Q-learning from colab notebook

Setup

In [43]:

```
# Python ≥3.5 is required
import sys
assert sys.version_info >= (3, 5)

# Is this notebook running on Colab or Kaggle?
IS_COLAB = "google.colab" in sys.modules
IS_KAGGLE = "kaggle_secrets" in sys.modules

if IS_COLAB or IS_KAGGLE:
    !apt update && apt install -y libpq-dev libsdl2-dev swig xorg-dev xvfb
    %pip install -U tf-agents pyvirtualdisplay
    %pip install -U gym>=0.21.0
    %pip install -U gym[box2d,atari,accept-rom-license]

# Scikit-Learn ≥0.20 is required
import sklearn
assert sklearn.__version__ >= "0.20"

# TensorFlow ≥2.0 is required
import tensorflow as tf
from tensorflow import keras
assert tf.__version__ >= "2.0"

if not tf.config.list_physical_devices('GPU'):
    print("No GPU was detected. CNNs can be very slow without a GPU.")
    if IS_COLAB:
        print("Go to Runtime > Change runtime and select a GPU hardware accelerator.")
    if IS_KAGGLE:
        print("Go to Settings > Accelerator and select GPU.")

# Common imports
import numpy as np
import os

# to make this notebook's output stable across runs
np.random.seed(42)
tf.random.set_seed(42)

# To plot pretty figures
%matplotlib inline
import matplotlib as mpl
import matplotlib.pyplot as plt
mpl.rc('axes', labelsize=14)
mpl.rc('xtick', labelsize=12)
mpl.rc('ytick', labelsize=12)

# To get smooth animations
import matplotlib.animation as animation
mpl.rc('animation', html='jshtml')

# Where to save the figures
PROJECT_ROOT_DIR = "."
CHAPTER_ID = "rl"
IMAGES_PATH = os.path.join(PROJECT_ROOT_DIR, "images", CHAPTER_ID)
os.makedirs(IMAGES_PATH, exist_ok=True)

def save_fig(fig_id, tight_layout=True, fig_extension="png", resolution=300):
    path = os.path.join(IMAGES_PATH, fig_id + "." + fig_extension)
    print("Saving figure", fig_id)
    if tight_layout:
```

```
        plt.tight_layout()
    plt.savefig(path, format=fig_extension, dpi=resolution)
```

```
Get:1 https://cloud.r-project.org/bin/linux/ubuntu bionic-cran40/ InRelease [3,626 B]
Hit:2 http://ppa.launchpad.net/c2d4u.team/c2d4u4.0+/ubuntu bionic InRelease
Get:3 http://security.ubuntu.com/ubuntu bionic-security InRelease [88.7 kB]
Hit:4 http://archive.ubuntu.com/ubuntu bionic InRelease
Get:5 http://archive.ubuntu.com/ubuntu bionic-updates InRelease [88.7 kB]
Hit:6 http://ppa.launchpad.net/cran/libgit2/ubuntu bionic InRelease
Ign:7 https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1804/x86_64  InRelease
Hit:8 http://ppa.launchpad.net/deadsnakes/ppa/ubuntu bionic InRelease
Get:9 http://archive.ubuntu.com/ubuntu bionic-backports InRelease [74.6 kB]
Hit:10 http://ppa.launchpad.net/graphics-drivers/ppa/ubuntu bionic InRelease
Ign:11 https://developer.download.nvidia.com/compute/machine-learning/repos/ubuntu1804/x86_64
InRelease
Hit:12 https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1804/x86_64  Release
Hit:13 https://developer.download.nvidia.com/compute/machine-learning/repos/ubuntu1804/x86_64
Release
Fetched 256 kB in 2s (117 kB/s)
Reading package lists... Done
Building dependency tree
Reading state information... Done
58 packages can be upgraded. Run 'apt list --upgradable' to see them.
Reading package lists... Done
Building dependency tree
Reading state information... Done
swig is already the newest version (3.0.12-1).
libpq-dev is already the newest version (10.19-0ubuntu0.18.04.1).
xorg-dev is already the newest version (1:7.7+19ubuntu7.1).
libsdl2-dev is already the newest version (2.0.8+dfsg1-1ubuntu1.18.04.4).
xvfb is already the newest version (2:1.19.6-1ubuntu4.9).
0 upgraded, 0 newly installed, 0 to remove and 58 not upgraded.
Requirement already satisfied: tf-agents in /usr/local/lib/python3.7/dist-packages (0.11.0)
Requirement already satisfied: pyvirtualdisplay in /usr/local/lib/python3.7/dist-packages (2.2)
Requirement already satisfied: absl-py>=0.6.1 in /usr/local/lib/python3.7/dist-packages (from t
f-agents) (0.12.0)
Requirement already satisfied: pillow in /usr/local/lib/python3.7/dist-packages (from tf-agent
s) (7.1.2)
Requirement already satisfied: six>=1.10.0 in /usr/local/lib/python3.7/dist-packages (from tf-a
gents) (1.15.0)
Requirement already satisfied: gin-config>=0.4.0 in /usr/local/lib/python3.7/dist-packages (fro
m tf-agents) (0.5.0)
Requirement already satisfied: protobuf>=3.11.3 in /usr/local/lib/python3.7/dist-packages (from
tf-agents) (3.17.3)
Requirement already satisfied: cloudpickle>=1.3 in /usr/local/lib/python3.7/dist-packages (from
tf-agents) (1.3.0)
Requirement already satisfied: wrapt>=1.11.1 in /usr/local/lib/python3.7/dist-packages (from tf
-agents) (1.13.3)
