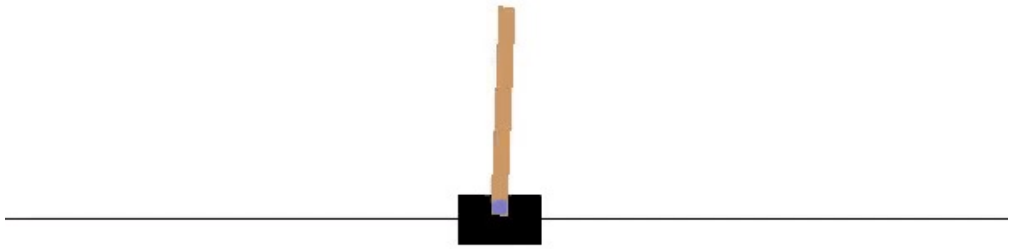
- Basata sul processo decisionale di Markov: una proprietà che ci fornisce la migliore mossa osservando esclusivamente l'ultimo stato
- L'agente eseguirà azioni con massima ricompensa, ottenuta grazie ad una funzione ricorsiva e chiamata Q-Value:

$$Q(s,a) = r(s,a) + \gamma \max_{a} Q(s',a)$$

- Per fronteggiare la potenziale lunghezza del Q learning si utilizza la Deep Q-Learning che approssima il valore di Q tramite una rete neurale:
  - Memoria salvata integralmente
  - Azione dettata dall'uscita massima della rete
  - Funzione di loss data da

$$Loss = (r + \gamma max_{a'} Q(s', a'; \theta') - Q(s, a; \theta))^2$$

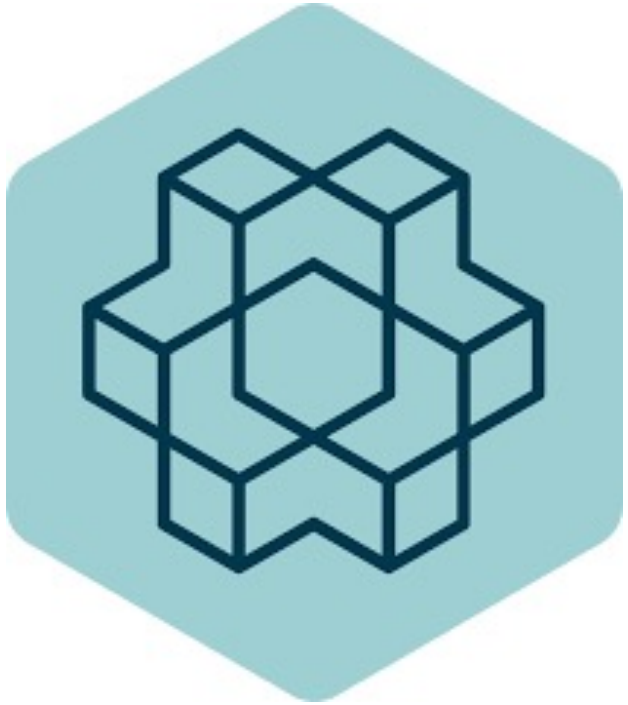
Ambiente CartPole-v0



- Ambiente appartenente a OpenAI Gym
- Le mosse possibili all'interno di questo ambiente sono spostare il carrello a destra o a sinistra
- La ricompensa ricevuta può avere valore 1 o -1
- Un episodio termina se il palo dista dalla verticale più di 15° oppure il carrello sarà a più di 2,4 unità dal centro

OpenaAI Gym - Introduzione

- OpenAi Gym è una collezione di ambienti che possono essere utilizzati per elaborare un algoritmo di Reinforcement Learning.
- Gli ambienti disponibili hanno un'interfaccia condivisa permettendo l'utilizzo di algoritmi generali
- Compatibile con qualisiasi libreria di calcolo numerico
- Spesso utilizzato come interfaccia tra un implementazione RL e l'ambiente fornito



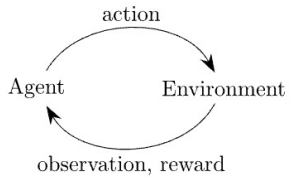
OpenAI Gym – Primitive

Environments

```
import gym
env = gym.make('CartPole-v0')
env.reset()
for _ in range(1000):
    env.render()
    env.step(env.action_space.sample()) # take a random action
env.close()
```

