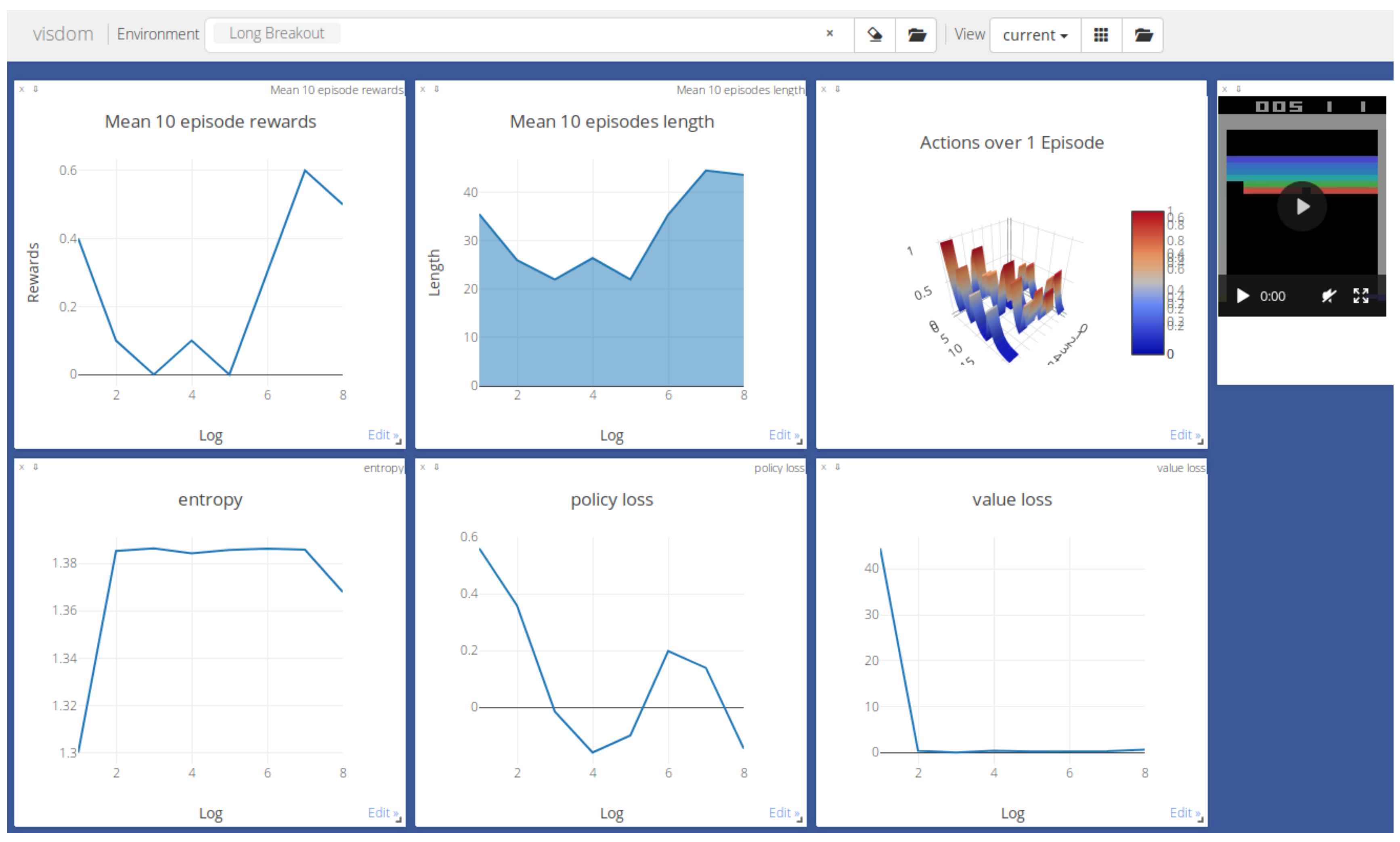
cherry

A Reinforcement Learning Library for Researchers

Summary

Cherry is a PyTorch library to help you write RL algorithms. Unlike many other RL libraries it only provides low-level utilities, no implementation provided! Instead, we strive to make it easy to implement any algorithm, whether from a textbook, paper, or your own mind. We provide many high-quality examples, and extensive documentation



```python
import cherry as ch

from cherry.algorithms import a2c
from cherry.models import atari

class NatureCNN(nn.Module):

    def __init__(self, env, hidden_size=512):
        super(NatureCNN, self).__init__()
        self.features = atari.NatureFeatures(env.state_size, hidden_size)
        self.critic = atari.NatureCritic(hidden_size)
        self.actor = atari.NatureActor(hidden_size, env.action_size)

    def forward(self, x):
        # compute forward
        return action, optional_info

ch.debug.debug()  # Enable debug-mode
env = gym.make('BreakoutNoFrameskip-v4')
env = envs.VisdomLogger(env, interval=1000)
env = envs.OpenAIAtari(env)
env = envs.Torch(env)
env = envs.Runner(env)
policy = NatureCNN(env)

for iteration in range(1e6):
    replay = env.run(get_action, steps=A2C_STEPS)

    # Compute advantages
    _, info = policy(replay[-1].next_state)
    rewards = ch.td.discount(0.99,
                             replay.reward(),
                             replay.done(),
                             bootstrap=info['value'])
    rewards = rewards.detach()
    advantages = rewards - replay.value().detach()

    # Compute loss
    entropy = replay.entropy().mean()
    policy_loss = a2c.policy_loss(replay.log_prob(), advantages)
    value_loss = a2c.state_value_loss(replay.value(), rewards)
    loss = policy_loss + 0.5 * value_loss - 0.01 * entropy

    # Take optimization step
    env.log('policy loss', policy_loss.item())
    opt.zero_grad()
    loss.backward()
    opt.step()
```

```
pip install cherry-rl
```

Debugging

Debugging RL is hard, that's why cherry comes with a set of debugging utilities.

- Debug mode: logging + `pdb` on `Exceptions`.
- Visdom dashboard with pre-defined monitoring.
- Defensive programming for algorithms utilities.
- Large set of high-quality examples.
- Unit and integration tests.

Compatibility

Cherry does not make assumptions on your software / hardware / algorithmic stack.

- Any environment. (Gym, dm_env, Lab, ALE)
- Any hardware. (CPU/GPU, parallel/distributed)
- Any algorithms. (tabular, deep, on-/off-policy)

If some setting is not supported, get in touch!

learnables/cherry

cherry-rl.net