Requirement already satisfied: typing-extensions>=3.7.4.3 in /usr/local/lib/python3.7/dist-pack
ages (from tf-agents) (3.10.0.2)
Requirement already satisfied: tensorflow-probability>=0.14.1 in /usr/local/lib/python3.7/dist-
packages (from tf-agents) (0.15.0)
Requirement already satisfied: numpy>=1.13.3 in /usr/local/lib/python3.7/dist-packages (from tf
-agents) (1.19.5)
Requirement already satisfied: gym>=0.17.0 in /usr/local/lib/python3.7/dist-packages (from tf-a
gents) (0.21.0)
Requirement already satisfied: importlib-metadata>=4.8.1 in /usr/local/lib/python3.7/dist-packa
ges (from gym>=0.17.0->tf-agents) (4.8.2)
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.7/dist-packages (from import
lib-metadata>=4.8.1->gym>=0.17.0->tf-agents) (3.6.0)
Requirement already satisfied: gast>=0.3.2 in /usr/local/lib/python3.7/dist-packages (from tens
orflow-probability>=0.14.1->tf-agents) (0.4.0)
Requirement already satisfied: decorator in /usr/local/lib/python3.7/dist-packages (from tensor
flow-probability>=0.14.1->tf-agents) (4.4.2)
Requirement already satisfied: dm-tree in /usr/local/lib/python3.7/dist-packages (from tensorfl
ow-probability>=0.14.1->tf-agents) (0.1.6)
Requirement already satisfied: EasyProcess in /usr/local/lib/python3.7/dist-packages (from pyvi
rtualdisplay) (0.3)
```

```
Requirement already satisfied: gym[accept-rom-license,atari,box2d] in /usr/local/lib/python3.7/
dist-packages (0.21.0)
Requirement already satisfied: cloudpickle>=1.2.0 in /usr/local/lib/python3.7/dist-packages (fr
om gym[accept-rom-license,atari,box2d]) (1.3.0)
Requirement already satisfied: importlib-metadata>=4.8.1 in /usr/local/lib/python3.7/dist-packa
ges (from gym[accept-rom-license,atari,box2d]) (4.8.2)
Requirement already satisfied: numpy>=1.18.0 in /usr/local/lib/python3.7/dist-packages (from gy
m[accept-rom-license,atari,box2d]) (1.19.5)
Requirement already satisfied: ale-py~=0.7.1 in /usr/local/lib/python3.7/dist-packages (from gy
m[accept-rom-license,atari,box2d]) (0.7.3)
Requirement already satisfied: pyglet>=1.4.0 in /usr/local/lib/python3.7/dist-packages (from gy
m[accept-rom-license,atari,box2d]) (1.5.0)
Requirement already satisfied: box2d-py==2.3.5 in /usr/local/lib/python3.7/dist-packages (from
gym[accept-rom-license,atari,box2d]) (2.3.5)
Requirement already satisfied: autorom[accept-rom-license]~=0.4.2 in /usr/local/lib/python3.7/d
ist-packages (from gym[accept-rom-license,atari,box2d]) (0.4.2)
Requirement already satisfied: importlib-resources in /usr/local/lib/python3.7/dist-packages (f
rom ale-py~=0.7.1->gym[accept-rom-license,atari,box2d]) (5.4.0)
Requirement already satisfied: click in /usr/local/lib/python3.7/dist-packages (from autorom[ac
cept-rom-license]~=0.4.2->gym[accept-rom-license,atari,box2d]) (7.1.2)
Requirement already satisfied: tqdm in /usr/local/lib/python3.7/dist-packages (from autorom[acc
ept-rom-license]~=0.4.2->gym[accept-rom-license,atari,box2d]) (4.62.3)
Requirement already satisfied: requests in /usr/local/lib/python3.7/dist-packages (from autorom
[accept-rom-license]~=0.4.2->gym[accept-rom-license,atari,box2d]) (2.23.0)
Requirement already satisfied: AutoROM.accept-rom-license in /usr/local/lib/python3.7/dist-pack
ages (from autorom[accept-rom-license]~=0.4.2->gym[accept-rom-license,atari,box2d]) (0.4.2)
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.7/dist-packages (from import
lib-metadata>=4.8.1->gym[accept-rom-license,atari,box2d]) (3.6.0)
Requirement already satisfied: typing-extensions>=3.6.4 in /usr/local/lib/python3.7/dist-packag
es (from importlib-metadata>=4.8.1->gym[accept-rom-license,atari,box2d]) (3.10.0.2)
Requirement already satisfied: future in /usr/local/lib/python3.7/dist-packages (from pyglet>=
1.4.0->gym[accept-rom-license,atari,box2d]) (0.16.0)
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-packages (from req
uests->autorom[accept-rom-license]~=0.4.2->gym[accept-rom-license,atari,box2d]) (2.10)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /usr/local/lib/python
3.7/dist-packages (from requests->autorom[accept-rom-license]~=0.4.2->gym[accept-rom-license,at
ari,box2d]) (1.24.3)
Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.7/dist-packages (fro
m requests->autorom[accept-rom-license]~=0.4.2->gym[accept-rom-license,atari,box2d]) (3.0.4)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.7/dist-packages (fr
om requests->autorom[accept-rom-license]~=0.4.2->gym[accept-rom-license,atari,box2d]) (2021.10.
8)
No GPU was detected. CNNs can be very slow without a GPU.
Go to Runtime > Change runtime and select a GPU hardware accelerator.
```