- gym.make() : importa l'ambiente desiderato
- env.render() : mostra l'ambiente ad ogni step
- env.step(): esegue un'azione sull'ambiente e restituisce informazioni importanti per lo step successivo
- env.reset(): porta l'ambiente con lo stato iniziale

Observation




```
observation, reward, done, info = env.step(action)
```

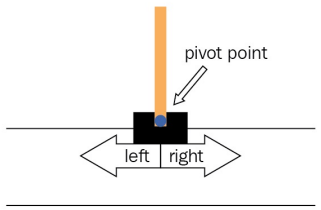
- observatin: oggetto rappresentante lo stato dell'ambiente
- reward: ricompensa ottenuta dall'azione appena
- done: indica se l'episodio è finito
- info: informazioni utili per il debug

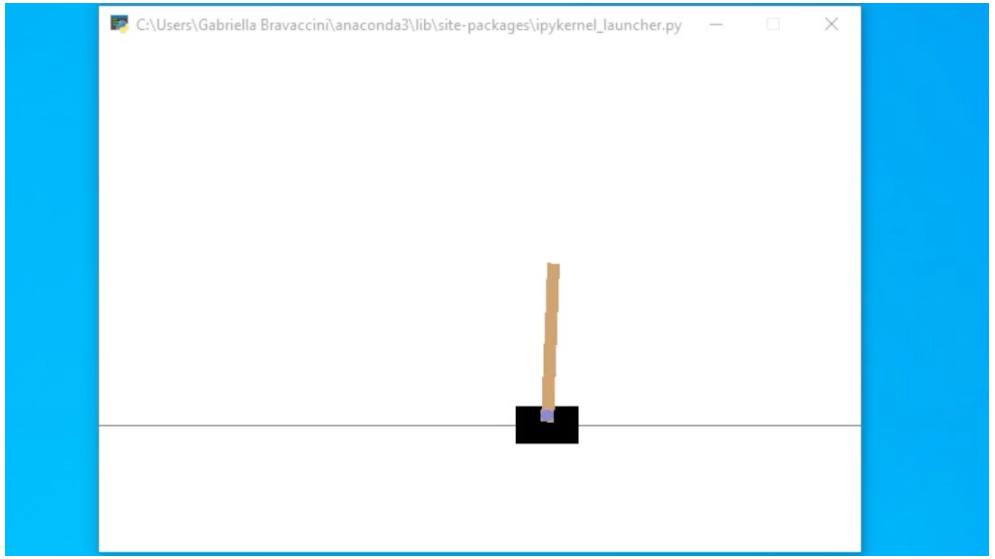
Spaces

```
import gym
env = gym.make('CartPole-v0')
print(env.action_space)
#> Discrete(2)
print(env.observation_space)
#> Box(4,)
```

- env.action_space : contiene le azioni valide per l'ambiente
- env.observation_space : contiene le osservazioni valide per l'ambiente

OpenAI Gym – Esperimento con «CartPole-v0»

- python™
- Gym
- 



- **Deep Q Learning Agent con**

Dopamine - Introduzione

- Dopamine è un framework di ricerca per la prototipazione rapida di algoritmi di Reinforcement Learning
- La caratteristica che lo contraddistingue è il suo basarsi su file di configurazione di tipo Google gin-config
- Supporta i seguenti Agenti: DQN, C51, Arcobaleno, IQN e SAC

Framework di configurazione gin-config

Gin è un framework leggero di configurazione per python basato sull'iniezione di dipendenza. Questo vuol dire che le funzioni o le classi possono essere decorate con la key @gin.configurable consentendo la configurazione dall'esterno attraverso un file gin-config.

Esempio con @gin.configurable:

- Decorazione in python

```
@gin.configurable
def dnn(inputs,
        num_outputs,
        layer_sizes=(512, 512),
        activation_fn=tf.nn.relu):
```

- Binding all'interno del file gin-config

```
# Inside "config.gin"
dnn.layer_sizes = (1024, 512, 128)
```

Possibili decorazioni nel file python:

- @gin.configurable
- @gin.register

Possibili binding nel file gin-config

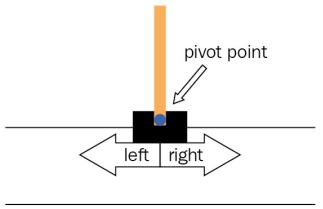
name.param = value
scope/name.parame = value

