transformer_PPO_frogfly

December 2, 2025

0.1 3.0 PPO + Transformer-Based Policy Network

To incorporate temporal context into PPO, we replace the MLP policy with a Transformer policy. Instead of conditioning only on the current state, the agent receives a short history of experience and attends to relevant past transitions.

0.1.1 Sequential Input Representation

At timestep (t), we construct a window of recent experience:

$$\mathcal{E}_t = [(s_{t-T+1}, a_{t-T+1}), \dots, (s_{t-1}, a_{t-1}), (s_t, a_t)]$$

Each element is embedded using a small MLP:

$$x_i = \text{MLP}_{\text{embed}}([s_i, a_i])$$

0.1.2 Temporal Positional Encoding

Because self-attention is order-invariant, we inject temporal indices using sinusoidal encodings:

$$PE(i) = [\sin(\omega_k i), \cos(\omega_k i)]_{k=1}^{F}$$

The input token for the Transformer becomes:

$$z_i = x_i \parallel PE(i)$$

0.1.3 Transformer Encoder Block

A Transformer layer processes the sequence by attending over past and recent experience:

$$H^{(l+1)} = \text{LN}\Big(H^{(l)} + \text{MHA}(H^{(l)}, H^{(l)}, H^{(l)})\Big)$$

$$H^{(l+1)} = \mathrm{LN}\Big(H^{(l)} + \mathrm{FFN}(H^{(l)})\Big)$$

The final hidden state (H_T) serves as a context summary for decision-making.

0.1.4 Policy & Value Outputs

$$\pi_\theta(a_t|\mathcal{E}_t) = \mathrm{Softmax}(\mathrm{MLP}_{\mathrm{policy}}(H_T))$$

$$V_\phi(\mathcal{E}_t) = \mathrm{MLP}_{\mathrm{value}}(H_T)$$

This allows PPO to act based on *recent experience*, not just a single observation.

[1]:
```
import numpy as np
import matplotlib.pyplot as plt
from matplotlib import animation
from mpl_toolkits.mplot3d import Axes3D  # noqa: F401 (needed for 3D)

import gymnasium as gym
from gymnasium import spaces

import ray
from ray.rllib.algorithms.ppo import PPOConfig
```

[2]:
```
def positional_encoding_3d(pos, num_freqs=4, max_freq=10.0):
    """
    pos: np.array shape (3,) with (x, y, z) in [-space_size, space_size].
    Returns: encoding vector of shape (3 * 2 * num_freqs,)
    """
    pos = np.asarray(pos, dtype=np.float32)
    assert pos.shape == (3,)

    freqs = np.linspace(1.0, max_freq, num_freqs)
    enc = []
    for coord in pos:            # x, y, z
        for f in freqs:
            enc.append(np.sin(f * coord))
            enc.append(np.cos(f * coord))
    return np.array(enc, dtype=np.float32)


class FrogFly3DEnv(gym.Env):
    """
    3D continuous environment:
    - Frog (agent) moves to catch a moving fly.
    - Observations: positions + relative vector + positional encodings.
```

```
    """
    metadata = {"render_modes": ["human"], "render_fps": 30}

    def __init__(self, config=None):
        super().__init__()

        if config is None:
            config = {}

        self.space_size = float(config.get("space_size", 1.0))
        self.step_size = float(config.get("step_size", 0.1))
        self.fly_speed = float(config.get("fly_speed", 0.05))
        self.catch_radius = float(config.get("catch_radius", 0.15))
        self.max_steps = int(config.get("max_steps", 200))
        self.num_freqs = int(config.get("num_freqs", 4))
        self.use_positional_encodings = bool(
            config.get("use_positional_encodings", True)
        )

        # Action = 3D movement in [-1, 1]^3
        self.action_space = spaces.Box(
            low=-1.0, high=1.0, shape=(3,), dtype=np.float32
        )

        # Observation components
        # frog_pos (3) + fly_pos (3) + rel (3) = 9
        base_dim = 9
        if self.use_positional_encodings:
            pe_dim = 3 * 2 * self.num_freqs  # per position
            obs_dim = base_dim + 2 * pe_dim
        else:
            obs_dim = base_dim

        self.observation_space = spaces.Box(
            low=-5.0, high=5.0, shape=(obs_dim,), dtype=np.float32
        )

        self.frog_pos = None
        self.fly_pos = None
        self.steps = 0
        self._value_out = None

    # ---------- Helpers ----------

    def _sample_position(self):
        return np.random.uniform(
            low=-self.space_size,
```

```
            high=self.space_size,
            size=(3,),
        ).astype(np.float32)

    def _clip_position(self, pos):
        return np.clip(pos, -self.space_size, self.space_size).astype(np.
↪float32)

    def _get_obs(self):
        rel = self.fly_pos - self.frog_pos
        parts = [self.frog_pos, self.fly_pos, rel]

        if self.use_positional_encodings:
            frog_pe = positional_encoding_3d(self.frog_pos, self.num_freqs)
            fly_pe = positional_encoding_3d(self.fly_pos, self.num_freqs)
            parts.extend([frog_pe, fly_pe])

        return np.concatenate(parts, axis=0).astype(np.float32)

    def _get_distance(self):
        return float(np.linalg.norm(self.fly_pos - self.frog_pos))

    # ---------- Gymnasium API ----------

    def reset(self, seed=None, options=None):
        super().reset(seed=seed)
        self.steps = 0
        self.frog_pos = self._sample_position()
        self.fly_pos = self._sample_position()

        obs = self._get_obs()
        info = {}
        return obs, info

    def step(self, action):
        self.steps += 1

        action = np.asarray(action, dtype=np.float32)
        action = np.clip(action, -1.0, 1.0)

        # -------------------------
        # Distance BEFORE movement
        # -------------------------
        prev_dist = self._get_distance()

        # -------------------------
        # Frog movement
```

```
        # -------------------------
        self.frog_pos = self.frog_pos + self.step_size * action
        self.frog_pos = self._clip_position(self.frog_pos)

        # -------------------------
        # Fly movement
        # -------------------------
        fly_dir = np.random.normal(size=(3,)).astype(np.float32)
        fly_dir /= np.linalg.norm(fly_dir) + 1e-8
        self.fly_pos = self.fly_pos + self.fly_speed * fly_dir
        self.fly_pos = self._clip_position(self.fly_pos)

        # -------------------------
        # AFTER movement
        # -------------------------
        dist = self._get_distance()
        caught = dist < self.catch_radius

        # -------------------------
        # BASE REWARD (distance improvement)
        # -------------------------
        dist_delta = prev_dist - dist
        reward = dist_delta

        # small time penalty to discourage wandering
        reward -= 0.01

        # -------------------------
        # ALIGNMENT BONUS (STRONG)
        # -------------------------
        direction = self.fly_pos - self.frog_pos
        dir_norm = np.linalg.norm(direction)
        act_norm = np.linalg.norm(action)

        if dir_norm > 1e-6 and act_norm > 1e-6:
            alignment = np.dot(action / act_norm, direction / dir_norm)
            reward += 0.4 * alignment   # <<< STRONG BUT NOT EXPLOSIVE

        # -------------------------
        # Catch bonus
        # -------------------------
        if caught:
            reward += 5.0
            terminated = True
        else:
            terminated = False
```

```
        truncated = self.steps >= self.max_steps

        obs = self._get_obs()
        info = {
            "distance": dist,
            "distance_delta": dist_delta,
            "caught": caught,
        }

        return obs, reward, terminated, truncated, info

    def render(self):
        print(
            f"Step {self.steps} | "
            f"Frog: {self.frog_pos} | Fly: {self.fly_pos} | "
            f"Dist: {self._get_distance():.3f}"
        )

    def close(self):
        pass
```

[3]:
```
from ray.tune.registry import register_env

# K-step observation history wrapper (if you don't already have this)
class ObsHistoryWrapper(gym.Wrapper):
    def __init__(self, env, K=24):
        super().__init__(env)
        self.K = K
        obs_shape = env.observation_space.shape  # (D,)
        self.history = np.zeros((K,) + obs_shape, dtype=np.float32)

        low  = np.repeat(env.observation_space.low[None, :],  K, axis=0)
        high = np.repeat(env.observation_space.high[None, :], K, axis=0)
        self.observation_space = gym.spaces.Box(low=low, high=high, dtype=np.
 ↪float32)

    def reset(self, **kwargs):
        obs, info = self.env.reset(**kwargs)
        self.history[:] = obs
        return self.history.copy(), info

    def step(self, action):
        obs, reward, terminated, truncated, info = self.env.step(action)
        self.history = np.roll(self.history, shift=-1, axis=0)
        self.history[-1] = obs
        return self.history.copy(), reward, terminated, truncated, info
```

```
from ray.tune.registry import register_env

def make_frog_hist_env(env_config):
    return ObsHistoryWrapper(FrogFly3DEnv(env_config), K=24)

register_env("frog_hist_env", make_frog_hist_env)

print("Registered frog_hist_env OK")
```

```
Registered frog_hist_env OK
```

[4]:
```
import numpy as np
import torch
import torch.nn as nn
from ray.rllib.models.torch.torch_modelv2 import TorchModelV2
from ray.rllib.models import ModelCatalog

ray.shutdown()
ray.init(ignore_reinit_error=True)

class FrogTransformerModel(TorchModelV2, nn.Module):
    def __init__(
        self,
        obs_space,
        action_space,
        num_outputs,
        model_config,
        name,
        **kwargs,
    ):
        TorchModelV2.__init__(self, obs_space, action_space, num_outputs,␣
 ↪model_config, name)
        nn.Module.__init__(self)

        # Expect obs shape = (K, obs_dim_single)
        assert len(obs_space.shape) == 2, f"Expected (K, D), got {obs_space.
 ↪shape}"
        self.seq_len    = obs_space.shape[0]  # K (time)
        self.feature_dim = obs_space.shape[1] # D (per-timestep features)

        cmc = model_config.get("custom_model_config", {}) or {}

        def get_hp(key, default):
            return kwargs.get(key, cmc.get(key, default))

        d_model    = get_hp("d_model", 64)
        nhead      = get_hp("nhead", 4)
```

```
        num_layers = get_hp("num_layers", 1)
        dim_ff     = get_hp("dim_feedforward", 128)
        dropout    = get_hp("dropout", 0.1)

        # Embedding per timestep (map R^{D} -> R^{d_model})
        self.embed = nn.Linear(self.feature_dim, d_model)

        encoder_layer = nn.TransformerEncoderLayer(
            d_model=d_model,
            nhead=nhead,
            dim_feedforward=dim_ff,
            dropout=dropout,
            batch_first=True,
            activation="relu",
        )
        self.transformer = nn.TransformerEncoder(
            encoder_layer,
            num_layers=num_layers,
        )

        self.policy_head = nn.Sequential(
            nn.Linear(d_model, d_model),
            nn.ReLU(),
            nn.Linear(d_model, num_outputs),
        )
        self.value_head = nn.Sequential(
            nn.Linear(d_model, d_model),
            nn.ReLU(),
            nn.Linear(d_model, 1),
        )

        self._features = None

    def forward(self, input_dict, state, seq_lens):
        # Get obs as float tensor, whatever its shape is
        x = input_dict["obs"].float()       # shape: (B, ...) with possibly >2␣
↪dims
        B = x.shape[0]

        # Flatten all non-batch dims into a single feature dimension
        x = x.view(B, -1)                   # shape: (B, F_flat)
        F = x.shape[1]

        # --- Infer how to interpret the obs as (B, K, D) sequence ---
        # Case 1: env gives a single feature vector of length feature_dim →␣
↪treat as K=1
        if F == self.feature_dim:
```

```
            K = 1
        # Case 2: env gives a flattened sequence of length seq_len * feature_dim
        elif F == self.seq_len * self.feature_dim:
            K = self.seq_len
        else:
            raise RuntimeError(
                f"Cannot reshape obs of size {F} into "
                f"(B, K, feature_dim={self.feature_dim}) with seq_len={self.
↪seq_len}."
            )

        # Reshape to (B, K, D)
        x = x.view(B, K, self.feature_dim)

        # --- Standard Transformer forward ---
        # (B, K, D) → (B, K, d_model)
        x = self.embed(x)

        # Optional: positional encodings, if you defined self.pos_embed
        if hasattr(self, "pos_embed"):
            # assume pos_embed has shape (1, max_K, d_model)
            x = x + self.pos_embed[:, :K, :]

        # TransformerEncoder expects (K, B, d_model)
        x = x.transpose(0, 1)           # (K, B, d_model)
        x = self.transformer(x)         # (K, B, d_model)
        x = x.transpose(0, 1)           # (B, K, d_model)

        # Pooling: take last token
        token = x[:, -1, :]             # (B, d_model)

        # Policy and value heads
        logits = self.policy_head(token)      # (B, num_actions)
        value  = self.value_head(token)       # (B, 1)

        # RLlib expects value via this attribute
        self._value_out = value.squeeze(-1)  # (B,)

        # Return logits + empty RNN state list
        return logits, []

    def value_function(self):
        # RLlib calls this after forward() to get V(s).
        if self._value_out is None:
            raise ValueError("value_function() called before forward().")
        return self._value_out
```

```
ModelCatalog.register_custom_model("frog_transformer_policy",␣
↪FrogTransformerModel)
```

```
2025-12-02 10:20:20,981 INFO worker.py:2023 -- Started a local Ray instance.
/home/platinumfish/miniconda3/envs/rl-project/lib/python3.10/site-
packages/ray/_private/worker.py:2062: FutureWarning: Tip: In future versions of
Ray, Ray will no longer override accelerator visible devices env var if
num_gpus=0 or num_gpus=None (default). To enable this behavior and turn off this
error message, set RAY_ACCEL_ENV_VAR_OVERRIDE_ON_ZERO=0
  warnings.warn(
```

[5]:
```
def make_frog_env(env_config):
    base_env = FrogFly3DEnv(env_config)
    return ObsHistoryWrapper(base_env, K=24)  # e.g., last 8 obs
```

[6]:
```
from ray.rllib.algorithms.ppo import PPOConfig

env_config = {
    "space_size": 1.0,
    "step_size": 0.1,
    "fly_speed": 0.05,
    "catch_radius": 0.15,
    "max_steps": 200,
    "num_freqs": 4,
    "use_positional_encodings": True,
}

config = (
    PPOConfig()
    .environment(
        env="frog_hist_env",
        env_config=env_config,
    )
    .framework("torch")
    .api_stack(
        enable_rl_module_and_learner=False,
        enable_env_runner_and_connector_v2=False,
    )
    .env_runners(
        num_env_runners=1,
        num_envs_per_env_runner=4,
        create_env_on_local_worker=True,
    )
    .resources(num_gpus=0)
    .training(
        model={
            "custom_model": "frog_transformer_policy",
            "custom_model_config": {
```

```
                "d_model": 128,          # stays moderate
                "nhead": 4,              # safe
                "num_layers": 2,         # avoids slowdown
                "dim_feedforward": 256, # reasonable
                "dropout": 0.05,         # reduce over-regularization
                "seq_len": 24,           # <<< MUST MATCH K
            },
        },
        gamma=0.99,

        # --- Optimization ---
        lr=3e-5,                # slightly smaller → more stability with longer␣
↪seq
        grad_clip=0.5,
        grad_clip_by="global_norm",

        # --- PPO batch settings ---
        train_batch_size=4000,  # increased slightly since seq_len grew
        num_epochs=5,
        minibatch_size=256,     # a bit larger for stability

        clip_param=0.2,

        # --- Exploration & value function ---
        entropy_coeff=0.005,    # lower than before; alignment reward already␣
↪guides
        vf_loss_coeff=1.0,

        # --- KL Regularization to prevent collapse ---
        use_kl_loss=True,
        kl_coeff=0.1,
        kl_target=0.02,
    )
)

algo = config.build()
print("Algo built OK with grad clipping")
```

```
2025-12-02 10:20:22,290 WARNING 2225844149.py:67 -- DeprecationWarning: `build`
has been deprecated. Use `AlgorithmConfig.build_algo` instead. This will raise
an error in the future!
/home/platinumfish/miniconda3/envs/rl-project/lib/python3.10/site-
packages/ray/rllib/algorithms/algorithm.py:525: RayDeprecationWarning: This API
is deprecated and may be removed in future Ray releases. You could suppress this
warning by setting env variable PYTHONWARNINGS="ignore::DeprecationWarning"
`UnifiedLogger` will be removed in Ray 2.7.
  return UnifiedLogger(config, logdir, loggers=None)
```

```
/home/platinumfish/miniconda3/envs/rl-project/lib/python3.10/site-
packages/ray/tune/logger/unified.py:53: RayDeprecationWarning: This API is
deprecated and may be removed in future Ray releases. You could suppress this
warning by setting env variable PYTHONWARNINGS="ignore::DeprecationWarning"
The `JsonLogger interface is deprecated in favor of the
`ray.tune.json.JsonLoggerCallback` interface and will be removed in Ray 2.7.
  self._loggers.append(cls(self.config, self.logdir, self.trial))
/home/platinumfish/miniconda3/envs/rl-project/lib/python3.10/site-
packages/ray/tune/logger/unified.py:53: RayDeprecationWarning: This API is
deprecated and may be removed in future Ray releases. You could suppress this
warning by setting env variable PYTHONWARNINGS="ignore::DeprecationWarning"
The `CSVLogger interface is deprecated in favor of the
`ray.tune.csv.CSVLoggerCallback` interface and will be removed in Ray 2.7.
  self._loggers.append(cls(self.config, self.logdir, self.trial))
/home/platinumfish/miniconda3/envs/rl-project/lib/python3.10/site-
packages/ray/tune/logger/unified.py:53: RayDeprecationWarning: This API is
deprecated and may be removed in future Ray releases. You could suppress this
warning by setting env variable PYTHONWARNINGS="ignore::DeprecationWarning"
The `TBXLogger interface is deprecated in favor of the
`ray.tune.tensorboardx.TBXLoggerCallback` interface and will be removed in Ray
2.7.
  self._loggers.append(cls(self.config, self.logdir, self.trial))
[2025-12-02 10:20:22,389 E 2877155 2877155] core_worker.cc:2223: Actor with
class name: 'RolloutWorker' and ID: 'c7ae1d9c4896d478b776496b01000000' has
constructor arguments in the object store and max_restarts > 0. If the arguments
in the object store go out of scope or are lost, the actor restart will fail.
See https://github.com/ray-project/ray/issues/53727 for more details.
2025-12-02 10:20:28,077 WARNING util.py:61 -- Install gputil for GPU system
monitoring.

Algo built OK with grad clipping
```