In [44]:

```
def update_scene(num, frames, patch):
    patch.set_data(frames[num])
    return patch,

def plot_animation(frames, repeat=False, interval=40):
    fig = plt.figure()
    patch = plt.imshow(frames[0])
    plt.axis('off')
    anim = animation.FuncAnimation(
        fig, update_scene, fargs=(frames, patch),
        frames=len(frames), repeat=repeat, interval=interval)
    plt.close()
    return anim
```

Deep Q-Network

Let's build the DQN. Given a state, it will estimate, for each possible action, the sum of discounted future rewards it can expect after it plays that action (but before it sees its outcome):

In [45]:

```
import gym
```

```
keras.backend.clear_session()
tf.random.set_seed(42)
np.random.seed(42)

env = gym.make("CartPole-v0")
input_shape = env.observation_space.shape
print(env.observation_space.shape)
n_outputs = 2 # == env.action_space.n

model = keras.models.Sequential([
    keras.layers.Dense(32, activation="elu", input_shape=input_shape),
    keras.layers.Dense(32, activation="elu"),
    keras.layers.Dense(n_outputs)
])
```

```
(4,)
```

To select an action using this DQN, we just pick the action with the largest predicted Q-value. However, to ensure that the agent explores the environment, we choose a random action with probability `epsilon`.

In [46]:
```
def epsilon_greedy_policy(state, epsilon=0):
    if np.random.rand() < epsilon:
        return np.random.randint(n_outputs)
    else:
        Q_values = model.predict(state[np.newaxis])
        return np.argmax(Q_values[0])
```

We will also need a replay memory. It will contain the agent's experiences, in the form of tuples: `(obs, action, reward, next_obs, done)`. We can use the `deque` class for that (but make sure to check out DeepMind's excellent Reverb library for a much more robust implementation of experience replay):

In [47]:
```
from collections import deque

replay_memory = deque(maxlen=2000)
```

And let's create a function to sample experiences from the replay memory. It will return 5 NumPy arrays: `[obs, actions, rewards, next_obs, dones]`.