Possibili riferimenti nel file gin-config

@soma_name
@scope/some_name
@some_name()
%MACRO_NAME

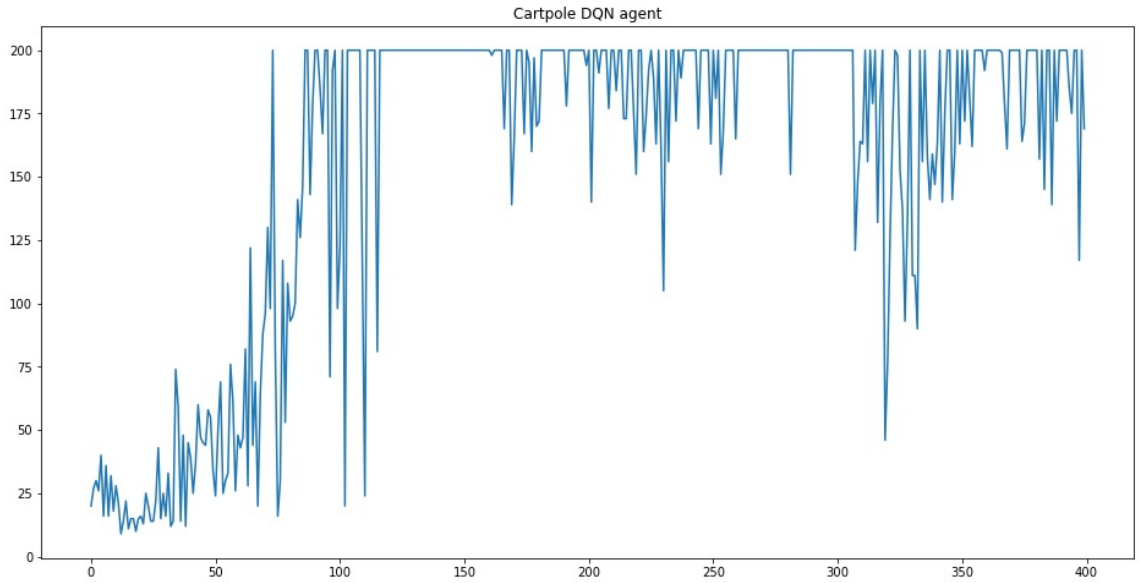
Dopamine – Esperimento con «CartPole-v0»

```
#Dopamine gin-config
DQN_PATH = '/tmp'
dqn_config = """
#Necessary import
import dopamine.discrete_domains.gym_lib
import dopamine.discrete_domains.run_experiment
import dopamine.agents.dqn.dqn_agent
import dopamine.replay_memory.circular_replay_buffer
import gin.tf.external_configurables

#Agent settings
DQNAgent.observation_shape = %gym_lib.CARTPOLE_OBSERVATION_SHAPE
DQNAgent.observation_dtype = %gym_lib.CARTPOLE_OBSERVATION_DTYPE
DQNAgent.stack_size = %gym_lib.CARTPOLE_STACK_SIZE
DQNAgent.network = @gym_lib.CartpoleDQNNetwork
DQNAgent.gamma = 0.99
DQNAgent.update_horizon = 1
DQNAgent.min_replay_history = 500
DQNAgent.update_period = 4
DQNAgent.target_update_period = 100
DQNAgent.epsilon_fn = @dqn_agent.identity_epsilon
DQNAgent.tf_device = '/gpu:0'
DQNAgent.optimizer = @tf.train.AdamOptimizer()
tf.train.AdamOptimizer.learning_rate = 0.001
tf.train.AdamOptimizer.epsilon = 0.0003125

#Environment settings
create_gym_environment.environment_name = 'CartPole'
create_gym_environment.version = 'v0'
create_agent.agent_name = 'dqn'
TrainRunner.create_environment_fn = @gym_lib.create_gym_environment

#Training settings
Runner.num_iterations = 400
Runner.training_steps = 1000
Runner.evaluation_steps = 1000
Runner.max_steps_per_episode = 200

#Buffer settings
WrappedReplayBuffer.replay_capacity = 50000
WrappedReplayBuffer.batch_size = 128
"""
```