[7]:
```
res = algo.train()
print(res.keys())
print(res)
```

```
2025-12-02 10:20:34,223 WARNING train_ops.py:114 -- DeprecationWarning:
`ray.rllib.execution.train_ops.multi_gpu_train_one_step` has been deprecated.
This will raise an error in the future!
(pid=gcs_server) [2025-12-02 10:20:49,983 E 2877351 2877351]
(gcs_server) gcs_server.cc:303: Failed to establish connection to the
event+metrics exporter agent. Events and metrics will not be exported. Exporter
agent status: RpcError: Running out of retries to initialize the metrics agent.
rpc_code: 14
(raylet) [2025-12-02 10:20:50,920 E 2877486 2877486] (raylet)
main.cc:979: Failed to establish connection to the metrics exporter agent.
Metrics will not be exported. Exporter agent status: RpcError: Running out of
retries to initialize the metrics agent. rpc_code: 14
(RolloutWorker pid=2877532) [2025-12-02 10:20:51,813 E 2877532 2877600]
```

```
core_worker_process.cc:837: Failed to establish connection to the metrics
exporter agent. Metrics will not be exported. Exporter agent status: RpcError:
Running out of retries to initialize the metrics agent. rpc_code: 14
[2025-12-02 10:20:52,272 E 2877155 2877531] core_worker_process.cc:837: Failed
to establish connection to the metrics exporter agent. Metrics will not be
exported. Exporter agent status: RpcError: Running out of retries to initialize
the metrics agent. rpc_code: 14
```

```
dict_keys(['custom_metrics', 'episode_media', 'info', 'env_runners',
'num_healthy_workers', 'actor_manager_num_outstanding_async_reqs',
'num_remote_worker_restarts', 'num_agent_steps_sampled',
'num_agent_steps_trained', 'num_env_steps_sampled', 'num_env_steps_trained',
'num_env_steps_sampled_this_iter', 'num_env_steps_trained_this_iter',
'num_env_steps_sampled_throughput_per_sec',
'num_env_steps_trained_throughput_per_sec', 'timesteps_total',
'num_env_steps_sampled_lifetime', 'num_agent_steps_sampled_lifetime',
'num_steps_trained_this_iter', 'agent_timesteps_total', 'timers', 'counters',
'done', 'training_iteration', 'trial_id', 'date', 'timestamp',
'time_this_iter_s', 'time_total_s', 'pid', 'hostname', 'node_ip', 'config',
'time_since_restore', 'iterations_since_restore', 'perf'])
{'custom_metrics': {}, 'episode_media': {}, 'info': {'learner':
{'default_policy': {'learner_stats': {'allreduce_latency': np.float64(0.0),
'grad_gnorm': np.float64(0.5), 'cur_kl_coeff': np.float64(0.09999999999999998),
'cur_lr': np.float64(3.0000000000000004e-05), 'total_loss':
np.float64(6.05520734945933), 'policy_loss': np.float64(0.10211424574255944),
'vf_loss': np.float64(5.9482206439971925), 'vf_explained_var':
np.float64(0.003973019123077392), 'kl': np.float64(0.268071946700414),
'entropy': np.float64(4.386928730010986), 'entropy_coeff': np.float64(0.005)},
'model': {}, 'custom_metrics': {}, 'num_agent_steps_trained': np.float64(256.0),
'num_grad_updates_lifetime': np.float64(38.0),
'diff_num_grad_updates_vs_sampler_policy': np.float64(37.0)}},
'num_env_steps_sampled': 4000, 'num_env_steps_trained': 4000,
'num_agent_steps_sampled': 4000, 'num_agent_steps_trained': 4000},
'env_runners': {'episode_reward_max': np.float32(-1.9170334),
'episode_reward_min': np.float32(-21.739199), 'episode_reward_mean':
np.float32(-11.328579), 'episode_len_mean': np.float64(200.0), 'episode_media':
{}, 'episodes_timesteps_total': 4000, 'policy_reward_min': {'default_policy':
np.float32(-21.739199)}, 'policy_reward_max': {'default_policy':
np.float32(-1.9170334)}, 'policy_reward_mean': {'default_policy':
np.float32(-11.328579)}, 'custom_metrics': {}, 'hist_stats': {'episode_reward':
[np.float32(-10.068259), np.float32(-2.2923248), np.float32(-11.493873),
np.float32(-1.9170334), np.float32(-14.735265), np.float32(-12.608536),
np.float32(-10.447851), np.float32(-11.813146), np.float32(-11.668814),
np.float32(-13.332979), np.float32(-10.0151205), np.float32(-11.84692),
np.float32(-18.461256), np.float32(-9.083086), np.float32(-8.17567),
np.float32(-21.739199), np.float32(-13.774385), np.float32(-18.273848),
np.float32(-3.3205953), np.float32(-11.503409)], 'episode_lengths': [200, 200,
200, 200, 200, 200, 200, 200, 200, 200, 200, 200, 200, 200, 200, 200, 200,
```

200, 200], 'policy_default_policy_reward': [np.float32(-10.068259),
np.float32(-2.2923248), np.float32(-11.493873), np.float32(-1.9170334),
np.float32(-14.735265), np.float32(-12.608536), np.float32(-10.447851),
np.float32(-11.813146), np.float32(-11.668814), np.float32(-13.332979),
np.float32(-10.0151205), np.float32(-11.84692), np.float32(-18.461256),
np.float32(-9.083086), np.float32(-8.17567), np.float32(-21.739199),
np.float32(-13.774385), np.float32(-18.273848), np.float32(-3.3205953),
np.float32(-11.503409)]}, 'sampler_perf': {'mean_raw_obs_processing_ms':
np.float64(1.057621006008152), 'mean_inference_ms':
np.float64(3.753669016606563), 'mean_action_processing_ms':
np.float64(0.4608778805880399), 'mean_env_wait_ms':
np.float64(0.7460179266991552), 'mean_env_render_ms': np.float64(0.0)},
'num_faulty_episodes': 0, 'connector_metrics': {'ObsPreprocessorConnector_ms':
np.float64(0.007367134094238281), 'StateBufferConnector_ms':
np.float64(0.021663904190063477), 'ViewRequirementAgentConnector_ms':
np.float64(0.11044979095458984)}, 'num_episodes': 20, 'episode_return_max':
np.float32(-1.9170334), 'episode_return_min': np.float32(-21.739199),
'episode_return_mean': np.float32(-11.328579), 'episodes_this_iter': 20},
'num_healthy_workers': 1, 'actor_manager_num_outstanding_async_reqs': 0,
'num_remote_worker_restarts': 0, 'num_agent_steps_sampled': 4000,
'num_agent_steps_trained': 4000, 'num_env_steps_sampled': 4000,
'num_env_steps_trained': 4000, 'num_env_steps_sampled_this_iter': 4000,
'num_env_steps_trained_this_iter': 4000,
'num_env_steps_sampled_throughput_per_sec': 93.0544672671495,
'num_env_steps_trained_throughput_per_sec': 93.0544672671495, 'timesteps_total':
4000, 'num_env_steps_sampled_lifetime': 4000,
'num_agent_steps_sampled_lifetime': 4000, 'num_steps_trained_this_iter': 4000,
'agent_timesteps_total': 4000, 'timers': {'training_iteration_time_ms':
42985.593, 'restore_workers_time_ms': 0.027, 'training_step_time_ms': 42985.515,
'sample_time_ms': 6133.529, 'load_time_ms': 0.292, 'load_throughput':
13684515.498, 'learn_time_ms': 36845.523, 'learn_throughput': 108.561,
'synch_weights_time_ms': 4.97}, 'counters': {'num_env_steps_sampled': 4000,
'num_env_steps_trained': 4000, 'num_agent_steps_sampled': 4000,
'num_agent_steps_trained': 4000}, 'done': False, 'training_iteration': 1,
'trial_id': 'default', 'date': '2025-12-02_10-21-11', 'timestamp': 1764688871,
'time_this_iter_s': 42.99264144897461, 'time_total_s': 42.99264144897461, 'pid':
2877155, 'hostname': 'platinumfish-Aspire', 'node_ip': '192.168.86.52',
'config': {'exploration_config': {'type': 'StochasticSampling'},
'extra_python_environs_for_driver': {}, 'extra_python_environs_for_worker': {},
'placement_strategy': 'PACK', 'num_gpus': 0, '_fake_gpus': False,
'num_cpus_for_main_process': 1, 'eager_tracing': True, 'eager_max_retraces': 20,
'tf_session_args': {'intra_op_parallelism_threads': 2,
'inter_op_parallelism_threads': 2, 'gpu_options': {'allow_growth': True},
'log_device_placement': False, 'device_count': {'CPU': 1},
'allow_soft_placement': True}, 'local_tf_session_args':
{'intra_op_parallelism_threads': 8, 'inter_op_parallelism_threads': 8},
'torch_compile_learner': False, 'torch_compile_learner_what_to_compile':
<TorchCompileWhatToCompile.FORWARD_TRAIN: 'forward_train'>,

'torch_compile_learner_dynamo_backend': 'inductor',
'torch_compile_learner_dynamo_mode': None, 'torch_compile_worker': False,
'torch_compile_worker_dynamo_backend': 'onnxrt',
'torch_compile_worker_dynamo_mode': None, 'torch_ddp_kwargs': {},
'torch_skip_nan_gradients': False, 'env': 'frog_hist_env', 'env_config':
{'space_size': 1.0, 'step_size': 0.1, 'fly_speed': 0.05, 'catch_radius': 0.15,
'max_steps': 200, 'num_freqs': 4, 'use_positional_encodings': True},
'observation_space': None, 'action_space': None, 'clip_rewards': None,
'normalize_actions': True, 'clip_actions': False, '_is_atari': None,
'disable_env_checking': False, 'render_env': False, 'action_mask_key':
'action_mask', 'env_runner_cls': None, 'num_env_runners': 1,
'create_local_env_runner': True, 'num_envs_per_env_runner': 4,
'gym_env_vectorize_mode': 'sync', 'num_cpus_per_env_runner': 1,
'num_gpus_per_env_runner': 0, 'custom_resources_per_env_runner': {},
'validate_env_runners_after_construction': True, 'episodes_to_numpy': True,
'max_requests_in_flight_per_env_runner': 1, 'sample_timeout_s': 60.0,
'_env_to_module_connector': None,
'add_default_connectors_to_env_to_module_pipeline': True,
'_module_to_env_connector': None,
'add_default_connectors_to_module_to_env_pipeline': True,
'merge_env_runner_states': 'training_only', 'broadcast_env_runner_states': True,
'episode_lookback_horizon': 1, 'rollout_fragment_length': 'auto', 'batch_mode':
'truncate_episodes', 'compress_observations': False, 'remote_worker_envs':
False, 'remote_env_batch_wait_ms': 0, 'enable_tf1_exec_eagerly': False,
'sample_collector': <class
'ray.rllib.evaluation.collectors.simple_list_collector.SimpleListCollector'>,
'preprocessor_pref': 'deepmind', 'observation_filter': 'NoFilter',
'update_worker_filter_stats': True, 'use_worker_filter_stats': True,
'sampler_perf_stats_ema_coef': None, '_is_online': True, 'num_learners': 0,
'num_gpus_per_learner': 0, 'num_cpus_per_learner': 'auto',
'num_aggregator_actors_per_learner': 0,
'max_requests_in_flight_per_aggregator_actor': 3, 'local_gpu_idx': 0,
'max_requests_in_flight_per_learner': 3, 'gamma': 0.99, 'lr': 3e-05,
'grad_clip': 0.5, 'grad_clip_by': 'global_norm',
'_train_batch_size_per_learner': None, 'train_batch_size': 4000, 'num_epochs':
5, 'minibatch_size': 256, 'shuffle_batch_per_epoch': True, 'model':
{'fcnet_hiddens': [256, 256], 'fcnet_activation': 'tanh',
'fcnet_weights_initializer': None, 'fcnet_weights_initializer_config': None,
'fcnet_bias_initializer': None, 'fcnet_bias_initializer_config': None,
'conv_filters': None, 'conv_activation': 'relu', 'conv_kernel_initializer':
None, 'conv_kernel_initializer_config': None, 'conv_bias_initializer': None,
'conv_bias_initializer_config': None, 'conv_transpose_kernel_initializer': None,
'conv_transpose_kernel_initializer_config': None,
'conv_transpose_bias_initializer': None,
'conv_transpose_bias_initializer_config': None, 'post_fcnet_hiddens': [],
'post_fcnet_activation': 'relu', 'post_fcnet_weights_initializer': None,
'post_fcnet_weights_initializer_config': None, 'post_fcnet_bias_initializer':
None, 'post_fcnet_bias_initializer_config': None, 'free_log_std': False,

'log_std_clip_param': 20.0, 'no_final_linear': False, 'vf_share_layers': False,
'use_lstm': False, 'max_seq_len': 20, 'lstm_cell_size': 256,
'lstm_use_prev_action': False, 'lstm_use_prev_reward': False,
'lstm_weights_initializer': None, 'lstm_weights_initializer_config': None,
'lstm_bias_initializer': None, 'lstm_bias_initializer_config': None,
'_time_major': False, 'use_attention': False, 'attention_num_transformer_units':
1, 'attention_dim': 64, 'attention_num_heads': 1, 'attention_head_dim': 32,
'attention_memory_inference': 50, 'attention_memory_training': 50,
'attention_position_wise_mlp_dim': 32, 'attention_init_gru_gate_bias': 2.0,
'attention_use_n_prev_actions': 0, 'attention_use_n_prev_rewards': 0,
'framestack': True, 'dim': 84, 'grayscale': False, 'zero_mean': True,
'custom_model': 'frog_transformer_policy', 'custom_model_config': {'d_model':
128, 'nhead': 4, 'num_layers': 2, 'dim_feedforward': 256, 'dropout': 0.05,
'seq_len': 24}, 'custom_action_dist': None, 'custom_preprocessor': None,
'encoder_latent_dim': None, 'always_check_shapes': False,
'lstm_use_prev_action_reward': -1, '_use_default_native_models': -1,
'_disable_preprocessor_api': False, '_disable_action_flattening': False},
'_learner_connector': None, 'add_default_connectors_to_learner_pipeline': True,
'learner_config_dict': {}, 'optimizer': {}, '_learner_class': None,
'callbacks_on_algorithm_init': None, 'callbacks_on_env_runners_recreated': None,
'callbacks_on_offline_eval_runners_recreated': None,
'callbacks_on_checkpoint_loaded': None, 'callbacks_on_environment_created':
None, 'callbacks_on_episode_created': None, 'callbacks_on_episode_start': None,
'callbacks_on_episode_step': None, 'callbacks_on_episode_end': None,
'callbacks_on_evaluate_start': None, 'callbacks_on_evaluate_end': None,
'callbacks_on_evaluate_offline_start': None,
'callbacks_on_evaluate_offline_end': None, 'callbacks_on_sample_end': None,
'callbacks_on_train_result': None, 'explore': True,
'enable_rl_module_and_learner': False, 'enable_env_runner_and_connector_v2':
False, '_prior_exploration_config': None, 'count_steps_by': 'env_steps',
'policy_map_capacity': 100, 'policy_mapping_fn': <function
AlgorithmConfig.DEFAULT_POLICY_MAPPING_FN at 0x7881bb1395a0>,
'policies_to_train': None, 'policy_states_are_swappable': False,
'observation_fn': None, 'offline_data_class': None, 'input_read_method':
'read_parquet', 'input_read_method_kwargs': {}, 'input_read_schema': {},
'input_read_episodes': False, 'input_read_sample_batches': False,
'input_read_batch_size': None, 'input_filesystem': None,
'input_filesystem_kwargs': {}, 'input_compress_columns': ['obs', 'new_obs'],
'input_spaces_jsonable': True, 'materialize_data': False,
'materialize_mapped_data': True, 'map_batches_kwargs': {},
'iter_batches_kwargs': {}, 'ignore_final_observation': False,
'prelearner_class': None, 'prelearner_buffer_class': None,
'prelearner_buffer_kwargs': {}, 'prelearner_module_synch_period': 10,
'dataset_num_iters_per_learner': None, 'input_config': {},
'actions_in_input_normalized': False, 'postprocess_inputs': False,
'shuffle_buffer_size': 0, 'output': None, 'output_config': {},
'output_compress_columns': ['obs', 'new_obs'], 'output_max_file_size': 67108864,
'output_max_rows_per_file': None, 'output_write_remaining_data': False,

'output_write_method': 'write_parquet', 'output_write_method_kwargs': {},
'output_filesystem': None, 'output_filesystem_kwargs': {},
'output_write_episodes': True, 'offline_sampling': False, 'evaluation_interval':
None, 'evaluation_duration': 10, 'evaluation_duration_unit': 'episodes',
'evaluation_sample_timeout_s': 120.0,
'evaluation_auto_duration_min_env_steps_per_sample': 100,
'evaluation_auto_duration_max_env_steps_per_sample': 2000,
'evaluation_parallel_to_training': False,
'evaluation_force_reset_envs_before_iteration': True, 'evaluation_config': None,
'off_policy_estimation_methods': {}, 'ope_split_batch_by_episode': True,
'evaluation_num_env_runners': 0, 'in_evaluation': False,
'sync_filters_on_rollout_workers_timeout_s': 10.0,
'offline_evaluation_interval': None, 'num_offline_eval_runners': 0,
'offline_evaluation_type': None, 'offline_eval_runner_class': None,
'offline_loss_for_module_fn': None, 'offline_evaluation_duration': 1,
'offline_evaluation_parallel_to_training': False,
'offline_evaluation_timeout_s': 120.0, 'num_cpus_per_offline_eval_runner': 1,
'num_gpus_per_offline_eval_runner': 0,
'custom_resources_per_offline_eval_runner': {},
'restart_failed_offline_eval_runners': True,
'ignore_offline_eval_runner_failures': False,
'max_num_offline_eval_runner_restarts': 1000,
'offline_eval_runner_restore_timeout_s': 1800.0,
'max_requests_in_flight_per_offline_eval_runner': 1,
'validate_offline_eval_runners_after_construction': True,
'offline_eval_runner_health_probe_timeout_s': 30.0,
'offline_eval_rl_module_inference_only': False,
'broadcast_offline_eval_runner_states': False,
'offline_eval_batch_size_per_runner': 256, 'dataset_num_iters_per_eval_runner':
1, 'keep_per_episode_custom_metrics': False,
'metrics_episode_collection_timeout_s': 60.0,
'metrics_num_episodes_for_smoothing': 100, 'min_time_s_per_iteration': None,
'min_train_timesteps_per_iteration': 0, 'min_sample_timesteps_per_iteration': 0,
'log_gradients': False, 'export_native_model_files': False,
'checkpoint_trainable_policies_only': False, 'logger_creator': None,
'logger_config': None, 'log_level': 'WARN', 'log_sys_usage': True,
'fake_sampler': False, 'seed': None, 'restart_failed_env_runners': True,
'ignore_env_runner_failures': False, 'max_num_env_runner_restarts': 1000,
'delay_between_env_runner_restarts_s': 60.0, 'restart_failed_sub_environments':
False, 'num_consecutive_env_runner_failures_tolerance': 100,
'env_runner_health_probe_timeout_s': 30.0, 'env_runner_restore_timeout_s':
1800.0, '_model_config': {}, '_rl_module_spec': None,
'algorithm_config_overrides_per_module': {}, '_per_module_overrides': {},
'_validate_config': True, '_use_msgpack_checkpoints': False,
'_torch_grad_scaler_class': None, '_torch_lr_scheduler_classes': None,
'_tf_policy_handles_more_than_one_loss': False, '_disable_preprocessor_api':
False, '_disable_action_flattening': False,
'_disable_initialize_loss_from_dummy_batch': False,

```
'_dont_auto_sync_env_runner_states': False, 'env_task_fn': -1,
'enable_connectors': -1, 'simple_optimizer': False, 'policy_map_cache': -1,
'worker_cls': -1, 'synchronize_filters': -1, 'enable_async_evaluation': -1,
'custom_async_evaluation_function': -1, '_enable_rl_module_api': -1,
'auto_wrap_old_gym_envs': -1, 'always_attach_evaluation_results': -1,
'replay_sequence_length': None, '_disable_execution_plan_api': -1, 'use_critic':
True, 'use_gae': True, 'use_kl_loss': True, 'kl_coeff': 0.1, 'kl_target': 0.02,
'vf_loss_coeff': 1.0, 'entropy_coeff': 0.005, 'clip_param': 0.2,
'vf_clip_param': 10.0, 'entropy_coeff_schedule': None, 'lr_schedule': None,
'sgd_minibatch_size': -1, 'vf_share_layers': -1, 'class': <class
'ray.rllib.algorithms.ppo.ppo.PPOConfig'>, 'lambda': 1.0, 'input': 'sampler',
'policies': {'default_policy': (None, None, None, None)}, 'callbacks': <class
'ray.rllib.callbacks.callbacks.RLlibCallback'>, 'create_env_on_driver': True,
'custom_eval_function': None, 'framework': 'torch'}, 'time_since_restore':
42.99264144897461, 'iterations_since_restore': 1, 'perf': {'cpu_util_percent':
np.float64(57.30645161290322), 'ram_util_percent':
np.float64(78.19838709677418)}}
```