In [48]:
```
def sample_experiences(batch_size):
    indices = np.random.randint(len(replay_memory), size=batch_size)
    batch = [replay_memory[index] for index in indices]
    states, actions, rewards, next_states, dones = [
        np.array([experience[field_index] for experience in batch])
        for field_index in range(5)]
    return states, actions, rewards, next_states, dones
```

Now we can create a function that will use the DQN to play one step, and record its experience in the replay memory:

In [49]:
```
def play_one_step(env, state, epsilon):
    action = epsilon_greedy_policy(state, epsilon)
    next_state, reward, done, info = env.step(action)
    replay_memory.append((state, action, reward, next_state, done))
    return next_state, reward, done, info
```

Lastly, let's create a function that will sample some experiences from the replay memory and perform a training step:

Notes:

- The first 3 releases of the 2nd edition were missing the `reshape()` operation which converts `target_Q_values` to a column vector (this is required by the `loss_fn()`).
- The book uses a learning rate of 1e-3, but in the code below I use 1e-2, as it significantly improves training. I also tuned the learning rates of the DQN variants below.

In [50]:
```
batch_size = 32
discount_rate = 0.95
optimizer = keras.optimizers.Adam(learning_rate=1e-2)
loss_fn = keras.losses.mean_squared_error

def training_step(batch_size):
    experiences = sample_experiences(batch_size)
    states, actions, rewards, next_states, dones = experiences
    next_Q_values = model.predict(next_states)
    max_next_Q_values = np.max(next_Q_values, axis=1)
    target_Q_values = (rewards +
                       (1 - dones) * discount_rate * max_next_Q_values)
    target_Q_values = target_Q_values.reshape(-1, 1)
    mask = tf.one_hot(actions, n_outputs)
    with tf.GradientTape() as tape:
        all_Q_values = model(states)
        Q_values = tf.reduce_sum(all_Q_values * mask, axis=1, keepdims=True)
        loss = tf.reduce_mean(loss_fn(target_Q_values, Q_values))
    grads = tape.gradient(loss, model.trainable_variables)
    optimizer.apply_gradients(zip(grads, model.trainable_variables))
```

And now, let's train the model!

In [51]:
```
env.seed(42)
np.random.seed(42)
tf.random.set_seed(42)

rewards = []
best_score = 0
```

