TensorForce - modular deep reinforcement learning in TensorFlow

TensorForce is an open source reinforcement learning library focused on providing clear APIs, readability and modularisation to deploy reinforcement learning solutions both in research and practice. TensorForce is built on top on TensorFlow.

Quick start

For a quick start, you can run one of our example scripts using the provided configurations, e.g. to run the TRPO agent on CartPole, execute from the examples folder:

```
python examples/openai_gym.py CartPole-v0 -a TRPOAgent -c examples/configs
/trpo_cartpole.json -n examples/configs/trpo_cartpole_network.json
```

In python, it could look like this:

```
# examples/quickstart.py

from tensorforce import Configuration
from tensorforce.agents import TRPOAgent
from tensorforce.environments.openai_gym import OpenAIGym
from tensorforce.execution import Runner
from tensorforce.core.networks import layered_network_builder

import numpy as np

# Create an OpenAIgym environment
env = OpenAIGym('CartPole-v0')

# Create a Trust Region Policy Optimization agent
agent = TRPOAgent(config=Configuration(
    log_level='info',
    batch_size=100,
    baseline=dict(
        type='mlp',
        size=32,
        repeat_update=100
    ),
    generalized_advantage_estimation=True,
    normalize_advantage=False,
    gae_lambda=0.97,
    override_line_search=False,
    cg_iterations=20,
    cg_damping=0.01,
    line_search_steps=20,
    max_kl_divergence=0.005,
    states=env.states,
    actions=env.actions,
    network=layered_network_builder([
        dict(type='dense', size=32, activation='tanh'),
        dict(type='dense', size=32, activation='tanh')
    ])
))

# Create the runner
runner = Runner(agent=agent, environment=env)


# Callback function printing episode statistics
def episode_finished(r):
    print("Finished episode {ep} after {ts} timesteps (reward:
{reward})".format(ep=r.episode, ts=r.timestep,

reward=r.episode_rewards[-1]))
    return True


# Start learning
runner.run(episodes=3000, max_timesteps=200, episode_finished=episode_finished)

# Print statistics
print("Learning finished. Total episodes: {ep}. Average reward of last 100 episodes:
{ar}.".format(ep=runner.episode,

ar=np.mean(

runner.episode_rewards[

-100:])))
```

Contents:

More information

You can find more information at our TensorForce GitHub repository.