[8]:
```python
def get_train_metrics(result):
    """Extract mean reward and len from an RLlib result dict (env_runners␣
    ↪layout)."""
    er = result.get("env_runners", {})
    reward_mean = float(er.get("episode_return_mean", float("nan")))
    len_mean    = float(er.get("episode_len_mean", float("nan")))
    return reward_mean, len_mean
```

[9]:
```python
import numpy as np

eval_env = FrogFly3DEnv(env_config)  # reuse your env_config


def evaluate_policy(algo, env, num_episodes=5, render=False):
    """
    Evaluate a single-agent policy using the legacy RLlib API.

    Uses algo.compute_single_action(obs, ...) which is the right call
    for this RLlib version (compute_actions is multi-agent only here).
    """
    returns = []
    lengths = []

    for _ in range(num_episodes):
        obs, info = env.reset()
        done = False
        ep_ret = 0.0
        steps = 0
```

```
        while not done:
            # Single-agent action: returns (action, state_out, info)
            action, _, _ = algo.compute_single_action(obs, explore=False)

            obs, reward, terminated, truncated, info = env.step(action)
            done = terminated or truncated

            ep_ret += reward
            steps += 1

            if render:
                env.render()

        returns.append(ep_ret)
        lengths.append(steps)

    return {
        "mean_return": float(np.mean(returns)),
        "std_return": float(np.std(returns)),
        "mean_length": float(np.mean(lengths)),
    }
```

[10]:
```
def evaluate_catch_rate(algo, env_config, episodes=50):
    env = FrogFly3DEnv(env_config)
    catches = 0

    for _ in range(episodes):
        obs, info = env.reset()
        for _ in range(env.max_steps):
            action, _, _ = algo.compute_single_action(obs, explore=False)
            obs, reward, terminated, truncated, info = env.step(action)
            if terminated:
                catches += 1
                break
            if truncated:
                break

    env.close()
    return catches / episodes
```

[11]:
```
import os

num_iterations = 200        # how long you want to train
eval_interval  = 10         # how often to run eval
eval_episodes  = 5

train_history = []          # for plotting later
```

```python
best_eval_return = -float("inf")
checkpoint_dir = "frog_transformer_checkpoints"
os.makedirs(checkpoint_dir, exist_ok=True)

for i in range(1, num_iterations + 1):
    # ---- 1. One PPO training iteration ----
    result = algo.train()
    train_reward, train_len = get_train_metrics(result)

    # ---- 2. Periodic evaluation ----
    if i % eval_interval == 0:
        eval_stats = evaluate_policy(
            algo,
            eval_env,
            num_episodes=eval_episodes,
            render=False,
        )
        eval_return = eval_stats["mean_return"]
        eval_len    = eval_stats["mean_length"]

        # NEW - compute catch rate
        catch_rate = evaluate_catch_rate(algo, env_config, episodes=20)

        print(
            f"[Iter {i:4d}] "
            f"train_reward={train_reward:8.2f}, "
            f"train_len={train_len:6.1f} | "
            f"eval_return={eval_return:8.2f}, "
            f"eval_len={eval_len:6.1f} | "
            f"catch_rate={catch_rate*100:5.1f}%"
        )

        train_history.append(
            {
                "iter": i,
                "train_reward": train_reward,
                "train_len": train_len,
                "eval_return": eval_return,
                "eval_len": eval_len,
                "catch_rate": catch_rate,
            }
        )

        # save best checkpoint
        if eval_return > best_eval_return:
            best_eval_return = eval_return
            checkpoint_path = algo.save(checkpoint_dir)
```

```
            print(f"  New best eval_return={best_eval_return:.2f}, saved to␣
↪{checkpoint_path}")

    else:
        # Lighter log on non-eval iterations
        print(
            f"[Iter {i:4d}] "
            f"train_reward={train_reward:8.2f}, "
            f"train_len={train_len:6.1f}"
        )
```

```
[Iter    1] train_reward=   -9.92, train_len= 195.2
[Iter    2] train_reward=   -9.54, train_len= 194.6
[Iter    3] train_reward=   -8.72, train_len= 190.2
[Iter    4] train_reward=   -8.24, train_len= 190.5
[Iter    5] train_reward=   -7.36, train_len= 187.3
[Iter    6] train_reward=   -6.72, train_len= 187.9
[Iter    7] train_reward=   -5.91, train_len= 185.3
[Iter    8] train_reward=   -5.68, train_len= 187.7
[Iter    9] train_reward=   -5.07, train_len= 184.0
```

```
2025-12-02 10:27:54,663 WARNING 1093147107.py:24 -- DeprecationWarning:
`compute_single_action` has been deprecated. `Algorithm.compute_single_action`
should no longer be used. Get the RLModule instance through
`Algorithm.get_module([module ID])`, then compute actions through
`RLModule.forward_inference({'obs': [obs batch]})`. This will raise an error in
the future!
```

```
[Iter   10] train_reward=   -4.14, train_len= 179.9 | eval_return=   -5.87,
eval_len= 200.0 | catch_rate=  0.0%
  New best eval_return=-5.87, saved to
TrainingResult(checkpoint=Checkpoint(filesystem=local,
path=frog_transformer_checkpoints), metrics={'custom_metrics': {},
'episode_media': {}, 'info': {'learner': {'default_policy': {'learner_stats':
{'allreduce_latency': np.float64(0.0), 'grad_gnorm': np.float64(0.5),
'cur_kl_coeff': np.float64(0.15000000000000002), 'cur_lr':
np.float64(3.0000000000000004e-05), 'total_loss': np.float64(3.493915492693583),
'policy_loss': np.float64(0.007435679137706756), 'vf_loss':
np.float64(3.504380354086558), 'vf_explained_var':
np.float64(0.003544132709503174), 'kl': np.float64(0.019123016869028408),
'entropy': np.float64(4.153797000249227), 'entropy_coeff': np.float64(0.005)},
'model': {}, 'custom_metrics': {}, 'num_agent_steps_trained': np.float64(256.0),
'num_grad_updates_lifetime': np.float64(788.0),
'diff_num_grad_updates_vs_sampler_policy': np.float64(37.0)}},
'num_env_steps_sampled': 44000, 'num_env_steps_trained': 44000,
'num_agent_steps_sampled': 44000, 'num_agent_steps_trained': 44000},
'env_runners': {'episode_reward_max': np.float32(7.8953385),
'episode_reward_min': np.float32(-21.221718), 'episode_reward_mean':
np.float32(-4.1420884), 'episode_len_mean': np.float64(179.87), 'episode_media':
```

{}, 'episodes_timesteps_total': 17987, 'policy_reward_min': {'default_policy':
np.float32(-21.221718)}, 'policy_reward_max': {'default_policy':
np.float32(7.8953385)}, 'policy_reward_mean': {'default_policy':
np.float32(-4.1420884)}, 'custom_metrics': {}, 'hist_stats': {'episode_reward':
[np.float32(-5.333854), np.float32(-7.5229692), np.float32(4.86356),
np.float32(-2.394507), np.float32(-4.9925313), np.float32(-6.1192827),
np.float32(-7.9469237), np.float32(-17.92068), np.float32(-9.711579),
np.float32(-3.5139256), np.float32(-2.32665), np.float32(-3.9742377),
np.float32(-6.7835736), np.float32(-8.939408), np.float32(-5.8057094),
np.float32(-11.244027), np.float32(7.8953385), np.float32(-8.749355),
np.float32(7.365556), np.float32(-6.80255), np.float32(-2.98707),
np.float32(-2.3951805), np.float32(-12.897769), np.float32(5.6352777),
np.float32(-12.707662), np.float32(-9.392365), np.float32(4.5876827),
np.float32(-0.8114708), np.float32(-9.87548), np.float32(4.019109),
np.float32(-5.25603), np.float32(2.7619252), np.float32(-8.7367),
np.float32(-5.1336164), np.float32(-4.110524), np.float32(-6.8451595),
np.float32(-6.337323), np.float32(-2.7130795), np.float32(-4.046983),
np.float32(-1.4435819), np.float32(-4.7599473), np.float32(2.5153782),
np.float32(-5.512398), np.float32(5.7912273), np.float32(-9.356359),
np.float32(-10.9148245), np.float32(-2.501645), np.float32(-2.640996),
np.float32(-5.0415354), np.float32(-7.5822496), np.float32(-21.221718),
np.float32(-15.721521), np.float32(-13.346188), np.float32(-7.7297716),
np.float32(6.7310276), np.float32(4.3433633), np.float32(2.844969),
np.float32(-4.9264607), np.float32(5.6981907), np.float32(-4.0407844),
np.float32(3.764347), np.float32(-4.0583806), np.float32(-10.039052),
np.float32(-5.046557), np.float32(-1.9292572), np.float32(-3.4675503),
np.float32(-12.357096), np.float32(-14.515488), np.float32(-6.136729),
np.float32(-7.6341004), np.float32(-4.1208777), np.float32(-6.313548),
np.float32(-2.5114777), np.float32(3.96917), np.float32(5.0964675),
np.float32(-4.383825), np.float32(-1.5644035), np.float32(-3.6552215),
np.float32(7.7317247), np.float32(-6.799751), np.float32(7.6642885),
np.float32(-2.693195), np.float32(-0.5183748), np.float32(-9.443509),
np.float32(-9.418401), np.float32(-5.3627725), np.float32(7.3253493),
np.float32(-9.520279), np.float32(-1.9364073), np.float32(-12.697355),
np.float32(-0.12953758), np.float32(-0.24767828), np.float32(-7.083513),
np.float32(-7.3449454), np.float32(-2.9323153), np.float32(-6.9198403),
np.float32(-8.408646), np.float32(3.6087449), np.float32(-2.843304),
np.float32(-5.319968)], 'episode_lengths': [200, 200, 154, 200, 200, 200, 200,
200, 200, 200, 200, 200, 200, 200, 200, 200, 200, 129, 200, 147, 200, 200, 200, 200,
4, 200, 200, 178, 200, 200, 81, 200, 130, 200, 200, 200, 200, 200, 200, 200,
200, 200, 156, 200, 180, 200, 200, 200, 200, 200, 200, 200, 200, 200, 200, 125,
194, 158, 200, 119, 200, 9, 200, 200, 200, 200, 200, 200, 200, 200, 200, 200,
200, 200, 103, 8, 200, 159, 200, 77, 200, 145, 200, 200, 200, 200, 200, 133,
200, 200, 200, 100, 151, 200, 200, 139, 200, 200, 8, 200, 200],
'policy_default_policy_reward': [np.float32(-5.333854), np.float32(-7.5229692),
np.float32(4.86356), np.float32(-2.394507), np.float32(-4.9925313),
np.float32(-6.1192827), np.float32(-7.9469237), np.float32(-17.92068),
np.float32(-9.711579), np.float32(-3.5139256), np.float32(-2.32665),

np.float32(-3.9742377), np.float32(-6.7835736), np.float32(-8.939408),
np.float32(-5.8057094), np.float32(-11.244027), np.float32(7.8953385),
np.float32(-8.749355), np.float32(7.365556), np.float32(-6.80255),
np.float32(-2.98707), np.float32(-2.3951805), np.float32(-12.897769),
np.float32(5.6352777), np.float32(-12.707662), np.float32(-9.392365),
np.float32(4.5876827), np.float32(-0.8114708), np.float32(-9.87548),
np.float32(4.019109), np.float32(-5.25603), np.float32(2.7619252),
np.float32(-8.7367), np.float32(-5.1336164), np.float32(-4.110524),
np.float32(-6.8451595), np.float32(-6.337323), np.float32(-2.7130795),
np.float32(-4.046983), np.float32(-1.4435819), np.float32(-4.7599473),
np.float32(2.5153782), np.float32(-5.512398), np.float32(5.7912273),
np.float32(-9.356359), np.float32(-10.9148245), np.float32(-2.501645),
np.float32(-2.640996), np.float32(-5.0415354), np.float32(-7.5822496),
np.float32(-21.221718), np.float32(-15.721521), np.float32(-13.346188),
np.float32(-7.7297716), np.float32(6.7310276), np.float32(4.3433633),
np.float32(2.844969), np.float32(-4.9264607), np.float32(5.6981907),
np.float32(-4.0407844), np.float32(3.764347), np.float32(-4.0583806),
np.float32(-10.039052), np.float32(-5.046557), np.float32(-1.9292572),
np.float32(-3.4675503), np.float32(-12.357096), np.float32(-14.515488),
np.float32(-6.136729), np.float32(-7.6341004), np.float32(-4.1208777),
np.float32(-6.313548), np.float32(-2.5114777), np.float32(3.96917),
np.float32(5.0964675), np.float32(-4.383825), np.float32(-1.5644035),
np.float32(-3.6552215), np.float32(7.7317247), np.float32(-6.799751),
np.float32(7.6642885), np.float32(-2.693195), np.float32(-0.5183748),
np.float32(-9.443509), np.float32(-9.418401), np.float32(-5.3627725),
np.float32(7.3253493), np.float32(-9.520279), np.float32(-1.9364073),
np.float32(-12.697355), np.float32(-0.12953758), np.float32(-0.24767828),
np.float32(-7.083513), np.float32(-7.3449454), np.float32(-2.9323153),
np.float32(-6.9198403), np.float32(-8.408646), np.float32(3.6087449),
np.float32(-2.843304), np.float32(-5.319968)]}, 'sampler_perf':
{'mean_raw_obs_processing_ms': np.float64(1.099827607023784),
'mean_inference_ms': np.float64(3.7694244264022063),
'mean_action_processing_ms': np.float64(0.45184711084239515),
'mean_env_wait_ms': np.float64(0.7481703188681288), 'mean_env_render_ms':
np.float64(0.0)}, 'num_faulty_episodes': 0, 'connector_metrics':
{'ObsPreprocessorConnector_ms': np.float64(0.005155801773071289),
'StateBufferConnector_ms': np.float64(0.00392603874206543),
'ViewRequirementAgentConnector_ms': np.float64(0.09748435020446777)},
'num_episodes': 23, 'episode_return_max': np.float32(7.8953385),
'episode_return_min': np.float32(-21.221718), 'episode_return_mean':
np.float32(-4.1420884), 'episodes_this_iter': 23}, 'num_healthy_workers': 1,
'actor_manager_num_outstanding_async_reqs': 0, 'num_remote_worker_restarts': 0,
'num_agent_steps_sampled': 44000, 'num_agent_steps_trained': 44000,
'num_env_steps_sampled': 44000, 'num_env_steps_trained': 44000,
'num_env_steps_sampled_this_iter': 4000, 'num_env_steps_trained_this_iter':
4000, 'num_env_steps_sampled_throughput_per_sec': 98.90081212160662,
'num_env_steps_trained_throughput_per_sec': 98.90081212160662,
'timesteps_total': 44000, 'num_env_steps_sampled_lifetime': 44000,

'num_agent_steps_sampled_lifetime': 44000, 'num_steps_trained_this_iter': 4000,
'agent_timesteps_total': 44000, 'timers': {'training_iteration_time_ms':
40328.828, 'restore_workers_time_ms': 0.027, 'training_step_time_ms': 40328.758,
'sample_time_ms': 6186.942, 'load_time_ms': 0.736, 'load_throughput':
5437967.069, 'learn_time_ms': 34135.582, 'learn_throughput': 117.18,
'synch_weights_time_ms': 5.103}, 'counters': {'num_env_steps_sampled': 44000,
'num_env_steps_trained': 44000, 'num_agent_steps_sampled': 44000,
'num_agent_steps_trained': 44000}, 'done': False, 'training_iteration': 11,
'trial_id': 'default', 'date': '2025-12-02_10-27-54', 'timestamp': 1764689274,
'time_this_iter_s': 40.452364683151245, 'time_total_s': 446.34595942497253,
'pid': 2877155, 'hostname': 'platinumfish-Aspire', 'node_ip': '192.168.86.52',
'config': {'exploration_config': {'type': 'StochasticSampling'},
'extra_python_environs_for_driver': {}, 'extra_python_environs_for_worker': {},
'placement_strategy': 'PACK', 'num_gpus': 0, '_fake_gpus': False,
'num_cpus_for_main_process': 1, 'eager_tracing': True, 'eager_max_retraces': 20,
'tf_session_args': {'intra_op_parallelism_threads': 2,
'inter_op_parallelism_threads': 2, 'gpu_options': {'allow_growth': True},
'log_device_placement': False, 'device_count': {'CPU': 1},
'allow_soft_placement': True}, 'local_tf_session_args':
{'intra_op_parallelism_threads': 8, 'inter_op_parallelism_threads': 8},
'torch_compile_learner': False, 'torch_compile_learner_what_to_compile':
<TorchCompileWhatToCompile.FORWARD_TRAIN: 'forward_train'>,
'torch_compile_learner_dynamo_backend': 'inductor',
'torch_compile_learner_dynamo_mode': None, 'torch_compile_worker': False,
'torch_compile_worker_dynamo_backend': 'onnxrt',
'torch_compile_worker_dynamo_mode': None, 'torch_ddp_kwargs': {},
'torch_skip_nan_gradients': False, 'env': 'frog_hist_env', 'env_config':
{'space_size': 1.0, 'step_size': 0.1, 'fly_speed': 0.05, 'catch_radius': 0.15,
'max_steps': 200, 'num_freqs': 4, 'use_positional_encodings': True},
'observation_space': None, 'action_space': None, 'clip_rewards': None,
'normalize_actions': True, 'clip_actions': False, '_is_atari': None,
'disable_env_checking': False, 'render_env': False, 'action_mask_key':
'action_mask', 'env_runner_cls': None, 'num_env_runners': 1,
'create_local_env_runner': True, 'num_envs_per_env_runner': 4,
'gym_env_vectorize_mode': 'sync', 'num_cpus_per_env_runner': 1,
'num_gpus_per_env_runner': 0, 'custom_resources_per_env_runner': {},
'validate_env_runners_after_construction': True, 'episodes_to_numpy': True,
'max_requests_in_flight_per_env_runner': 1, 'sample_timeout_s': 60.0,
'_env_to_module_connector': None,
'add_default_connectors_to_env_to_module_pipeline': True,
'_module_to_env_connector': None,
'add_default_connectors_to_module_to_env_pipeline': True,
'merge_env_runner_states': 'training_only', 'broadcast_env_runner_states': True,
'episode_lookback_horizon': 1, 'rollout_fragment_length': 'auto', 'batch_mode':
'truncate_episodes', 'compress_observations': False, 'remote_worker_envs':
False, 'remote_env_batch_wait_ms': 0, 'enable_tf1_exec_eagerly': False,
'sample_collector': <class
'ray.rllib.evaluation.collectors.simple_list_collector.SimpleListCollector'>,