In [52]:
```
%%bash
apt-get install swig
# install required system dependencies
apt-get install -y xvfb x11-utils

# install required python dependencies (might need to install additional gym extras depending)
pip install gym[box2d] pyvirtualdisplay PyOpenGL PyOpenGL-accelerate

pip install piglet
```

```
Reading package lists...
Building dependency tree...
Reading state information...
swig is already the newest version (3.0.12-1).
0 upgraded, 0 newly installed, 0 to remove and 58 not upgraded.
Reading package lists...
Building dependency tree...
Reading state information...
x11-utils is already the newest version (7.7+3build1).
xvfb is already the newest version (2:1.19.6-1ubuntu4.9).
0 upgraded, 0 newly installed, 0 to remove and 58 not upgraded.
Requirement already satisfied: gym[box2d] in /usr/local/lib/python3.7/dist-packages (0.21.0)
Requirement already satisfied: pyvirtualdisplay in /usr/local/lib/python3.7/dist-packages (2.2)
Requirement already satisfied: PyOpenGL in /usr/local/lib/python3.7/dist-packages (3.1.5)
Requirement already satisfied: PyOpenGL-accelerate in /usr/local/lib/python3.7/dist-packages
(3.1.5)
Requirement already satisfied: EasyProcess in /usr/local/lib/python3.7/dist-packages (from pyvi
rtualdisplay) (0.3)
Requirement already satisfied: cloudpickle>=1.2.0 in /usr/local/lib/python3.7/dist-packages (fr
```

```
om gym[box2d]) (1.3.0)
Requirement already satisfied: numpy>=1.18.0 in /usr/local/lib/python3.7/dist-packages (from gy
m[box2d]) (1.19.5)
Requirement already satisfied: importlib-metadata>=4.8.1 in /usr/local/lib/python3.7/dist-packa
ges (from gym[box2d]) (4.8.2)
Requirement already satisfied: box2d-py==2.3.5 in /usr/local/lib/python3.7/dist-packages (from
gym[box2d]) (2.3.5)
Requirement already satisfied: pyglet>=1.4.0 in /usr/local/lib/python3.7/dist-packages (from gy
m[box2d]) (1.5.0)
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.7/dist-packages (from import
lib-metadata>=4.8.1->gym[box2d]) (3.6.0)
Requirement already satisfied: typing-extensions>=3.6.4 in /usr/local/lib/python3.7/dist-packag
es (from importlib-metadata>=4.8.1->gym[box2d]) (3.10.0.2)
Requirement already satisfied: future in /usr/local/lib/python3.7/dist-packages (from pyglet>=
1.4.0->gym[box2d]) (0.16.0)
Requirement already satisfied: piglet in /usr/local/lib/python3.7/dist-packages (1.0.0)
Requirement already satisfied: piglet-templates in /usr/local/lib/python3.7/dist-packages (from
piglet) (1.2.0)
Requirement already satisfied: astunparse in /usr/local/lib/python3.7/dist-packages (from pigle
t-templates->piglet) (1.6.3)
Requirement already satisfied: pyparsing in /usr/local/lib/python3.7/dist-packages (from piglet
-templates->piglet) (3.0.6)
Requirement already satisfied: attrs in /usr/local/lib/python3.7/dist-packages (from piglet-tem
plates->piglet) (21.2.0)
Requirement already satisfied: markupsafe in /usr/local/lib/python3.7/dist-packages (from pigle
t-templates->piglet) (2.0.1)
Requirement already satisfied: wheel<1.0,>=0.23.0 in /usr/local/lib/python3.7/dist-packages (fr
om astunparse->piglet-templates->piglet) (0.37.0)
Requirement already satisfied: six<2.0,>=1.6.1 in /usr/local/lib/python3.7/dist-packages (from
astunparse->piglet-templates->piglet) (1.15.0)
```

In [53]:

```
import pyvirtualdisplay

from pyvirtualdisplay import Display
display = Display(visible=0, size=(1400, 900))
display.start()


!apt install xvfb -y
!pip install pyvirtualdisplay
!pip install piglet

from pyvirtualdisplay import Display
display = Display(visible=0, size=(1400, 900))
display.start()

#_display = pyvirtualdisplay.Display(visible=False,  # use False with Xvfb
#                                    size=(1400, 900))
#_ = _display.start()
```

```
Reading package lists... Done
Building dependency tree
Reading state information... Done
xvfb is already the newest version (2:1.19.6-1ubuntu4.9).
0 upgraded, 0 newly installed, 0 to remove and 58 not upgraded.
Requirement already satisfied: pyvirtualdisplay in /usr/local/lib/python3.7/dist-packages (2.2)
Requirement already satisfied: EasyProcess in /usr/local/lib/python3.7/dist-packages (from pyvi
rtualdisplay) (0.3)
Requirement already satisfied: piglet in /usr/local/lib/python3.7/dist-packages (1.0.0)
Requirement already satisfied: piglet-templates in /usr/local/lib/python3.7/dist-packages (from
piglet) (1.2.0)
Requirement already satisfied: pyparsing in /usr/local/lib/python3.7/dist-packages (from piglet
-templates->piglet) (3.0.6)
Requirement already satisfied: markupsafe in /usr/local/lib/python3.7/dist-packages (from pigle
t-templates->piglet) (2.0.1)
Requirement already satisfied: attrs in /usr/local/lib/python3.7/dist-packages (from piglet-tem
plates->piglet) (21.2.0)
Requirement already satisfied: astunparse in /usr/local/lib/python3.7/dist-packages (from pigle
```

```
t-templates->piglet) (1.6.3)
Requirement already satisfied: six<2.0,>=1.6.1 in /usr/local/lib/python3.7/dist-packages (from
astunparse->piglet-templates->piglet) (1.15.0)
Requirement already satisfied: wheel<1.0,>=0.23.0 in /usr/local/lib/python3.7/dist-packages (fr
om astunparse->piglet-templates->piglet) (0.37.0)
```