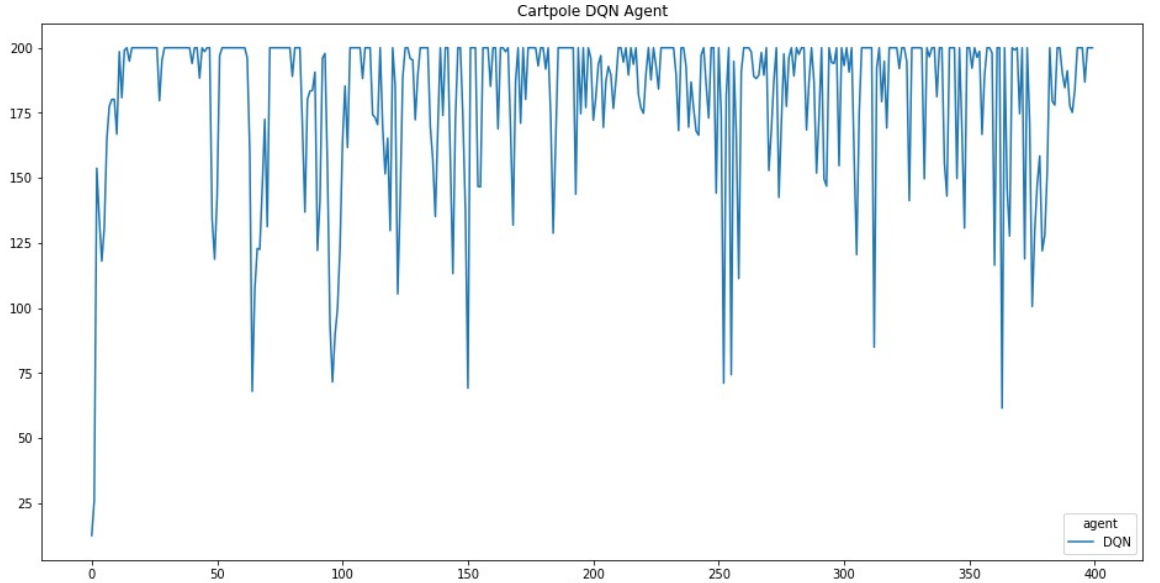
- **Google gin-config e** python™
- Dopamine
- 

- **Deep Q Learning Agent**

Ricompense a confronto



OpenAI Gym



Dopamine

Differenze tra OpenAI Gym e Dopamine

Gym	Dopamine
Insieme di Ambienti	Framework
Facilità nella comprensione del meccanismo di apprendimento con rinforzo	Forte astrazione rispetto alle dinamiche di apprendimento con rinforzo
Difficoltà nella scrittura di Agenti complessi	Facilità nell'utilizzo di Agenti complessi già disponibili
Focus sulla scrittura del codice	Focus sulla configurazione di file gin-config

Riferimenti

Fonti per lo studio del Reinforcment Learning

- https://it.wikipedia.org/wiki/Apprendimento_per_rinforzo
- https://en.wikipedia.org/wiki/Reinforcement_learning
- https://www.youtube.com/watch?v=JgvyzIkgxF0&ab_channel=ArxivInsights
- https://www.analyticsvidhya.com/blog/2019/04/introduction-deep-q-learning-python/

Fonti per lo sviluppo dell'esperimento con OpenAI Gym

- https://gym.openai.com/
- https://towardsdatascience.com/reinforcement-learning-with-openai-d445c2c687d2
- https://github.com/openai/gym

Fonti per lo sviluppo dell'esperimento con Dopamine

- https://medium.com/the-21st-century/google-dopamine-new-rl-framework-f84a35b7fb3f
- https://holmdk.github.io/2020/07/16/DQN_agent_ALE_dopamine.html
- https://github.com/google/gin-config/blob/master/docs/walkthrough.md
- https://opensource.google/projects/dopamine

Riferimenti al materiale GitHub

- Repository dell'attività progettuale: https://github.com/MattiaSpazzoli/Spazzoli-ReinforcementLearning-Experiments
- Esperimento OpenAI Gym: https://github.com/MattiaSpazzoli/Spazzoli-ReinforcementLearning-Experiments/blob/main/Experiments/OpenAiExperiment.ipynb
- Esperimento Dopamine: https://github.com/MattiaSpazzoli/Spazzoli-ReinforcementLearning-Experiments/blob/main/Experiments/DopamineExperiment.ipynb