'preprocessor_pref': 'deepmind', 'observation_filter': 'NoFilter',
'update_worker_filter_stats': True, 'use_worker_filter_stats': True,
'sampler_perf_stats_ema_coef': None, '_is_online': True, 'num_learners': 0,
'num_gpus_per_learner': 0, 'num_cpus_per_learner': 'auto',
'num_aggregator_actors_per_learner': 0,
'max_requests_in_flight_per_aggregator_actor': 3, 'local_gpu_idx': 0,
'max_requests_in_flight_per_learner': 3, 'gamma': 0.99, 'lr': 3e-05,
'grad_clip': 0.5, 'grad_clip_by': 'global_norm',
'_train_batch_size_per_learner': None, 'train_batch_size': 4000, 'num_epochs':
5, 'minibatch_size': 256, 'shuffle_batch_per_epoch': True, 'model':
{'fcnet_hiddens': [256, 256], 'fcnet_activation': 'tanh',
'fcnet_weights_initializer': None, 'fcnet_weights_initializer_config': None,
'fcnet_bias_initializer': None, 'fcnet_bias_initializer_config': None,
'conv_filters': None, 'conv_activation': 'relu', 'conv_kernel_initializer':
None, 'conv_kernel_initializer_config': None, 'conv_bias_initializer': None,
'conv_bias_initializer_config': None, 'conv_transpose_kernel_initializer': None,
'conv_transpose_kernel_initializer_config': None,
'conv_transpose_bias_initializer': None,
'conv_transpose_bias_initializer_config': None, 'post_fcnet_hiddens': [],
'post_fcnet_activation': 'relu', 'post_fcnet_weights_initializer': None,
'post_fcnet_weights_initializer_config': None, 'post_fcnet_bias_initializer':
None, 'post_fcnet_bias_initializer_config': None, 'free_log_std': False,
'log_std_clip_param': 20.0, 'no_final_linear': False, 'vf_share_layers': False,
'use_lstm': False, 'max_seq_len': 20, 'lstm_cell_size': 256,
'lstm_use_prev_action': False, 'lstm_use_prev_reward': False,
'lstm_weights_initializer': None, 'lstm_weights_initializer_config': None,
'lstm_bias_initializer': None, 'lstm_bias_initializer_config': None,
'_time_major': False, 'use_attention': False, 'attention_num_transformer_units':
1, 'attention_dim': 64, 'attention_num_heads': 1, 'attention_head_dim': 32,
'attention_memory_inference': 50, 'attention_memory_training': 50,
'attention_position_wise_mlp_dim': 32, 'attention_init_gru_gate_bias': 2.0,
'attention_use_n_prev_actions': 0, 'attention_use_n_prev_rewards': 0,
'framestack': True, 'dim': 84, 'grayscale': False, 'zero_mean': True,
'custom_model': 'frog_transformer_policy', 'custom_model_config': {'d_model':
128, 'nhead': 4, 'num_layers': 2, 'dim_feedforward': 256, 'dropout': 0.05,
'seq_len': 24}, 'custom_action_dist': None, 'custom_preprocessor': None,
'encoder_latent_dim': None, 'always_check_shapes': False,
'lstm_use_prev_action_reward': -1, '_use_default_native_models': -1,
'_disable_preprocessor_api': False, '_disable_action_flattening': False},
'_learner_connector': None, 'add_default_connectors_to_learner_pipeline': True,
'learner_config_dict': {}, 'optimizer': {}, '_learner_class': None,
'callbacks_on_algorithm_init': None, 'callbacks_on_env_runners_recreated': None,
'callbacks_on_offline_eval_runners_recreated': None,
'callbacks_on_checkpoint_loaded': None, 'callbacks_on_environment_created':
None, 'callbacks_on_episode_created': None, 'callbacks_on_episode_start': None,
'callbacks_on_episode_step': None, 'callbacks_on_episode_end': None,
'callbacks_on_evaluate_start': None, 'callbacks_on_evaluate_end': None,
'callbacks_on_evaluate_offline_start': None,

'callbacks_on_evaluate_offline_end': None, 'callbacks_on_sample_end': None,
'callbacks_on_train_result': None, 'explore': True,
'enable_rl_module_and_learner': False, 'enable_env_runner_and_connector_v2':
False, '_prior_exploration_config': None, 'count_steps_by': 'env_steps',
'policy_map_capacity': 100, 'policy_mapping_fn': <function
AlgorithmConfig.DEFAULT_POLICY_MAPPING_FN at 0x7881bb1395a0>,
'policies_to_train': None, 'policy_states_are_swappable': False,
'observation_fn': None, 'offline_data_class': None, 'input_read_method':
'read_parquet', 'input_read_method_kwargs': {}, 'input_read_schema': {},
'input_read_episodes': False, 'input_read_sample_batches': False,
'input_read_batch_size': None, 'input_filesystem': None,
'input_filesystem_kwargs': {}, 'input_compress_columns': ['obs', 'new_obs'],
'input_spaces_jsonable': True, 'materialize_data': False,
'materialize_mapped_data': True, 'map_batches_kwargs': {},
'iter_batches_kwargs': {}, 'ignore_final_observation': False,
'prelearner_class': None, 'prelearner_buffer_class': None,
'prelearner_buffer_kwargs': {}, 'prelearner_module_synch_period': 10,
'dataset_num_iters_per_learner': None, 'input_config': {},
'actions_in_input_normalized': False, 'postprocess_inputs': False,
'shuffle_buffer_size': 0, 'output': None, 'output_config': {},
'output_compress_columns': ['obs', 'new_obs'], 'output_max_file_size': 67108864,
'output_max_rows_per_file': None, 'output_write_remaining_data': False,
'output_write_method': 'write_parquet', 'output_write_method_kwargs': {},
'output_filesystem': None, 'output_filesystem_kwargs': {},
'output_write_episodes': True, 'offline_sampling': False, 'evaluation_interval':
None, 'evaluation_duration': 10, 'evaluation_duration_unit': 'episodes',
'evaluation_sample_timeout_s': 120.0,
'evaluation_auto_duration_min_env_steps_per_sample': 100,
'evaluation_auto_duration_max_env_steps_per_sample': 2000,
'evaluation_parallel_to_training': False,
'evaluation_force_reset_envs_before_iteration': True, 'evaluation_config': None,
'off_policy_estimation_methods': {}, 'ope_split_batch_by_episode': True,
'evaluation_num_env_runners': 0, 'in_evaluation': False,
'sync_filters_on_rollout_workers_timeout_s': 10.0,
'offline_evaluation_interval': None, 'num_offline_eval_runners': 0,
'offline_evaluation_type': None, 'offline_eval_runner_class': None,
'offline_loss_for_module_fn': None, 'offline_evaluation_duration': 1,
'offline_evaluation_parallel_to_training': False,
'offline_evaluation_timeout_s': 120.0, 'num_cpus_per_offline_eval_runner': 1,
'num_gpus_per_offline_eval_runner': 0,
'custom_resources_per_offline_eval_runner': {},
'restart_failed_offline_eval_runners': True,
'ignore_offline_eval_runner_failures': False,
'max_num_offline_eval_runner_restarts': 1000,
'offline_eval_runner_restore_timeout_s': 1800.0,
'max_requests_in_flight_per_offline_eval_runner': 1,
'validate_offline_eval_runners_after_construction': True,
'offline_eval_runner_health_probe_timeout_s': 30.0,

'offline_eval_rl_module_inference_only': False,
'broadcast_offline_eval_runner_states': False,
'offline_eval_batch_size_per_runner': 256, 'dataset_num_iters_per_eval_runner':
1, 'keep_per_episode_custom_metrics': False,
'metrics_episode_collection_timeout_s': 60.0,
'metrics_num_episodes_for_smoothing': 100, 'min_time_s_per_iteration': None,
'min_train_timesteps_per_iteration': 0, 'min_sample_timesteps_per_iteration': 0,
'log_gradients': False, 'export_native_model_files': False,
'checkpoint_trainable_policies_only': False, 'logger_creator': None,
'logger_config': None, 'log_level': 'WARN', 'log_sys_usage': True,
'fake_sampler': False, 'seed': None, 'restart_failed_env_runners': True,
'ignore_env_runner_failures': False, 'max_num_env_runner_restarts': 1000,
'delay_between_env_runner_restarts_s': 60.0, 'restart_failed_sub_environments':
False, 'num_consecutive_env_runner_failures_tolerance': 100,
'env_runner_health_probe_timeout_s': 30.0, 'env_runner_restore_timeout_s':
1800.0, '_model_config': {}, '_rl_module_spec': None,
'algorithm_config_overrides_per_module': {}, '_per_module_overrides': {},
'_validate_config': True, '_use_msgpack_checkpoints': False,
'_torch_grad_scaler_class': None, '_torch_lr_scheduler_classes': None,
'_tf_policy_handles_more_than_one_loss': False, '_disable_preprocessor_api':
False, '_disable_action_flattening': False,
'_disable_initialize_loss_from_dummy_batch': False,
'_dont_auto_sync_env_runner_states': False, 'env_task_fn': -1,
'enable_connectors': -1, 'simple_optimizer': False, 'policy_map_cache': -1,
'worker_cls': -1, 'synchronize_filters': -1, 'enable_async_evaluation': -1,
'custom_async_evaluation_function': -1, '_enable_rl_module_api': -1,
'auto_wrap_old_gym_envs': -1, 'always_attach_evaluation_results': -1,
'replay_sequence_length': None, '_disable_execution_plan_api': -1, 'use_critic':
True, 'use_gae': True, 'use_kl_loss': True, 'kl_coeff': 0.1, 'kl_target': 0.02,
'vf_loss_coeff': 1.0, 'entropy_coeff': 0.005, 'clip_param': 0.2,
'vf_clip_param': 10.0, 'entropy_coeff_schedule': None, 'lr_schedule': None,
'sgd_minibatch_size': -1, 'vf_share_layers': -1, 'class': <class
'ray.rllib.algorithms.ppo.ppo.PPOConfig'>, 'lambda': 1.0, 'input': 'sampler',
'policies': {'default_policy': (None, None, None, None)}, 'callbacks': <class
'ray.rllib.callbacks.callbacks.RLlibCallback'>, 'create_env_on_driver': True,
'custom_eval_function': None, 'framework': 'torch'}, 'time_since_restore':
446.34595942497253, 'iterations_since_restore': 11, 'perf': {'cpu_util_percent':
np.float64(56.52105263157895), 'ram_util_percent':
np.float64(78.01228070175438)}})
[Iter 11] train_reward= -3.86, train_len= 176.7
[Iter 12] train_reward= -4.18, train_len= 179.3
[Iter 13] train_reward= -4.46, train_len= 182.1
[Iter 14] train_reward= -5.06, train_len= 186.8
[Iter 15] train_reward= -4.68, train_len= 184.9
[Iter 16] train_reward= -4.92, train_len= 187.6
[Iter 17] train_reward= -4.37, train_len= 183.7
[Iter 18] train_reward= -2.84, train_len= 173.0
[Iter 19] train_reward= -2.76, train_len= 172.8

```
[Iter   20] train_reward=   -2.22, train_len= 168.5 | eval_return=   -2.45,
eval_len= 200.0 | catch_rate=  0.0%
  New best eval_return=-2.45, saved to
TrainingResult(checkpoint=Checkpoint(filesystem=local,
path=frog_transformer_checkpoints), metrics={'custom_metrics': {},
'episode_media': {}, 'info': {'learner': {'default_policy': {'learner_stats':
{'allreduce_latency': np.float64(0.0), 'grad_gnorm': np.float64(0.5),
'cur_kl_coeff': np.float64(0.15000000000000002), 'cur_lr':
np.float64(3.0000000000000004e-05), 'total_loss':
np.float64(3.3154715132713317), 'policy_loss':
np.float64(-0.010020614316066106), 'vf_loss': np.float64(3.3426452414194743),
'vf_explained_var': np.float64(0.08238390366236369), 'kl':
np.float64(0.02189735772709052), 'entropy': np.float64(4.087542664210002),
'entropy_coeff': np.float64(0.005)}, 'model': {}, 'custom_metrics': {},
'num_agent_steps_trained': np.float64(256.0), 'num_grad_updates_lifetime':
np.float64(1538.0), 'diff_num_grad_updates_vs_sampler_policy':
np.float64(37.0)}}, 'num_env_steps_sampled': 84000, 'num_env_steps_trained':
84000, 'num_agent_steps_sampled': 84000, 'num_agent_steps_trained': 84000},
'env_runners': {'episode_reward_max': np.float32(9.1833935),
'episode_reward_min': np.float32(-13.165361), 'episode_reward_mean':
np.float32(-2.2206495), 'episode_len_mean': np.float64(168.52), 'episode_media':
{}, 'episodes_timesteps_total': 16852, 'policy_reward_min': {'default_policy':
np.float32(-13.165361)}, 'policy_reward_max': {'default_policy':
np.float32(9.1833935)}, 'policy_reward_mean': {'default_policy':
np.float32(-2.2206495)}, 'custom_metrics': {}, 'hist_stats': {'episode_reward':
[np.float32(2.538818), np.float32(-0.8406616), np.float32(-4.271262),
np.float32(-8.381506), np.float32(-5.0873876), np.float32(-1.4871317),
np.float32(4.2555933), np.float32(-6.7118793), np.float32(-6.5461125),
np.float32(-6.5250206), np.float32(-8.684255), np.float32(-4.442069),
np.float32(6.760786), np.float32(-5.6148996), np.float32(-3.6395888),
np.float32(-11.322632), np.float32(-10.424947), np.float32(5.271671),
np.float32(-8.271837), np.float32(8.259489), np.float32(-4.6714616),
np.float32(-11.726808), np.float32(-1.7148817), np.float32(-8.389935),
np.float32(0.13431415), np.float32(9.1833935), np.float32(-2.5561063),
np.float32(5.1780686), np.float32(7.711688), np.float32(-4.6557117),
np.float32(-5.161529), np.float32(8.6074705), np.float32(-7.6020727),
np.float32(-0.12470627), np.float32(1.690691), np.float32(4.8273215),
np.float32(-6.977495), np.float32(1.6267164), np.float32(-8.925802),
np.float32(-3.1253533), np.float32(-8.810638), np.float32(-3.42461),
np.float32(0.17936182), np.float32(-3.6511197), np.float32(4.001464),
np.float32(7.2582483), np.float32(3.2973187), np.float32(3.7054057),
np.float32(-4.409406), np.float32(-8.776671), np.float32(3.9593644),
np.float32(3.397308), np.float32(-7.5503807), np.float32(-3.0262043),
np.float32(5.4652967), np.float32(-7.4788365), np.float32(-5.097444),
np.float32(-5.8758993), np.float32(-2.9443853), np.float32(-6.5860643),
np.float32(-3.6791503), np.float32(5.0403585), np.float32(-3.533342),
np.float32(5.259043), np.float32(-6.760206), np.float32(-2.0618153),
np.float32(0.00053441525), np.float32(-7.3527613), np.float32(6.483295),
```

np.float32(-2.2602038), np.float32(-2.2616081), np.float32(-6.081985),
np.float32(-4.59378), np.float32(6.69173), np.float32(-13.165361),
np.float32(-5.9166126), np.float32(-4.2219653), np.float32(8.346106),
np.float32(2.1268399), np.float32(-9.0286045), np.float32(-9.582153),
np.float32(-8.138669), np.float32(-6.6707907), np.float32(1.1220101),
np.float32(-6.6276045), np.float32(-7.1571226), np.float32(3.5500755),
np.float32(4.488101), np.float32(-6.4207683), np.float32(-6.427759),
np.float32(-0.607273), np.float32(2.6402054), np.float32(-3.9054213),
np.float32(-5.2923975), np.float32(3.5199623), np.float32(-2.1829906),
np.float32(-4.9426384), np.float32(-9.144619), np.float32(-3.9730198),
np.float32(8.862352)], 'episode_lengths': [52, 200, 200, 200, 200, 200, 89, 200,
200, 200, 200, 200, 114, 200, 200, 200, 200, 98, 200, 142, 200, 200, 200, 200,
200, 157, 200, 37, 93, 200, 200, 61, 200, 194, 49, 82, 200, 115, 200, 200, 200,
200, 140, 200, 144, 148, 60, 200, 200, 200, 21, 147, 200, 200, 40, 200, 200,
200, 200, 200, 200, 120, 200, 32, 200, 200, 200, 200, 118, 200, 200, 200, 200,
57, 200, 200, 200, 92, 114, 200, 200, 200, 200, 200, 200, 200, 78, 78, 200, 200,
200, 53, 200, 200, 156, 200, 200, 200, 200, 171],
'policy_default_policy_reward': [np.float32(2.538818), np.float32(-0.8406616),
np.float32(-4.271262), np.float32(-8.381506), np.float32(-5.0873876),
np.float32(-1.4871317), np.float32(4.2555933), np.float32(-6.7118793),
np.float32(-6.5461125), np.float32(-6.5250206), np.float32(-8.684255),
np.float32(-4.442069), np.float32(6.760786), np.float32(-5.6148996),
np.float32(-3.6395888), np.float32(-11.322632), np.float32(-10.424947),
np.float32(5.271671), np.float32(-8.271837), np.float32(8.259489),
np.float32(-4.6714616), np.float32(-11.726808), np.float32(-1.7148817),
np.float32(-8.389935), np.float32(0.13431415), np.float32(9.1833935),
np.float32(-2.5561063), np.float32(5.1780686), np.float32(7.711688),
np.float32(-4.6557117), np.float32(-5.161529), np.float32(8.6074705),
np.float32(-7.6020727), np.float32(-0.12470627), np.float32(1.690691),
np.float32(4.8273215), np.float32(-6.977495), np.float32(1.6267164),
np.float32(-8.925802), np.float32(-3.1253533), np.float32(-8.810638),
np.float32(-3.42461), np.float32(0.17936182), np.float32(-3.6511197),
np.float32(4.001464), np.float32(7.2582483), np.float32(3.2973187),
np.float32(3.7054057), np.float32(-4.409406), np.float32(-8.776671),
np.float32(3.9593644), np.float32(3.397308), np.float32(-7.5503807),
np.float32(-3.0262043), np.float32(5.4652967), np.float32(-7.4788365),
np.float32(-5.097444), np.float32(-5.8758993), np.float32(-2.9443853),
np.float32(-6.5860643), np.float32(-3.6791503), np.float32(5.0403585),
np.float32(-3.533342), np.float32(5.259043), np.float32(-6.760206),
np.float32(-2.0618153), np.float32(0.00053441525), np.float32(-7.3527613),
np.float32(6.483295), np.float32(-2.2602038), np.float32(-2.2616081),
np.float32(-6.081985), np.float32(-4.59378), np.float32(6.69173),
np.float32(-13.165361), np.float32(-5.9166126), np.float32(-4.2219653),
np.float32(8.346106), np.float32(2.1268399), np.float32(-9.0286045),
np.float32(-9.582153), np.float32(-8.138669), np.float32(-6.6707907),
np.float32(1.1220101), np.float32(-6.6276045), np.float32(-7.1571226),
np.float32(3.5500755), np.float32(4.488101), np.float32(-6.4207683),
np.float32(-6.427759), np.float32(-0.607273), np.float32(2.6402054),