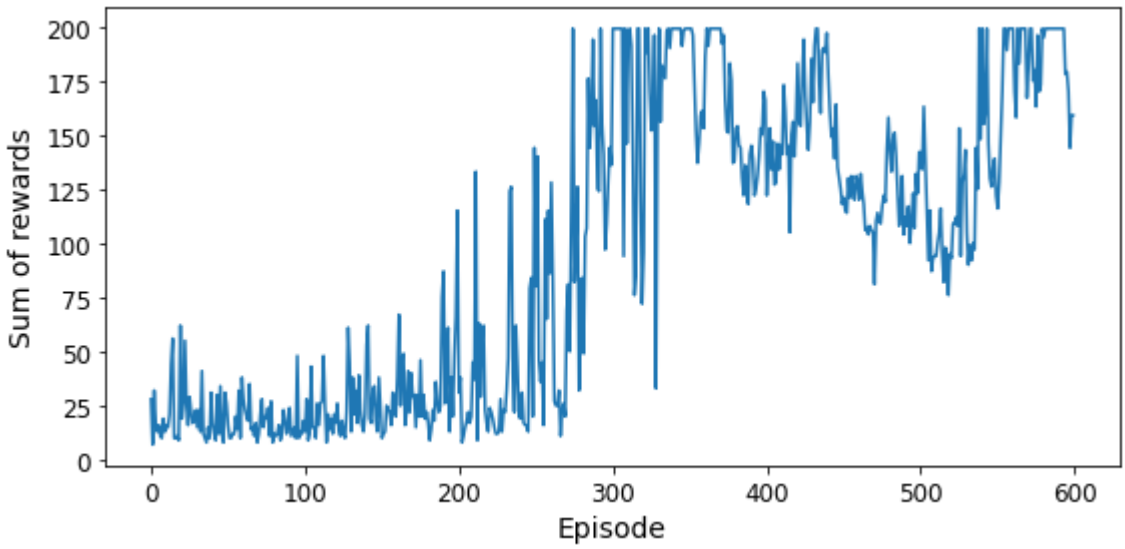
Out[53]:
```
<pyvirtualdisplay.display.Display at 0x7f1991963890>
```

In [54]:
```
for episode in range(600):
    obs = env.reset()
    for step in range(200):
        epsilon = max(1 - episode / 500, 0.01)
        obs, reward, done, info = play_one_step(env, obs, epsilon)
        if done:
            break
    rewards.append(step) # Not shown in the book
    if step >= best_score: # Not shown
        best_weights = model.get_weights() # Not shown
        best_score = step # Not shown
    print("\rEpisode: {}, Steps: {}, eps: {:.3f}".format(episode, step + 1, epsilon), end="") #
    if episode > 50:
        training_step(batch_size)

model.set_weights(best_weights)
```

```
Episode: 599, Steps: 160, eps: 0.010
```

In [60]:
```
plt.figure(figsize=(8, 4))
plt.plot(rewards)
print(max(rewards))
plt.xlabel("Episode", fontsize=14)
plt.ylabel("Sum of rewards", fontsize=14)
save_fig("dqn_rewards_plot")
plt.show()
```

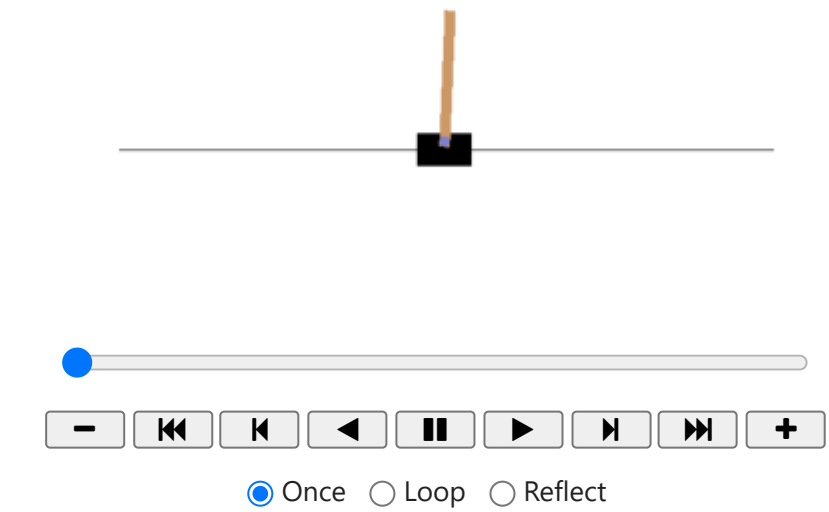
```
199
Saving figure dqn_rewards_plot
```

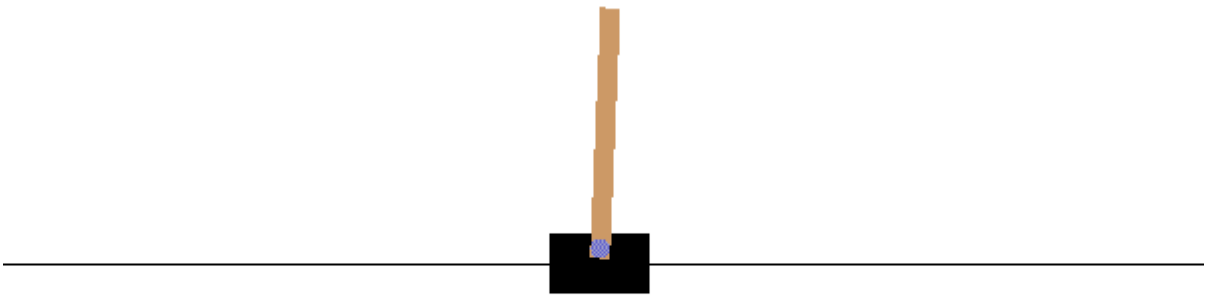


In [59]:
```
env.seed(42)
state = env.reset()

frames = []

for step in range(1000):
    action = epsilon_greedy_policy(state)
    state, reward, done, info = env.step(action)
    if done:
        break
    img = env.render(mode="rgb_array")
    frames.append(img)
```

```
plot_animation(frames)
```

Out[59]:





Double DQN

In [15]:

```
keras.backend.clear_session()
tf.random.set_seed(42)
np.random.seed(42)

model = keras.models.Sequential([
    keras.layers.Dense(32, activation="elu", input_shape=[4]),
    keras.layers.Dense(32, activation="elu"),
    keras.layers.Dense(n_outputs)
])

target = keras.models.clone_model(model)
target.set_weights(model.get_weights())
```

In [16]:

```
batch_size = 32
discount_rate = 0.95
optimizer = keras.optimizers.Adam(learning_rate=6e-3)
loss_fn = keras.losses.Huber()

def training_step(batch_size):
    experiences = sample_experiences(batch_size)
    states, actions, rewards, next_states, dones = experiences
    next_Q_values = model.predict(next_states)
    best_next_actions = np.argmax(next_Q_values, axis=1)
    next_mask = tf.one_hot(best_next_actions, n_outputs).numpy()
    next_best_Q_values = (target.predict(next_states) * next_mask).sum(axis=1)
    target_Q_values = (rewards +
                       (1 - dones) * discount_rate * next_best_Q_values)
    target_Q_values = target_Q_values.reshape(-1, 1)
    mask = tf.one_hot(actions, n_outputs)
    with tf.GradientTape() as tape:
        all_Q_values = model(states)
        Q_values = tf.reduce_sum(all_Q_values * mask, axis=1, keepdims=True)
        loss = tf.reduce_mean(loss_fn(target_Q_values, Q_values))
    grads = tape.gradient(loss, model.trainable_variables)
    optimizer.apply_gradients(zip(grads, model.trainable_variables))
```

In [17]:

```
replay_memory = deque(maxlen=2000)
```

In [18]:

```
env.seed(42)
np.random.seed(42)
tf.random.set_seed(42)

rewards = []
best_score = 0

for episode in range(600):
    obs = env.reset()
    for step in range(200):
        epsilon = max(1 - episode / 500, 0.01)
        obs, reward, done, info = play_one_step(env, obs, epsilon)
        if done:
            break
    rewards.append(step)
    if step >= best_score:
        best_weights = model.get_weights()
        best_score = step
    print("\rEpisode: {}, Steps: {}, eps: {:.3f}".format(episode, step + 1, epsilon), end="")
    if episode >= 50:
        training_step(batch_size)
        if episode % 50 == 0:
            target.set_weights(model.get_weights())
    # Alternatively, you can do soft updates at each step:
    #if episode >= 50:
        #target_weights = target.get_weights()
        #online_weights = model.get_weights()
        #for index in range(len(target_weights)):
        #    target_weights[index] = 0.99 * target_weights[index] + 0.01 * online_weights[index
        #target.set_weights(target_weights)

model.set_weights(best_weights)
```