np.float32(-3.9054213), np.float32(-5.2923975), np.float32(3.5199623),
np.float32(-2.1829906), np.float32(-4.9426384), np.float32(-9.144619),
np.float32(-3.9730198), np.float32(8.862352)]}, 'sampler_perf':
{'mean_raw_obs_processing_ms': np.float64(1.1076413086444818),
'mean_inference_ms': np.float64(3.7776440132319498),
'mean_action_processing_ms': np.float64(0.45272295967509835),
'mean_env_wait_ms': np.float64(0.7489239974335652), 'mean_env_render_ms':
np.float64(0.0)}, 'num_faulty_episodes': 0, 'connector_metrics':
{'ObsPreprocessorConnector_ms': np.float64(0.0049898624420166016),
'StateBufferConnector_ms': np.float64(0.004492759704589844),
'ViewRequirementAgentConnector_ms': np.float64(0.09476470947265625)},
'num_episodes': 23, 'episode_return_max': np.float32(9.1833935),
'episode_return_min': np.float32(-13.165361), 'episode_return_mean':
np.float32(-2.2206495), 'episodes_this_iter': 23}, 'num_healthy_workers': 1,
'actor_manager_num_outstanding_async_reqs': 0, 'num_remote_worker_restarts': 0,
'num_agent_steps_sampled': 84000, 'num_agent_steps_trained': 84000,
'num_env_steps_sampled': 84000, 'num_env_steps_trained': 84000,
'num_env_steps_sampled_this_iter': 4000, 'num_env_steps_trained_this_iter':
4000, 'num_env_steps_sampled_throughput_per_sec': 100.72789118497066,
'num_env_steps_trained_throughput_per_sec': 100.72789118497066,
'timesteps_total': 84000, 'num_env_steps_sampled_lifetime': 84000,
'num_agent_steps_sampled_lifetime': 84000, 'num_steps_trained_this_iter': 4000,
'agent_timesteps_total': 84000, 'timers': {'training_iteration_time_ms':
40270.631, 'restore_workers_time_ms': 0.023, 'training_step_time_ms': 40270.562,
'sample_time_ms': 6174.612, 'load_time_ms': 0.924, 'load_throughput': 4328487.1,
'learn_time_ms': 34089.415, 'learn_throughput': 117.338,
'synch_weights_time_ms': 5.231}, 'counters': {'num_env_steps_sampled': 84000,
'num_env_steps_trained': 84000, 'num_agent_steps_sampled': 84000,
'num_agent_steps_trained': 84000}, 'done': False, 'training_iteration': 21,
'trial_id': 'default', 'date': '2025-12-02_10-34-49', 'timestamp': 1764689689,
'time_this_iter_s': 39.71819877624512, 'time_total_s': 849.1208851337433, 'pid':
2877155, 'hostname': 'platinumfish-Aspire', 'node_ip': '192.168.86.52',
'config': {'exploration_config': {'type': 'StochasticSampling'},
'extra_python_environs_for_driver': {}, 'extra_python_environs_for_worker': {},
'placement_strategy': 'PACK', 'num_gpus': 0, '_fake_gpus': False,
'num_cpus_for_main_process': 1, 'eager_tracing': True, 'eager_max_retraces': 20,
'tf_session_args': {'intra_op_parallelism_threads': 2,
'inter_op_parallelism_threads': 2, 'gpu_options': {'allow_growth': True},
'log_device_placement': False, 'device_count': {'CPU': 1},
'allow_soft_placement': True}, 'local_tf_session_args':
{'intra_op_parallelism_threads': 8, 'inter_op_parallelism_threads': 8},
'torch_compile_learner': False, 'torch_compile_learner_what_to_compile':
<TorchCompileWhatToCompile.FORWARD_TRAIN: 'forward_train'>,
'torch_compile_learner_dynamo_backend': 'inductor',
'torch_compile_learner_dynamo_mode': None, 'torch_compile_worker': False,
'torch_compile_worker_dynamo_backend': 'onnxrt',
'torch_compile_worker_dynamo_mode': None, 'torch_ddp_kwargs': {},
'torch_skip_nan_gradients': False, 'env': 'frog_hist_env', 'env_config':

{'space_size': 1.0, 'step_size': 0.1, 'fly_speed': 0.05, 'catch_radius': 0.15,
'max_steps': 200, 'num_freqs': 4, 'use_positional_encodings': True},
'observation_space': None, 'action_space': None, 'clip_rewards': None,
'normalize_actions': True, 'clip_actions': False, '_is_atari': None,
'disable_env_checking': False, 'render_env': False, 'action_mask_key':
'action_mask', 'env_runner_cls': None, 'num_env_runners': 1,
'create_local_env_runner': True, 'num_envs_per_env_runner': 4,
'gym_env_vectorize_mode': 'sync', 'num_cpus_per_env_runner': 1,
'num_gpus_per_env_runner': 0, 'custom_resources_per_env_runner': {},
'validate_env_runners_after_construction': True, 'episodes_to_numpy': True,
'max_requests_in_flight_per_env_runner': 1, 'sample_timeout_s': 60.0,
'_env_to_module_connector': None,
'add_default_connectors_to_env_to_module_pipeline': True,
'_module_to_env_connector': None,
'add_default_connectors_to_module_to_env_pipeline': True,
'merge_env_runner_states': 'training_only', 'broadcast_env_runner_states': True,
'episode_lookback_horizon': 1, 'rollout_fragment_length': 'auto', 'batch_mode':
'truncate_episodes', 'compress_observations': False, 'remote_worker_envs':
False, 'remote_env_batch_wait_ms': 0, 'enable_tf1_exec_eagerly': False,
'sample_collector': <class
'ray.rllib.evaluation.collectors.simple_list_collector.SimpleListCollector'>,
'preprocessor_pref': 'deepmind', 'observation_filter': 'NoFilter',
'update_worker_filter_stats': True, 'use_worker_filter_stats': True,
'sampler_perf_stats_ema_coef': None, '_is_online': True, 'num_learners': 0,
'num_gpus_per_learner': 0, 'num_cpus_per_learner': 'auto',
'num_aggregator_actors_per_learner': 0,
'max_requests_in_flight_per_aggregator_actor': 3, 'local_gpu_idx': 0,
'max_requests_in_flight_per_learner': 3, 'gamma': 0.99, 'lr': 3e-05,
'grad_clip': 0.5, 'grad_clip_by': 'global_norm',
'_train_batch_size_per_learner': None, 'train_batch_size': 4000, 'num_epochs':
5, 'minibatch_size': 256, 'shuffle_batch_per_epoch': True, 'model':
{'fcnet_hiddens': [256, 256], 'fcnet_activation': 'tanh',
'fcnet_weights_initializer': None, 'fcnet_weights_initializer_config': None,
'fcnet_bias_initializer': None, 'fcnet_bias_initializer_config': None,
'conv_filters': None, 'conv_activation': 'relu', 'conv_kernel_initializer':
None, 'conv_kernel_initializer_config': None, 'conv_bias_initializer': None,
'conv_bias_initializer_config': None, 'conv_transpose_kernel_initializer': None,
'conv_transpose_kernel_initializer_config': None,
'conv_transpose_bias_initializer': None,
'conv_transpose_bias_initializer_config': None, 'post_fcnet_hiddens': [],
'post_fcnet_activation': 'relu', 'post_fcnet_weights_initializer': None,
'post_fcnet_weights_initializer_config': None, 'post_fcnet_bias_initializer':
None, 'post_fcnet_bias_initializer_config': None, 'free_log_std': False,
'log_std_clip_param': 20.0, 'no_final_linear': False, 'vf_share_layers': False,
'use_lstm': False, 'max_seq_len': 20, 'lstm_cell_size': 256,
'lstm_use_prev_action': False, 'lstm_use_prev_reward': False,
'lstm_weights_initializer': None, 'lstm_weights_initializer_config': None,
'lstm_bias_initializer': None, 'lstm_bias_initializer_config': None,

'_time_major': False, 'use_attention': False, 'attention_num_transformer_units':
1, 'attention_dim': 64, 'attention_num_heads': 1, 'attention_head_dim': 32,
'attention_memory_inference': 50, 'attention_memory_training': 50,
'attention_position_wise_mlp_dim': 32, 'attention_init_gru_gate_bias': 2.0,
'attention_use_n_prev_actions': 0, 'attention_use_n_prev_rewards': 0,
'framestack': True, 'dim': 84, 'grayscale': False, 'zero_mean': True,
'custom_model': 'frog_transformer_policy', 'custom_model_config': {'d_model':
128, 'nhead': 4, 'num_layers': 2, 'dim_feedforward': 256, 'dropout': 0.05,
'seq_len': 24}, 'custom_action_dist': None, 'custom_preprocessor': None,
'encoder_latent_dim': None, 'always_check_shapes': False,
'lstm_use_prev_action_reward': -1, '_use_default_native_models': -1,
'_disable_preprocessor_api': False, '_disable_action_flattening': False},
'_learner_connector': None, 'add_default_connectors_to_learner_pipeline': True,
'learner_config_dict': {}, 'optimizer': {}, '_learner_class': None,
'callbacks_on_algorithm_init': None, 'callbacks_on_env_runners_recreated': None,
'callbacks_on_offline_eval_runners_recreated': None,
'callbacks_on_checkpoint_loaded': None, 'callbacks_on_environment_created':
None, 'callbacks_on_episode_created': None, 'callbacks_on_episode_start': None,
'callbacks_on_episode_step': None, 'callbacks_on_episode_end': None,
'callbacks_on_evaluate_start': None, 'callbacks_on_evaluate_end': None,
'callbacks_on_evaluate_offline_start': None,
'callbacks_on_evaluate_offline_end': None, 'callbacks_on_sample_end': None,
'callbacks_on_train_result': None, 'explore': True,
'enable_rl_module_and_learner': False, 'enable_env_runner_and_connector_v2':
False, '_prior_exploration_config': None, 'count_steps_by': 'env_steps',
'policy_map_capacity': 100, 'policy_mapping_fn': <function
AlgorithmConfig.DEFAULT_POLICY_MAPPING_FN at 0x7881bb1395a0>,
'policies_to_train': None, 'policy_states_are_swappable': False,
'observation_fn': None, 'offline_data_class': None, 'input_read_method':
'read_parquet', 'input_read_method_kwargs': {}, 'input_read_schema': {},
'input_read_episodes': False, 'input_read_sample_batches': False,
'input_read_batch_size': None, 'input_filesystem': None,
'input_filesystem_kwargs': {}, 'input_compress_columns': ['obs', 'new_obs'],
'input_spaces_jsonable': True, 'materialize_data': False,
'materialize_mapped_data': True, 'map_batches_kwargs': {},
'iter_batches_kwargs': {}, 'ignore_final_observation': False,
'prelearner_class': None, 'prelearner_buffer_class': None,
'prelearner_buffer_kwargs': {}, 'prelearner_module_synch_period': 10,
'dataset_num_iters_per_learner': None, 'input_config': {},
'actions_in_input_normalized': False, 'postprocess_inputs': False,
'shuffle_buffer_size': 0, 'output': None, 'output_config': {},
'output_compress_columns': ['obs', 'new_obs'], 'output_max_file_size': 67108864,
'output_max_rows_per_file': None, 'output_write_remaining_data': False,
'output_write_method': 'write_parquet', 'output_write_method_kwargs': {},
'output_filesystem': None, 'output_filesystem_kwargs': {},
'output_write_episodes': True, 'offline_sampling': False, 'evaluation_interval':
None, 'evaluation_duration': 10, 'evaluation_duration_unit': 'episodes',
'evaluation_sample_timeout_s': 120.0,

'evaluation_auto_duration_min_env_steps_per_sample': 100,
'evaluation_auto_duration_max_env_steps_per_sample': 2000,
'evaluation_parallel_to_training': False,
'evaluation_force_reset_envs_before_iteration': True, 'evaluation_config': None,
'off_policy_estimation_methods': {}, 'ope_split_batch_by_episode': True,
'evaluation_num_env_runners': 0, 'in_evaluation': False,
'sync_filters_on_rollout_workers_timeout_s': 10.0,
'offline_evaluation_interval': None, 'num_offline_eval_runners': 0,
'offline_evaluation_type': None, 'offline_eval_runner_class': None,
'offline_loss_for_module_fn': None, 'offline_evaluation_duration': 1,
'offline_evaluation_parallel_to_training': False,
'offline_evaluation_timeout_s': 120.0, 'num_cpus_per_offline_eval_runner': 1,
'num_gpus_per_offline_eval_runner': 0,
'custom_resources_per_offline_eval_runner': {},
'restart_failed_offline_eval_runners': True,
'ignore_offline_eval_runner_failures': False,
'max_num_offline_eval_runner_restarts': 1000,
'offline_eval_runner_restore_timeout_s': 1800.0,
'max_requests_in_flight_per_offline_eval_runner': 1,
'validate_offline_eval_runners_after_construction': True,
'offline_eval_runner_health_probe_timeout_s': 30.0,
'offline_eval_rl_module_inference_only': False,
'broadcast_offline_eval_runner_states': False,
'offline_eval_batch_size_per_runner': 256, 'dataset_num_iters_per_eval_runner':
1, 'keep_per_episode_custom_metrics': False,
'metrics_episode_collection_timeout_s': 60.0,
'metrics_num_episodes_for_smoothing': 100, 'min_time_s_per_iteration': None,
'min_train_timesteps_per_iteration': 0, 'min_sample_timesteps_per_iteration': 0,
'log_gradients': False, 'export_native_model_files': False,
'checkpoint_trainable_policies_only': False, 'logger_creator': None,
'logger_config': None, 'log_level': 'WARN', 'log_sys_usage': True,
'fake_sampler': False, 'seed': None, 'restart_failed_env_runners': True,
'ignore_env_runner_failures': False, 'max_num_env_runner_restarts': 1000,
'delay_between_env_runner_restarts_s': 60.0, 'restart_failed_sub_environments':
False, 'num_consecutive_env_runner_failures_tolerance': 100,
'env_runner_health_probe_timeout_s': 30.0, 'env_runner_restore_timeout_s':
1800.0, '_model_config': {}, '_rl_module_spec': None,
'algorithm_config_overrides_per_module': {}, '_per_module_overrides': {},
'_validate_config': True, '_use_msgpack_checkpoints': False,
'_torch_grad_scaler_class': None, '_torch_lr_scheduler_classes': None,
'_tf_policy_handles_more_than_one_loss': False, '_disable_preprocessor_api':
False, '_disable_action_flattening': False,
'_disable_initialize_loss_from_dummy_batch': False,
'_dont_auto_sync_env_runner_states': False, 'env_task_fn': -1,
'enable_connectors': -1, 'simple_optimizer': False, 'policy_map_cache': -1,
'worker_cls': -1, 'synchronize_filters': -1, 'enable_async_evaluation': -1,
'custom_async_evaluation_function': -1, '_enable_rl_module_api': -1,
'auto_wrap_old_gym_envs': -1, 'always_attach_evaluation_results': -1,

```
'replay_sequence_length': None, '_disable_execution_plan_api': -1, 'use_critic':
True, 'use_gae': True, 'use_kl_loss': True, 'kl_coeff': 0.1, 'kl_target': 0.02,
'vf_loss_coeff': 1.0, 'entropy_coeff': 0.005, 'clip_param': 0.2,
'vf_clip_param': 10.0, 'entropy_coeff_schedule': None, 'lr_schedule': None,
'sgd_minibatch_size': -1, 'vf_share_layers': -1, 'class': <class
'ray.rllib.algorithms.ppo.ppo.PPOConfig'>, 'lambda': 1.0, 'input': 'sampler',
'policies': {'default_policy': (None, None, None, None)}, 'callbacks': <class
'ray.rllib.callbacks.callbacks.RLlibCallback'>, 'create_env_on_driver': True,
'custom_eval_function': None, 'framework': 'torch'}, 'time_since_restore':
849.1208851337433, 'iterations_since_restore': 21, 'perf': {'cpu_util_percent':
np.float64(56.83684210526315), 'ram_util_percent':
np.float64(78.67894736842105)}})
[Iter   21] train_reward=   -1.69, train_len= 166.2
[Iter   22] train_reward=   -2.09, train_len= 169.6
[Iter   23] train_reward=   -1.91, train_len= 167.1
[Iter   24] train_reward=   -1.92, train_len= 170.4
[Iter   25] train_reward=   -1.48, train_len= 168.5
[Iter   26] train_reward=   -1.19, train_len= 167.6
[Iter   27] train_reward=   -0.94, train_len= 167.3
[Iter   28] train_reward=   -0.98, train_len= 166.8
[Iter   29] train_reward=   -1.16, train_len= 167.3
[Iter   30] train_reward=   -0.99, train_len= 158.1 | eval_return=   -6.40,
eval_len= 200.0 | catch_rate=  5.0%
[Iter   31] train_reward=   -0.85, train_len= 153.1
[Iter   32] train_reward=   -0.21, train_len= 149.5
[Iter   33] train_reward=   -0.14, train_len= 148.4
[Iter   34] train_reward=    0.36, train_len= 145.8
[Iter   35] train_reward=    0.40, train_len= 145.3
[Iter   36] train_reward=    0.82, train_len= 142.3
[Iter   37] train_reward=    1.11, train_len= 140.9
[Iter   38] train_reward=    1.27, train_len= 142.5
[Iter   39] train_reward=    1.74, train_len= 143.8
[Iter   40] train_reward=    1.63, train_len= 144.6 | eval_return=   -7.79,
eval_len= 200.0 | catch_rate=  0.0%
[Iter   41] train_reward=    1.28, train_len= 150.4
[Iter   42] train_reward=    1.66, train_len= 149.0
[Iter   43] train_reward=    1.60, train_len= 151.3
[Iter   44] train_reward=    1.36, train_len= 153.1
[Iter   45] train_reward=    1.36, train_len= 152.9
[Iter   46] train_reward=    1.31, train_len= 150.3
[Iter   47] train_reward=    1.22, train_len= 148.1
[Iter   48] train_reward=    1.86, train_len= 142.9
[Iter   49] train_reward=    2.62, train_len= 132.8
[Iter   50] train_reward=    2.97, train_len= 131.8 | eval_return=   -5.96,
eval_len= 200.0 | catch_rate=  5.0%
[Iter   51] train_reward=    2.89, train_len= 126.3
[Iter   52] train_reward=    2.91, train_len= 125.5
[Iter   53] train_reward=    3.70, train_len= 114.3
```

```
[Iter   54] train_reward=    3.74, train_len= 114.1
[Iter   55] train_reward=    3.24, train_len= 117.5
[Iter   56] train_reward=    2.77, train_len= 119.2
[Iter   57] train_reward=    2.85, train_len= 119.0
[Iter   58] train_reward=    3.66, train_len= 110.6
[Iter   59] train_reward=    3.58, train_len= 116.4
[Iter   60] train_reward=    2.97, train_len= 120.6 | eval_return=  -12.08,
eval_len= 200.0 | catch_rate=  5.0%
[Iter   61] train_reward=    2.97, train_len= 123.3
[Iter   62] train_reward=    3.10, train_len= 117.7
[Iter   63] train_reward=    3.63, train_len= 114.4
[Iter   64] train_reward=    3.66, train_len= 115.9
[Iter   65] train_reward=    3.30, train_len= 122.7
[Iter   66] train_reward=    3.70, train_len= 119.5
[Iter   67] train_reward=    3.60, train_len= 119.0
[Iter   68] train_reward=    3.79, train_len= 115.2
[Iter   69] train_reward=    3.31, train_len= 118.5
[Iter   70] train_reward=    2.81, train_len= 120.1 | eval_return=   -0.16,
eval_len= 200.0 | catch_rate= 15.0%
  New best eval_return=-0.16, saved to
TrainingResult(checkpoint=Checkpoint(filesystem=local,
path=frog_transformer_checkpoints), metrics={'custom_metrics': {},
'episode_media': {}, 'info': {'learner': {'default_policy': {'learner_stats':
{'allreduce_latency': np.float64(0.0), 'grad_gnorm': np.float64(0.5),
'cur_kl_coeff': np.float64(0.15000000000000002), 'cur_lr':
np.float64(3.0000000000000004e-05), 'total_loss': np.float64(5.697734851837158),
'policy_loss': np.float64(-0.006176560620466868), 'vf_loss':
np.float64(5.71805336634318), 'vf_explained_var':
np.float64(-0.12223828077316284), 'kl': np.float64(0.03520173112551371),
'entropy': np.float64(3.8844488271077475), 'entropy_coeff': np.float64(0.005)},
'model': {}, 'custom_metrics': {}, 'num_agent_steps_trained': np.float64(256.0),
'num_grad_updates_lifetime': np.float64(5288.0),
'diff_num_grad_updates_vs_sampler_policy': np.float64(37.0)}},
'num_env_steps_sampled': 284000, 'num_env_steps_trained': 284000,
'num_agent_steps_sampled': 284000, 'num_agent_steps_trained': 284000},
'env_runners': {'episode_reward_max': np.float32(12.317617),
'episode_reward_min': np.float32(-13.891272), 'episode_reward_mean':
np.float32(2.807567), 'episode_len_mean': np.float64(120.11), 'episode_media':
{}, 'episodes_timesteps_total': 12011, 'policy_reward_min': {'default_policy':
np.float32(-13.891272)}, 'policy_reward_max': {'default_policy':
np.float32(12.317617)}, 'policy_reward_mean': {'default_policy':
np.float32(2.807567)}, 'custom_metrics': {}, 'hist_stats': {'episode_reward':
[np.float32(-10.457832), np.float32(8.117405), np.float32(-3.7938516),
np.float32(5.4970846), np.float32(5.2759094), np.float32(9.203596),
np.float32(-7.424378), np.float32(9.040428), np.float32(5.3009114),
np.float32(9.539022), np.float32(3.793823), np.float32(-4.11168),
np.float32(-2.6547172), np.float32(-5.904425), np.float32(5.5972586),
np.float32(7.1318893), np.float32(4.846459), np.float32(4.39871),
```

np.float32(8.181108), np.float32(1.9051981), np.float32(9.350109),
np.float32(0.6476369), np.float32(-0.17583257), np.float32(0.41677475),
np.float32(8.531247), np.float32(-3.4318216), np.float32(-4.5562263),
np.float32(-4.190144), np.float32(7.955389), np.float32(12.317617),
np.float32(-4.4986434), np.float32(7.693143), np.float32(6.3853974),
np.float32(-6.290005), np.float32(8.601054), np.float32(-0.73383904),
np.float32(5.768003), np.float32(0.4825883), np.float32(10.742048),
np.float32(6.0926385), np.float32(6.1502647), np.float32(4.619221),
np.float32(-7.8306646), np.float32(8.508225), np.float32(2.9037404),
np.float32(5.4326625), np.float32(11.150709), np.float32(4.662238),
np.float32(-6.380782), np.float32(4.5540757), np.float32(10.104273),
np.float32(0.34935462), np.float32(8.22995), np.float32(0.97370964),
np.float32(5.066594), np.float32(0.10930835), np.float32(10.321445),
np.float32(6.6493616), np.float32(1.2618794), np.float32(0.9869504),
np.float32(7.4588385), np.float32(3.8148284), np.float32(-3.9868813),
np.float32(6.7612524), np.float32(5.409159), np.float32(7.4827256),
np.float32(8.019243), np.float32(3.921918), np.float32(7.39614),
np.float32(2.865319), np.float32(1.875334), np.float32(5.7098346),
np.float32(2.0441294), np.float32(7.3767285), np.float32(5.241224),
np.float32(3.9747982), np.float32(-13.891272), np.float32(5.640334),
np.float32(-6.1638904), np.float32(3.4611707), np.float32(9.989265),
np.float32(2.6193821), np.float32(-0.21753788), np.float32(8.606829),
np.float32(-6.1696963), np.float32(-5.0867596), np.float32(7.5358696),
np.float32(3.7012157), np.float32(-9.141574), np.float32(10.756256),
np.float32(6.7030926), np.float32(6.4686394), np.float32(-1.7882595),
np.float32(4.1359396), np.float32(6.1948614), np.float32(-8.240129),
np.float32(5.939473), np.float32(-6.0761194), np.float32(-7.303649),
np.float32(-4.6929207)], 'episode_lengths': [200, 188, 200, 167, 171, 42, 200,
89, 146, 107, 100, 200, 200, 200, 8, 56, 134, 85, 52, 73, 43, 119, 200, 67, 49,
200, 200, 200, 72, 46, 200, 54, 165, 200, 31, 143, 28, 184, 42, 129, 83, 89,
200, 45, 48, 5, 64, 195, 200, 74, 79, 200, 131, 200, 148, 200, 55, 157, 104,
137, 58, 138, 200, 47, 27, 82, 84, 54, 48, 140, 126, 40, 196, 20, 51, 161, 200,
42, 200, 50, 78, 155, 100, 49, 200, 200, 58, 140, 200, 78, 48, 165, 132, 143,
32, 200, 86, 200, 200, 179], 'policy_default_policy_reward':
[np.float32(-10.457832), np.float32(8.117405), np.float32(-3.7938516),
np.float32(5.4970846), np.float32(5.2759094), np.float32(9.203596),
np.float32(-7.424378), np.float32(9.040428), np.float32(5.3009114),
np.float32(9.539022), np.float32(3.793823), np.float32(-4.11168),
np.float32(-2.6547172), np.float32(-5.904425), np.float32(5.5972586),
np.float32(7.1318893), np.float32(4.846459), np.float32(4.39871),
np.float32(8.181108), np.float32(1.9051981), np.float32(9.350109),
np.float32(0.6476369), np.float32(-0.17583257), np.float32(0.41677475),
np.float32(8.531247), np.float32(-3.4318216), np.float32(-4.5562263),
np.float32(-4.190144), np.float32(7.955389), np.float32(12.317617),
np.float32(-4.4986434), np.float32(7.693143), np.float32(6.3853974),
np.float32(-6.290005), np.float32(8.601054), np.float32(-0.73383904),
np.float32(5.768003), np.float32(0.4825883), np.float32(10.742048),
np.float32(6.0926385), np.float32(6.1502647), np.float32(4.619221),

np.float32(-7.8306646), np.float32(8.508225), np.float32(2.9037404),
np.float32(5.4326625), np.float32(11.150709), np.float32(4.662238),
np.float32(-6.380782), np.float32(4.5540757), np.float32(10.104273),
np.float32(0.34935462), np.float32(8.22995), np.float32(0.97370964),
np.float32(5.066594), np.float32(0.10930835), np.float32(10.321445),
np.float32(6.6493616), np.float32(1.2618794), np.float32(0.9869504),
np.float32(7.4588385), np.float32(3.8148284), np.float32(-3.9868813),
np.float32(6.7612524), np.float32(5.409159), np.float32(7.4827256),
np.float32(8.019243), np.float32(3.921918), np.float32(7.39614),
np.float32(2.865319), np.float32(1.875334), np.float32(5.7098346),
np.float32(2.0441294), np.float32(7.3767285), np.float32(5.241224),
np.float32(3.9747982), np.float32(-13.891272), np.float32(5.640334),
np.float32(-6.1638904), np.float32(3.4611707), np.float32(9.989265),
np.float32(2.6193821), np.float32(-0.21753788), np.float32(8.606829),
np.float32(-6.1696963), np.float32(-5.0867596), np.float32(7.5358696),
np.float32(3.7012157), np.float32(-9.141574), np.float32(10.756256),
np.float32(6.7030926), np.float32(6.4686394), np.float32(-1.7882595),
np.float32(4.1359396), np.float32(6.1948614), np.float32(-8.240129),
np.float32(5.939473), np.float32(-6.0761194), np.float32(-7.303649),
np.float32(-4.6929207)]}, 'sampler_perf': {'mean_raw_obs_processing_ms':
np.float64(1.1020909110906907), 'mean_inference_ms':
np.float64(3.773527399750661), 'mean_action_processing_ms':
np.float64(0.45281442954694584), 'mean_env_wait_ms':
np.float64(0.7472091563830346), 'mean_env_render_ms': np.float64(0.0)},
'num_faulty_episodes': 0, 'connector_metrics': {'ObsPreprocessorConnector_ms':
np.float64(0.005308389663696289), 'StateBufferConnector_ms':
np.float64(0.0047185420989990234), 'ViewRequirementAgentConnector_ms':
np.float64(0.09519529342651367)}, 'num_episodes': 32, 'episode_return_max':
np.float32(12.317617), 'episode_return_min': np.float32(-13.891272),
'episode_return_mean': np.float32(2.807567), 'episodes_this_iter': 32},
'num_healthy_workers': 1, 'actor_manager_num_outstanding_async_reqs': 0,
'num_remote_worker_restarts': 0, 'num_agent_steps_sampled': 284000,
'num_agent_steps_trained': 284000, 'num_env_steps_sampled': 284000,
'num_env_steps_trained': 284000, 'num_env_steps_sampled_this_iter': 4000,
'num_env_steps_trained_this_iter': 4000,
'num_env_steps_sampled_throughput_per_sec': 99.0323915570758,
'num_env_steps_trained_throughput_per_sec': 99.0323915570758, 'timesteps_total':
284000, 'num_env_steps_sampled_lifetime': 284000,
'num_agent_steps_sampled_lifetime': 284000, 'num_steps_trained_this_iter': 4000,
'agent_timesteps_total': 284000, 'timers': {'training_iteration_time_ms':
39772.353, 'restore_workers_time_ms': 0.019, 'training_step_time_ms': 39772.291,
'sample_time_ms': 6187.115, 'load_time_ms': 1.58, 'load_throughput':
2531874.925, 'learn_time_ms': 33577.551, 'learn_throughput': 119.127,
'synch_weights_time_ms': 5.626}, 'counters': {'num_env_steps_sampled': 284000,
'num_env_steps_trained': 284000, 'num_agent_steps_sampled': 284000,
'num_agent_steps_trained': 284000}, 'done': False, 'training_iteration': 71,
'trial_id': 'default', 'date': '2025-12-02_11-08-54', 'timestamp': 1764691734,
'time_this_iter_s': 40.39683556556702, 'time_total_s': 2833.950895547867, 'pid':

2877155, 'hostname': 'platinumfish-Aspire', 'node_ip': '192.168.86.52',
'config': {'exploration_config': {'type': 'StochasticSampling'},
'extra_python_environs_for_driver': {}, 'extra_python_environs_for_worker': {},
'placement_strategy': 'PACK', 'num_gpus': 0, '_fake_gpus': False,
'num_cpus_for_main_process': 1, 'eager_tracing': True, 'eager_max_retraces': 20,
'tf_session_args': {'intra_op_parallelism_threads': 2,
'inter_op_parallelism_threads': 2, 'gpu_options': {'allow_growth': True},
'log_device_placement': False, 'device_count': {'CPU': 1},
'allow_soft_placement': True}, 'local_tf_session_args':
{'intra_op_parallelism_threads': 8, 'inter_op_parallelism_threads': 8},
'torch_compile_learner': False, 'torch_compile_learner_what_to_compile':
<TorchCompileWhatToCompile.FORWARD_TRAIN: 'forward_train'>,
'torch_compile_learner_dynamo_backend': 'inductor',
'torch_compile_learner_dynamo_mode': None, 'torch_compile_worker': False,
'torch_compile_worker_dynamo_backend': 'onnxrt',
'torch_compile_worker_dynamo_mode': None, 'torch_ddp_kwargs': {},
'torch_skip_nan_gradients': False, 'env': 'frog_hist_env', 'env_config':
{'space_size': 1.0, 'step_size': 0.1, 'fly_speed': 0.05, 'catch_radius': 0.15,
'max_steps': 200, 'num_freqs': 4, 'use_positional_encodings': True},
'observation_space': None, 'action_space': None, 'clip_rewards': None,
'normalize_actions': True, 'clip_actions': False, '_is_atari': None,
'disable_env_checking': False, 'render_env': False, 'action_mask_key':
'action_mask', 'env_runner_cls': None, 'num_env_runners': 1,
'create_local_env_runner': True, 'num_envs_per_env_runner': 4,
'gym_env_vectorize_mode': 'sync', 'num_cpus_per_env_runner': 1,
'num_gpus_per_env_runner': 0, 'custom_resources_per_env_runner': {},
'validate_env_runners_after_construction': True, 'episodes_to_numpy': True,
'max_requests_in_flight_per_env_runner': 1, 'sample_timeout_s': 60.0,
'_env_to_module_connector': None,
'add_default_connectors_to_env_to_module_pipeline': True,
'_module_to_env_connector': None,
'add_default_connectors_to_module_to_env_pipeline': True,
'merge_env_runner_states': 'training_only', 'broadcast_env_runner_states': True,
'episode_lookback_horizon': 1, 'rollout_fragment_length': 'auto', 'batch_mode':
'truncate_episodes', 'compress_observations': False, 'remote_worker_envs':
False, 'remote_env_batch_wait_ms': 0, 'enable_tf1_exec_eagerly': False,
'sample_collector': <class
'ray.rllib.evaluation.collectors.simple_list_collector.SimpleListCollector'>,
'preprocessor_pref': 'deepmind', 'observation_filter': 'NoFilter',
'update_worker_filter_stats': True, 'use_worker_filter_stats': True,
'sampler_perf_stats_ema_coef': None, '_is_online': True, 'num_learners': 0,
'num_gpus_per_learner': 0, 'num_cpus_per_learner': 'auto',
'num_aggregator_actors_per_learner': 0,
'max_requests_in_flight_per_aggregator_actor': 3, 'local_gpu_idx': 0,
'max_requests_in_flight_per_learner': 3, 'gamma': 0.99, 'lr': 3e-05,
'grad_clip': 0.5, 'grad_clip_by': 'global_norm',
'_train_batch_size_per_learner': None, 'train_batch_size': 4000, 'num_epochs':
5, 'minibatch_size': 256, 'shuffle_batch_per_epoch': True, 'model':

{'fcnet_hiddens': [256, 256], 'fcnet_activation': 'tanh',
'fcnet_weights_initializer': None, 'fcnet_weights_initializer_config': None,
'fcnet_bias_initializer': None, 'fcnet_bias_initializer_config': None,
'conv_filters': None, 'conv_activation': 'relu', 'conv_kernel_initializer':
None, 'conv_kernel_initializer_config': None, 'conv_bias_initializer': None,
'conv_bias_initializer_config': None, 'conv_transpose_kernel_initializer': None,
'conv_transpose_kernel_initializer_config': None,
'conv_transpose_bias_initializer': None,
'conv_transpose_bias_initializer_config': None, 'post_fcnet_hiddens': [],
'post_fcnet_activation': 'relu', 'post_fcnet_weights_initializer': None,
'post_fcnet_weights_initializer_config': None, 'post_fcnet_bias_initializer':
None, 'post_fcnet_bias_initializer_config': None, 'free_log_std': False,
'log_std_clip_param': 20.0, 'no_final_linear': False, 'vf_share_layers': False,
'use_lstm': False, 'max_seq_len': 20, 'lstm_cell_size': 256,
'lstm_use_prev_action': False, 'lstm_use_prev_reward': False,
'lstm_weights_initializer': None, 'lstm_weights_initializer_config': None,
'lstm_bias_initializer': None, 'lstm_bias_initializer_config': None,
'_time_major': False, 'use_attention': False, 'attention_num_transformer_units':
1, 'attention_dim': 64, 'attention_num_heads': 1, 'attention_head_dim': 32,
'attention_memory_inference': 50, 'attention_memory_training': 50,
'attention_position_wise_mlp_dim': 32, 'attention_init_gru_gate_bias': 2.0,
'attention_use_n_prev_actions': 0, 'attention_use_n_prev_rewards': 0,
'framestack': True, 'dim': 84, 'grayscale': False, 'zero_mean': True,
'custom_model': 'frog_transformer_policy', 'custom_model_config': {'d_model':
128, 'nhead': 4, 'num_layers': 2, 'dim_feedforward': 256, 'dropout': 0.05,
'seq_len': 24}, 'custom_action_dist': None, 'custom_preprocessor': None,
'encoder_latent_dim': None, 'always_check_shapes': False,
'lstm_use_prev_action_reward': -1, '_use_default_native_models': -1,
'_disable_preprocessor_api': False, '_disable_action_flattening': False},
'_learner_connector': None, 'add_default_connectors_to_learner_pipeline': True,
'learner_config_dict': {}, 'optimizer': {}, '_learner_class': None,
'callbacks_on_algorithm_init': None, 'callbacks_on_env_runners_recreated': None,
'callbacks_on_offline_eval_runners_recreated': None,
'callbacks_on_checkpoint_loaded': None, 'callbacks_on_environment_created':
None, 'callbacks_on_episode_created': None, 'callbacks_on_episode_start': None,
'callbacks_on_episode_step': None, 'callbacks_on_episode_end': None,
'callbacks_on_evaluate_start': None, 'callbacks_on_evaluate_end': None,
'callbacks_on_evaluate_offline_start': None,
'callbacks_on_evaluate_offline_end': None, 'callbacks_on_sample_end': None,
'callbacks_on_train_result': None, 'explore': True,
'enable_rl_module_and_learner': False, 'enable_env_runner_and_connector_v2':
False, '_prior_exploration_config': None, 'count_steps_by': 'env_steps',
'policy_map_capacity': 100, 'policy_mapping_fn': <function
AlgorithmConfig.DEFAULT_POLICY_MAPPING_FN at 0x7881bb1395a0>,
'policies_to_train': None, 'policy_states_are_swappable': False,
'observation_fn': None, 'offline_data_class': None, 'input_read_method':
'read_parquet', 'input_read_method_kwargs': {}, 'input_read_schema': {},
'input_read_episodes': False, 'input_read_sample_batches': False,