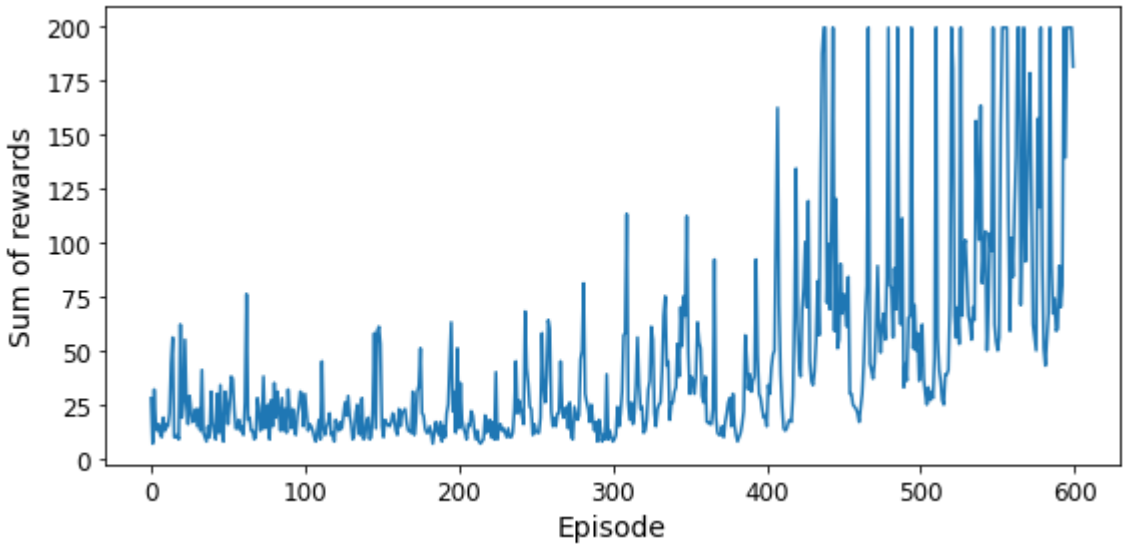
```
Episode: 599, Steps: 182, eps: 0.010
```

In [42]:

```
plt.figure(figsize=(8, 4))
plt.plot(rewards)
print(max(rewards))
plt.xlabel("Episode", fontsize=14)
```

```
plt.ylabel("Sum of rewards", fontsize=14)
save_fig("double_dqn_rewards_plot")
plt.show()
```

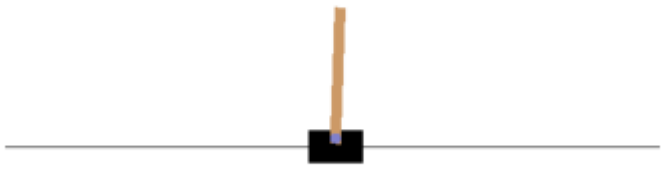
```
199
Saving figure double_dqn_rewards_plot
```

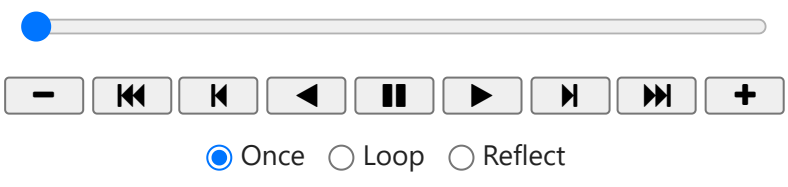


In [21]:

```
env.seed(43)
state = env.reset()

frames = []

for step in range(1000):
    action = epsilon_greedy_policy(state)
    state, reward, done, info = env.step(action)
    if done:
        break
    img = env.render(mode="rgb_array")
    frames.append(img)

plot_animation(frames)
```

Out[21]:





Once Loop Reflect