'input_read_batch_size': None, 'input_filesystem': None,
'input_filesystem_kwargs': {}, 'input_compress_columns': ['obs', 'new_obs'],
'input_spaces_jsonable': True, 'materialize_data': False,
'materialize_mapped_data': True, 'map_batches_kwargs': {},
'iter_batches_kwargs': {}, 'ignore_final_observation': False,
'prelearner_class': None, 'prelearner_buffer_class': None,
'prelearner_buffer_kwargs': {}, 'prelearner_module_synch_period': 10,
'dataset_num_iters_per_learner': None, 'input_config': {},
'actions_in_input_normalized': False, 'postprocess_inputs': False,
'shuffle_buffer_size': 0, 'output': None, 'output_config': {},
'output_compress_columns': ['obs', 'new_obs'], 'output_max_file_size': 67108864,
'output_max_rows_per_file': None, 'output_write_remaining_data': False,
'output_write_method': 'write_parquet', 'output_write_method_kwargs': {},
'output_filesystem': None, 'output_filesystem_kwargs': {},
'output_write_episodes': True, 'offline_sampling': False, 'evaluation_interval':
None, 'evaluation_duration': 10, 'evaluation_duration_unit': 'episodes',
'evaluation_sample_timeout_s': 120.0,
'evaluation_auto_duration_min_env_steps_per_sample': 100,
'evaluation_auto_duration_max_env_steps_per_sample': 2000,
'evaluation_parallel_to_training': False,
'evaluation_force_reset_envs_before_iteration': True, 'evaluation_config': None,
'off_policy_estimation_methods': {}, 'ope_split_batch_by_episode': True,
'evaluation_num_env_runners': 0, 'in_evaluation': False,
'sync_filters_on_rollout_workers_timeout_s': 10.0,
'offline_evaluation_interval': None, 'num_offline_eval_runners': 0,
'offline_evaluation_type': None, 'offline_eval_runner_class': None,
'offline_loss_for_module_fn': None, 'offline_evaluation_duration': 1,
'offline_evaluation_parallel_to_training': False,
'offline_evaluation_timeout_s': 120.0, 'num_cpus_per_offline_eval_runner': 1,
'num_gpus_per_offline_eval_runner': 0,
'custom_resources_per_offline_eval_runner': {},
'restart_failed_offline_eval_runners': True,
'ignore_offline_eval_runner_failures': False,
'max_num_offline_eval_runner_restarts': 1000,
'offline_eval_runner_restore_timeout_s': 1800.0,
'max_requests_in_flight_per_offline_eval_runner': 1,
'validate_offline_eval_runners_after_construction': True,
'offline_eval_runner_health_probe_timeout_s': 30.0,
'offline_eval_rl_module_inference_only': False,
'broadcast_offline_eval_runner_states': False,
'offline_eval_batch_size_per_runner': 256, 'dataset_num_iters_per_eval_runner':
1, 'keep_per_episode_custom_metrics': False,
'metrics_episode_collection_timeout_s': 60.0,
'metrics_num_episodes_for_smoothing': 100, 'min_time_s_per_iteration': None,
'min_train_timesteps_per_iteration': 0, 'min_sample_timesteps_per_iteration': 0,
'log_gradients': False, 'export_native_model_files': False,
'checkpoint_trainable_policies_only': False, 'logger_creator': None,
'logger_config': None, 'log_level': 'WARN', 'log_sys_usage': True,

'fake_sampler': False, 'seed': None, 'restart_failed_env_runners': True,
'ignore_env_runner_failures': False, 'max_num_env_runner_restarts': 1000,
'delay_between_env_runner_restarts_s': 60.0, 'restart_failed_sub_environments':
False, 'num_consecutive_env_runner_failures_tolerance': 100,
'env_runner_health_probe_timeout_s': 30.0, 'env_runner_restore_timeout_s':
1800.0, '_model_config': {}, '_rl_module_spec': None,
'algorithm_config_overrides_per_module': {}, '_per_module_overrides': {},
'_validate_config': True, '_use_msgpack_checkpoints': False,
'_torch_grad_scaler_class': None, '_torch_lr_scheduler_classes': None,
'_tf_policy_handles_more_than_one_loss': False, '_disable_preprocessor_api':
False, '_disable_action_flattening': False,
'_disable_initialize_loss_from_dummy_batch': False,
'_dont_auto_sync_env_runner_states': False, 'env_task_fn': -1,
'enable_connectors': -1, 'simple_optimizer': False, 'policy_map_cache': -1,
'worker_cls': -1, 'synchronize_filters': -1, 'enable_async_evaluation': -1,
'custom_async_evaluation_function': -1, '_enable_rl_module_api': -1,
'auto_wrap_old_gym_envs': -1, 'always_attach_evaluation_results': -1,
'replay_sequence_length': None, '_disable_execution_plan_api': -1, 'use_critic':
True, 'use_gae': True, 'use_kl_loss': True, 'kl_coeff': 0.1, 'kl_target': 0.02,
'vf_loss_coeff': 1.0, 'entropy_coeff': 0.005, 'clip_param': 0.2,
'vf_clip_param': 10.0, 'entropy_coeff_schedule': None, 'lr_schedule': None,
'sgd_minibatch_size': -1, 'vf_share_layers': -1, 'class': <class
'ray.rllib.algorithms.ppo.ppo.PPOConfig'>, 'lambda': 1.0, 'input': 'sampler',
'policies': {'default_policy': (None, None, None, None)}, 'callbacks': <class
'ray.rllib.callbacks.callbacks.RLlibCallback'>, 'create_env_on_driver': True,
'custom_eval_function': None, 'framework': 'torch'}, 'time_since_restore':
2833.950895547867, 'iterations_since_restore': 71, 'perf': {'cpu_util_percent':
np.float64(58.23508771929825), 'ram_util_percent':
np.float64(78.69298245614036)}})
[Iter 71] train_reward= 2.87, train_len= 119.9
[Iter 72] train_reward= 3.17, train_len= 112.6
[Iter 73] train_reward= 3.68, train_len= 110.7
[Iter 74] train_reward= 3.67, train_len= 106.3
[Iter 75] train_reward= 2.88, train_len= 118.1
[Iter 76] train_reward= 3.04, train_len= 121.4
[Iter 77] train_reward= 3.05, train_len= 127.4
[Iter 78] train_reward= 3.74, train_len= 119.8
[Iter 79] train_reward= 4.48, train_len= 112.7
[Iter 80] train_reward= 5.12, train_len= 99.3 | eval_return= 2.87,
eval_len= 200.0 | catch_rate= 0.0%
New best eval_return=2.87, saved to
TrainingResult(checkpoint=Checkpoint(filesystem=local,
path=frog_transformer_checkpoints), metrics={'custom_metrics': {},
'episode_media': {}, 'info': {'learner': {'default_policy': {'learner_stats':
{'allreduce_latency': np.float64(0.0), 'grad_gnorm': np.float64(0.5),
'cur_kl_coeff': np.float64(0.15000000000000002), 'cur_lr':
np.float64(3.0000000000000004e-05), 'total_loss': np.float64(5.281740563710531),
'policy_loss': np.float64(0.011182364722092946), 'vf_loss':

np.float64(5.284133207003276), 'vf_explained_var':
np.float64(-0.0009879342714945475), 'kl': np.float64(0.037877723947167395),
'entropy': np.float64(3.851337715784709), 'entropy_coeff': np.float64(0.005)},
'model': {}, 'custom_metrics': {}, 'num_agent_steps_trained': np.float64(256.0),
'num_grad_updates_lifetime': np.float64(6038.0),
'diff_num_grad_updates_vs_sampler_policy': np.float64(37.0)}},
'num_env_steps_sampled': 324000, 'num_env_steps_trained': 324000,
'num_agent_steps_sampled': 324000, 'num_agent_steps_trained': 324000},
'env_runners': {'episode_reward_max': np.float32(11.424236),
'episode_reward_min': np.float32(-9.488344), 'episode_reward_mean':
np.float32(5.1239967), 'episode_len_mean': np.float64(99.29), 'episode_media':
{}, 'episodes_timesteps_total': 9929, 'policy_reward_min': {'default_policy':
np.float32(-9.488344)}, 'policy_reward_max': {'default_policy':
np.float32(11.424236)}, 'policy_reward_mean': {'default_policy':
np.float32(5.1239967)}, 'custom_metrics': {}, 'hist_stats': {'episode_reward':
[np.float32(6.9431615), np.float32(7.226815), np.float32(5.081476),
np.float32(4.0155544), np.float32(5.814813), np.float32(9.369193),
np.float32(1.1174669), np.float32(5.974256), np.float32(5.994256),
np.float32(6.0474796), np.float32(9.636221), np.float32(-2.5289516),
np.float32(5.247898), np.float32(-9.488344), np.float32(10.355425),
np.float32(5.523029), np.float32(9.085692), np.float32(4.93439),
np.float32(2.6570694), np.float32(-1.8666972), np.float32(8.000383),
np.float32(9.901908), np.float32(6.8930264), np.float32(6.2096305),
np.float32(3.5267248), np.float32(5.819652), np.float32(7.9042826),
np.float32(8.791913), np.float32(6.2766457), np.float32(8.451248),
np.float32(4.3818707), np.float32(5.3311863), np.float32(9.600046),
np.float32(5.6164002), np.float32(2.9007401), np.float32(2.0854042),
np.float32(-0.20201683), np.float32(11.424236), np.float32(3.2493634),
np.float32(-2.9184146), np.float32(1.6010876), np.float32(-1.5486133),
np.float32(10.260895), np.float32(10.356154), np.float32(8.8355055),
np.float32(4.0273924), np.float32(5.853879), np.float32(-0.65157837),
np.float32(4.7485294), np.float32(6.7541947), np.float32(5.6967673),
np.float32(4.2513623), np.float32(8.221378), np.float32(4.497136),
np.float32(8.628666), np.float32(1.8892264), np.float32(9.364117),
np.float32(4.97035), np.float32(7.596871), np.float32(1.7313671),
np.float32(-1.9103842), np.float32(5.9706335), np.float32(8.953151),
np.float32(6.759527), np.float32(7.312421), np.float32(7.035463),
np.float32(4.2419176), np.float32(5.9613996), np.float32(9.426376),
np.float32(6.0360985), np.float32(9.089771), np.float32(9.026042),
np.float32(0.80618095), np.float32(-4.1088285), np.float32(-3.806323),
np.float32(9.5460205), np.float32(-4.782803), np.float32(5.000638),
np.float32(8.499597), np.float32(-1.2848957), np.float32(10.483772),
np.float32(4.354721), np.float32(4.735107), np.float32(5.4351926),
np.float32(8.744544), np.float32(6.34044), np.float32(3.8593795),
np.float32(3.948627), np.float32(8.247536), np.float32(-0.052104726),
np.float32(8.446475), np.float32(7.7447104), np.float32(6.914058),
np.float32(8.314492), np.float32(6.0663986), np.float32(11.41954),
np.float32(2.9295423), np.float32(5.445882), np.float32(5.0657787),

np.float32(-5.285611)], 'episode_lengths': [142, 46, 80, 142, 11, 57, 113, 102,
150, 39, 31, 200, 197, 187, 134, 143, 55, 101, 178, 200, 15, 89, 90, 100, 134,
90, 96, 51, 64, 133, 196, 57, 37, 20, 184, 155, 188, 73, 67, 200, 83, 200, 63,
30, 56, 182, 86, 200, 141, 101, 71, 93, 148, 115, 47, 61, 68, 22, 13, 145, 200,
37, 45, 126, 107, 73, 103, 85, 82, 64, 28, 30, 193, 200, 200, 48, 200, 97, 24,
200, 27, 12, 23, 148, 25, 68, 72, 92, 94, 200, 74, 36, 50, 24, 10, 67, 183, 68,
42, 200], 'policy_default_policy_reward': [np.float32(6.9431615),
np.float32(7.226815), np.float32(5.081476), np.float32(4.0155544),
np.float32(5.814813), np.float32(9.369193), np.float32(1.1174669),
np.float32(5.974256), np.float32(5.994256), np.float32(6.0474796),
np.float32(9.636221), np.float32(-2.5289516), np.float32(5.247898),
np.float32(-9.488344), np.float32(10.355425), np.float32(5.523029),
np.float32(9.085692), np.float32(4.93439), np.float32(2.6570694),
np.float32(-1.8666972), np.float32(8.000383), np.float32(9.901908),
np.float32(6.8930264), np.float32(6.2096305), np.float32(3.5267248),
np.float32(5.819652), np.float32(7.9042826), np.float32(8.791913),
np.float32(6.2766457), np.float32(8.451248), np.float32(4.3818707),
np.float32(5.3311863), np.float32(9.600046), np.float32(5.6164002),
np.float32(2.9007401), np.float32(2.0854042), np.float32(-0.20201683),
np.float32(11.424236), np.float32(3.2493634), np.float32(-2.9184146),
np.float32(1.6010876), np.float32(-1.5486133), np.float32(10.260895),
np.float32(10.356154), np.float32(8.8355055), np.float32(4.0273924),
np.float32(5.853879), np.float32(-0.65157837), np.float32(4.7485294),
np.float32(6.7541947), np.float32(5.6967673), np.float32(4.2513623),
np.float32(8.221378), np.float32(4.497136), np.float32(8.628666),
np.float32(1.8892264), np.float32(9.364117), np.float32(4.97035),
np.float32(7.596871), np.float32(1.7313671), np.float32(-1.9103842),
np.float32(5.9706335), np.float32(8.953151), np.float32(6.759527),
np.float32(7.312421), np.float32(7.035463), np.float32(4.2419176),
np.float32(5.9613996), np.float32(9.426376), np.float32(6.0360985),
np.float32(9.089771), np.float32(9.026042), np.float32(0.80618095),
np.float32(-4.1088285), np.float32(-3.806323), np.float32(9.5460205),
np.float32(-4.782803), np.float32(5.000638), np.float32(8.499597),
np.float32(-1.2848957), np.float32(10.483772), np.float32(4.354721),
np.float32(4.735107), np.float32(5.4351926), np.float32(8.744544),
np.float32(6.34044), np.float32(3.8593795), np.float32(3.948627),
np.float32(8.247536), np.float32(-0.052104726), np.float32(8.446475),
np.float32(7.7447104), np.float32(6.914058), np.float32(8.314492),
np.float32(6.0663986), np.float32(11.41954), np.float32(2.9295423),
np.float32(5.445882), np.float32(5.0657787), np.float32(-5.285611)]},
'sampler_perf': {'mean_raw_obs_processing_ms': np.float64(1.0999896506384923),
'mean_inference_ms': np.float64(3.7694587185063906),
'mean_action_processing_ms': np.float64(0.45242239665886413),
'mean_env_wait_ms': np.float64(0.7461564053422921), 'mean_env_render_ms':
np.float64(0.0)}, 'num_faulty_episodes': 0, 'connector_metrics':
{'ObsPreprocessorConnector_ms': np.float64(0.006312370300292969),
'StateBufferConnector_ms': np.float64(0.004434108734130859),
'ViewRequirementAgentConnector_ms': np.float64(0.09534811973571777)},

'num_episodes': 45, 'episode_return_max': np.float32(11.424236),
'episode_return_min': np.float32(-9.488344), 'episode_return_mean':
np.float32(5.1239967), 'episodes_this_iter': 45}, 'num_healthy_workers': 1,
'actor_manager_num_outstanding_async_reqs': 0, 'num_remote_worker_restarts': 0,
'num_agent_steps_sampled': 324000, 'num_agent_steps_trained': 324000,
'num_env_steps_sampled': 324000, 'num_env_steps_trained': 324000,
'num_env_steps_sampled_this_iter': 4000, 'num_env_steps_trained_this_iter':
4000, 'num_env_steps_sampled_throughput_per_sec': 99.72332942600283,
'num_env_steps_trained_throughput_per_sec': 99.72332942600283,
'timesteps_total': 324000, 'num_env_steps_sampled_lifetime': 324000,
'num_agent_steps_sampled_lifetime': 324000, 'num_steps_trained_this_iter': 4000,
'agent_timesteps_total': 324000, 'timers': {'training_iteration_time_ms':
39759.528, 'restore_workers_time_ms': 0.023, 'training_step_time_ms': 39759.459,
'sample_time_ms': 6095.337, 'load_time_ms': 1.604, 'load_throughput':
2493381.485, 'learn_time_ms': 33656.376, 'learn_throughput': 118.848,
'synch_weights_time_ms': 5.716}, 'counters': {'num_env_steps_sampled': 324000,
'num_env_steps_trained': 324000, 'num_agent_steps_sampled': 324000,
'num_agent_steps_trained': 324000}, 'done': False, 'training_iteration': 81,
'trial_id': 'default', 'date': '2025-12-02_11-15-43', 'timestamp': 1764692143,
'time_this_iter_s': 40.117260456085205, 'time_total_s': 3231.6118206977844,
'pid': 2877155, 'hostname': 'platinumfish-Aspire', 'node_ip': '192.168.86.52',
'config': {'exploration_config': {'type': 'StochasticSampling'},
'extra_python_environs_for_driver': {}, 'extra_python_environs_for_worker': {},
'placement_strategy': 'PACK', 'num_gpus': 0, '_fake_gpus': False,
'num_cpus_for_main_process': 1, 'eager_tracing': True, 'eager_max_retraces': 20,
'tf_session_args': {'intra_op_parallelism_threads': 2,
'inter_op_parallelism_threads': 2, 'gpu_options': {'allow_growth': True},
'log_device_placement': False, 'device_count': {'CPU': 1},
'allow_soft_placement': True}, 'local_tf_session_args':
{'intra_op_parallelism_threads': 8, 'inter_op_parallelism_threads': 8},
'torch_compile_learner': False, 'torch_compile_learner_what_to_compile':
<TorchCompileWhatToCompile.FORWARD_TRAIN: 'forward_train'>,
'torch_compile_learner_dynamo_backend': 'inductor',
'torch_compile_learner_dynamo_mode': None, 'torch_compile_worker': False,
'torch_compile_worker_dynamo_backend': 'onnxrt',
'torch_compile_worker_dynamo_mode': None, 'torch_ddp_kwargs': {},
'torch_skip_nan_gradients': False, 'env': 'frog_hist_env', 'env_config':
{'space_size': 1.0, 'step_size': 0.1, 'fly_speed': 0.05, 'catch_radius': 0.15,
'max_steps': 200, 'num_freqs': 4, 'use_positional_encodings': True},
'observation_space': None, 'action_space': None, 'clip_rewards': None,
'normalize_actions': True, 'clip_actions': False, '_is_atari': None,
'disable_env_checking': False, 'render_env': False, 'action_mask_key':
'action_mask', 'env_runner_cls': None, 'num_env_runners': 1,
'create_local_env_runner': True, 'num_envs_per_env_runner': 4,
'gym_env_vectorize_mode': 'sync', 'num_cpus_per_env_runner': 1,
'num_gpus_per_env_runner': 0, 'custom_resources_per_env_runner': {},
'validate_env_runners_after_construction': True, 'episodes_to_numpy': True,
'max_requests_in_flight_per_env_runner': 1, 'sample_timeout_s': 60.0,

'_env_to_module_connector': None,
'add_default_connectors_to_env_to_module_pipeline': True,
'_module_to_env_connector': None,
'add_default_connectors_to_module_to_env_pipeline': True,
'merge_env_runner_states': 'training_only', 'broadcast_env_runner_states': True,
'episode_lookback_horizon': 1, 'rollout_fragment_length': 'auto', 'batch_mode':
'truncate_episodes', 'compress_observations': False, 'remote_worker_envs':
False, 'remote_env_batch_wait_ms': 0, 'enable_tf1_exec_eagerly': False,
'sample_collector': <class
'ray.rllib.evaluation.collectors.simple_list_collector.SimpleListCollector'>,
'preprocessor_pref': 'deepmind', 'observation_filter': 'NoFilter',
'update_worker_filter_stats': True, 'use_worker_filter_stats': True,
'sampler_perf_stats_ema_coef': None, '_is_online': True, 'num_learners': 0,
'num_gpus_per_learner': 0, 'num_cpus_per_learner': 'auto',
'num_aggregator_actors_per_learner': 0,
'max_requests_in_flight_per_aggregator_actor': 3, 'local_gpu_idx': 0,
'max_requests_in_flight_per_learner': 3, 'gamma': 0.99, 'lr': 3e-05,
'grad_clip': 0.5, 'grad_clip_by': 'global_norm',
'_train_batch_size_per_learner': None, 'train_batch_size': 4000, 'num_epochs':
5, 'minibatch_size': 256, 'shuffle_batch_per_epoch': True, 'model':
{'fcnet_hiddens': [256, 256], 'fcnet_activation': 'tanh',
'fcnet_weights_initializer': None, 'fcnet_weights_initializer_config': None,
'fcnet_bias_initializer': None, 'fcnet_bias_initializer_config': None,
'conv_filters': None, 'conv_activation': 'relu', 'conv_kernel_initializer':
None, 'conv_kernel_initializer_config': None, 'conv_bias_initializer': None,
'conv_bias_initializer_config': None, 'conv_transpose_kernel_initializer': None,
'conv_transpose_kernel_initializer_config': None,
'conv_transpose_bias_initializer': None,
'conv_transpose_bias_initializer_config': None, 'post_fcnet_hiddens': [],
'post_fcnet_activation': 'relu', 'post_fcnet_weights_initializer': None,
'post_fcnet_weights_initializer_config': None, 'post_fcnet_bias_initializer':
None, 'post_fcnet_bias_initializer_config': None, 'free_log_std': False,
'log_std_clip_param': 20.0, 'no_final_linear': False, 'vf_share_layers': False,
'use_lstm': False, 'max_seq_len': 20, 'lstm_cell_size': 256,
'lstm_use_prev_action': False, 'lstm_use_prev_reward': False,
'lstm_weights_initializer': None, 'lstm_weights_initializer_config': None,
'lstm_bias_initializer': None, 'lstm_bias_initializer_config': None,
'_time_major': False, 'use_attention': False, 'attention_num_transformer_units':
1, 'attention_dim': 64, 'attention_num_heads': 1, 'attention_head_dim': 32,
'attention_memory_inference': 50, 'attention_memory_training': 50,
'attention_position_wise_mlp_dim': 32, 'attention_init_gru_gate_bias': 2.0,
'attention_use_n_prev_actions': 0, 'attention_use_n_prev_rewards': 0,
'framestack': True, 'dim': 84, 'grayscale': False, 'zero_mean': True,
'custom_model': 'frog_transformer_policy', 'custom_model_config': {'d_model':
128, 'nhead': 4, 'num_layers': 2, 'dim_feedforward': 256, 'dropout': 0.05,
'seq_len': 24}, 'custom_action_dist': None, 'custom_preprocessor': None,
'encoder_latent_dim': None, 'always_check_shapes': False,
'lstm_use_prev_action_reward': -1, '_use_default_native_models': -1,

'_disable_preprocessor_api': False, '_disable_action_flattening': False},
'_learner_connector': None, 'add_default_connectors_to_learner_pipeline': True,
'learner_config_dict': {}, 'optimizer': {}, '_learner_class': None,
'callbacks_on_algorithm_init': None, 'callbacks_on_env_runners_recreated': None,
'callbacks_on_offline_eval_runners_recreated': None,
'callbacks_on_checkpoint_loaded': None, 'callbacks_on_environment_created':
None, 'callbacks_on_episode_created': None, 'callbacks_on_episode_start': None,
'callbacks_on_episode_step': None, 'callbacks_on_episode_end': None,
'callbacks_on_evaluate_start': None, 'callbacks_on_evaluate_end': None,
'callbacks_on_evaluate_offline_start': None,
'callbacks_on_evaluate_offline_end': None, 'callbacks_on_sample_end': None,
'callbacks_on_train_result': None, 'explore': True,
'enable_rl_module_and_learner': False, 'enable_env_runner_and_connector_v2':
False, '_prior_exploration_config': None, 'count_steps_by': 'env_steps',
'policy_map_capacity': 100, 'policy_mapping_fn': <function
AlgorithmConfig.DEFAULT_POLICY_MAPPING_FN at 0x7881bb1395a0>,
'policies_to_train': None, 'policy_states_are_swappable': False,
'observation_fn': None, 'offline_data_class': None, 'input_read_method':
'read_parquet', 'input_read_method_kwargs': {}, 'input_read_schema': {},
'input_read_episodes': False, 'input_read_sample_batches': False,
'input_read_batch_size': None, 'input_filesystem': None,
'input_filesystem_kwargs': {}, 'input_compress_columns': ['obs', 'new_obs'],
'input_spaces_jsonable': True, 'materialize_data': False,
'materialize_mapped_data': True, 'map_batches_kwargs': {},
'iter_batches_kwargs': {}, 'ignore_final_observation': False,
'prelearner_class': None, 'prelearner_buffer_class': None,
'prelearner_buffer_kwargs': {}, 'prelearner_module_synch_period': 10,
'dataset_num_iters_per_learner': None, 'input_config': {},
'actions_in_input_normalized': False, 'postprocess_inputs': False,
'shuffle_buffer_size': 0, 'output': None, 'output_config': {},
'output_compress_columns': ['obs', 'new_obs'], 'output_max_file_size': 67108864,
'output_max_rows_per_file': None, 'output_write_remaining_data': False,
'output_write_method': 'write_parquet', 'output_write_method_kwargs': {},
'output_filesystem': None, 'output_filesystem_kwargs': {},
'output_write_episodes': True, 'offline_sampling': False, 'evaluation_interval':
None, 'evaluation_duration': 10, 'evaluation_duration_unit': 'episodes',
'evaluation_sample_timeout_s': 120.0,
'evaluation_auto_duration_min_env_steps_per_sample': 100,
'evaluation_auto_duration_max_env_steps_per_sample': 2000,
'evaluation_parallel_to_training': False,
'evaluation_force_reset_envs_before_iteration': True, 'evaluation_config': None,
'off_policy_estimation_methods': {}, 'ope_split_batch_by_episode': True,
'evaluation_num_env_runners': 0, 'in_evaluation': False,
'sync_filters_on_rollout_workers_timeout_s': 10.0,
'offline_evaluation_interval': None, 'num_offline_eval_runners': 0,
'offline_evaluation_type': None, 'offline_eval_runner_class': None,
'offline_loss_for_module_fn': None, 'offline_evaluation_duration': 1,
'offline_evaluation_parallel_to_training': False,

'offline_evaluation_timeout_s': 120.0, 'num_cpus_per_offline_eval_runner': 1,
'num_gpus_per_offline_eval_runner': 0,
'custom_resources_per_offline_eval_runner': {},
'restart_failed_offline_eval_runners': True,
'ignore_offline_eval_runner_failures': False,
'max_num_offline_eval_runner_restarts': 1000,
'offline_eval_runner_restore_timeout_s': 1800.0,
'max_requests_in_flight_per_offline_eval_runner': 1,
'validate_offline_eval_runners_after_construction': True,
'offline_eval_runner_health_probe_timeout_s': 30.0,
'offline_eval_rl_module_inference_only': False,
'broadcast_offline_eval_runner_states': False,
'offline_eval_batch_size_per_runner': 256, 'dataset_num_iters_per_eval_runner':
1, 'keep_per_episode_custom_metrics': False,
'metrics_episode_collection_timeout_s': 60.0,
'metrics_num_episodes_for_smoothing': 100, 'min_time_s_per_iteration': None,
'min_train_timesteps_per_iteration': 0, 'min_sample_timesteps_per_iteration': 0,
'log_gradients': False, 'export_native_model_files': False,
'checkpoint_trainable_policies_only': False, 'logger_creator': None,
'logger_config': None, 'log_level': 'WARN', 'log_sys_usage': True,
'fake_sampler': False, 'seed': None, 'restart_failed_env_runners': True,
'ignore_env_runner_failures': False, 'max_num_env_runner_restarts': 1000,
'delay_between_env_runner_restarts_s': 60.0, 'restart_failed_sub_environments':
False, 'num_consecutive_env_runner_failures_tolerance': 100,
'env_runner_health_probe_timeout_s': 30.0, 'env_runner_restore_timeout_s':
1800.0, '_model_config': {}, '_rl_module_spec': None,
'algorithm_config_overrides_per_module': {}, '_per_module_overrides': {},
'_validate_config': True, '_use_msgpack_checkpoints': False,
'_torch_grad_scaler_class': None, '_torch_lr_scheduler_classes': None,
'_tf_policy_handles_more_than_one_loss': False, '_disable_preprocessor_api':
False, '_disable_action_flattening': False,
'_disable_initialize_loss_from_dummy_batch': False,
'_dont_auto_sync_env_runner_states': False, 'env_task_fn': -1,
'enable_connectors': -1, 'simple_optimizer': False, 'policy_map_cache': -1,
'worker_cls': -1, 'synchronize_filters': -1, 'enable_async_evaluation': -1,
'custom_async_evaluation_function': -1, '_enable_rl_module_api': -1,
'auto_wrap_old_gym_envs': -1, 'always_attach_evaluation_results': -1,
'replay_sequence_length': None, '_disable_execution_plan_api': -1, 'use_critic':
True, 'use_gae': True, 'use_kl_loss': True, 'kl_coeff': 0.1, 'kl_target': 0.02,
'vf_loss_coeff': 1.0, 'entropy_coeff': 0.005, 'clip_param': 0.2,
'vf_clip_param': 10.0, 'entropy_coeff_schedule': None, 'lr_schedule': None,
'sgd_minibatch_size': -1, 'vf_share_layers': -1, 'class': <class
'ray.rllib.algorithms.ppo.ppo.PPOConfig'>, 'lambda': 1.0, 'input': 'sampler',
'policies': {'default_policy': (None, None, None, None)}, 'callbacks': <class
'ray.rllib.callbacks.callbacks.RLlibCallback'>, 'create_env_on_driver': True,
'custom_eval_function': None, 'framework': 'torch'}, 'time_since_restore':
3231.6118206977844, 'iterations_since_restore': 81, 'perf': {'cpu_util_percent':
np.float64(55.656140350877195), 'ram_util_percent':

```
np.float64(77.83157894736841)}})
[Iter   81] train_reward=    4.84, train_len= 103.9
[Iter   82] train_reward=    4.41, train_len= 107.5
[Iter   83] train_reward=    4.50, train_len= 106.2
[Iter   84] train_reward=    5.08, train_len= 101.2
[Iter   85] train_reward=    5.24, train_len= 100.2
[Iter   86] train_reward=    5.07, train_len=  96.9
[Iter   87] train_reward=    4.16, train_len= 101.6
[Iter   88] train_reward=    4.29, train_len= 103.3
[Iter   89] train_reward=    4.47, train_len= 102.8
[Iter   90] train_reward=    4.79, train_len= 110.5 | eval_return=   -0.56,
eval_len= 200.0 | catch_rate=  0.0%
[Iter   91] train_reward=    3.92, train_len= 115.0
[Iter   92] train_reward=    4.20, train_len= 109.3
[Iter   93] train_reward=    3.80, train_len= 111.3
[Iter   94] train_reward=    4.23, train_len= 108.4
[Iter   95] train_reward=    4.39, train_len= 108.5
[Iter   96] train_reward=    4.78, train_len= 102.9
[Iter   97] train_reward=    4.60, train_len= 101.7
[Iter   98] train_reward=    4.70, train_len=  99.6
[Iter   99] train_reward=    4.07, train_len= 103.2
[Iter  100] train_reward=    3.93, train_len=  99.7 | eval_return=    1.34,
eval_len= 166.8 | catch_rate= 10.0%
[Iter  101] train_reward=    4.25, train_len= 102.0
[Iter  102] train_reward=    3.95, train_len= 105.2
[Iter  103] train_reward=    3.95, train_len= 100.9
[Iter  104] train_reward=    4.65, train_len=  93.6
[Iter  105] train_reward=    4.19, train_len= 101.8
[Iter  106] train_reward=    4.26, train_len= 103.9
[Iter  107] train_reward=    3.98, train_len= 105.6
[Iter  108] train_reward=    4.02, train_len= 108.6
[Iter  109] train_reward=    4.79, train_len=  93.7
[Iter  110] train_reward=    4.87, train_len=  85.6 | eval_return=   -2.94,
eval_len= 200.0 | catch_rate=  5.0%
[Iter  111] train_reward=    4.81, train_len=  90.3
[Iter  112] train_reward=    4.58, train_len=  99.4
[Iter  113] train_reward=    4.47, train_len=  98.0
[Iter  114] train_reward=    4.39, train_len=  91.6
[Iter  115] train_reward=    4.53, train_len=  94.6
[Iter  116] train_reward=    4.94, train_len=  93.1
[Iter  117] train_reward=    4.72, train_len=  99.2
[Iter  118] train_reward=    4.35, train_len= 104.4
[Iter  119] train_reward=    4.43, train_len=  98.2
[Iter  120] train_reward=    4.44, train_len=  92.7 | eval_return=   -0.40,
eval_len= 200.0 | catch_rate= 10.0%
[Iter  121] train_reward=    5.10, train_len=  88.2
[Iter  122] train_reward=    5.40, train_len=  89.2
[Iter  123] train_reward=    5.43, train_len=  93.4
```

```
[Iter  124] train_reward=    6.33, train_len=  75.4
[Iter  125] train_reward=    5.51, train_len=  84.0
[Iter  126] train_reward=    4.46, train_len=  87.6
[Iter  127] train_reward=    4.95, train_len=  80.3
[Iter  128] train_reward=    4.89, train_len=  84.2
[Iter  129] train_reward=    5.38, train_len=  84.7
[Iter  130] train_reward=    5.61, train_len=  85.5 | eval_return=   -1.98,
eval_len= 200.0 | catch_rate=  5.0%
[Iter  131] train_reward=    4.96, train_len=  95.0
[Iter  132] train_reward=    4.29, train_len=  95.6
[Iter  133] train_reward=    4.74, train_len=  87.8
[Iter  134] train_reward=    5.55, train_len=  84.9
[Iter  135] train_reward=    6.23, train_len=  73.7
[Iter  136] train_reward=    5.74, train_len=  73.2
[Iter  137] train_reward=    5.37, train_len=  78.9
[Iter  138] train_reward=    5.82, train_len=  77.2
[Iter  139] train_reward=    5.47, train_len=  80.0
[Iter  140] train_reward=    5.43, train_len=  79.5 | eval_return=   -3.27,
eval_len= 200.0 | catch_rate=  5.0%
[Iter  141] train_reward=    6.34, train_len=  76.7
[Iter  142] train_reward=    6.65, train_len=  77.6
[Iter  143] train_reward=    6.29, train_len=  77.5
[Iter  144] train_reward=    5.88, train_len=  82.2
[Iter  145] train_reward=    6.26, train_len=  79.4
[Iter  146] train_reward=    6.27, train_len=  68.2
[Iter  147] train_reward=    6.11, train_len=  65.9
[Iter  148] train_reward=    6.33, train_len=  73.0
[Iter  149] train_reward=    6.36, train_len=  65.7
[Iter  150] train_reward=    5.64, train_len=  66.2 | eval_return=    0.84,
eval_len= 148.2 | catch_rate= 10.0%
[Iter  151] train_reward=    5.81, train_len=  69.7
[Iter  152] train_reward=    6.03, train_len=  68.7
[Iter  153] train_reward=    5.48, train_len=  76.2
[Iter  154] train_reward=    5.24, train_len=  80.1
[Iter  155] train_reward=    5.74, train_len=  75.4
[Iter  156] train_reward=    5.97, train_len=  69.0
[Iter  157] train_reward=    5.54, train_len=  72.2
[Iter  158] train_reward=    5.28, train_len=  79.0
[Iter  159] train_reward=    5.56, train_len=  70.9
[Iter  160] train_reward=    6.55, train_len=  64.6 | eval_return=   -0.60,
eval_len= 200.0 | catch_rate= 25.0%
[Iter  161] train_reward=    6.69, train_len=  69.2
[Iter  162] train_reward=    5.56, train_len=  75.5
[Iter  163] train_reward=    5.54, train_len=  76.4
[Iter  164] train_reward=    5.77, train_len=  71.9
[Iter  165] train_reward=    5.87, train_len=  69.7
[Iter  166] train_reward=    6.89, train_len=  67.6
[Iter  167] train_reward=    6.28, train_len=  73.1
```

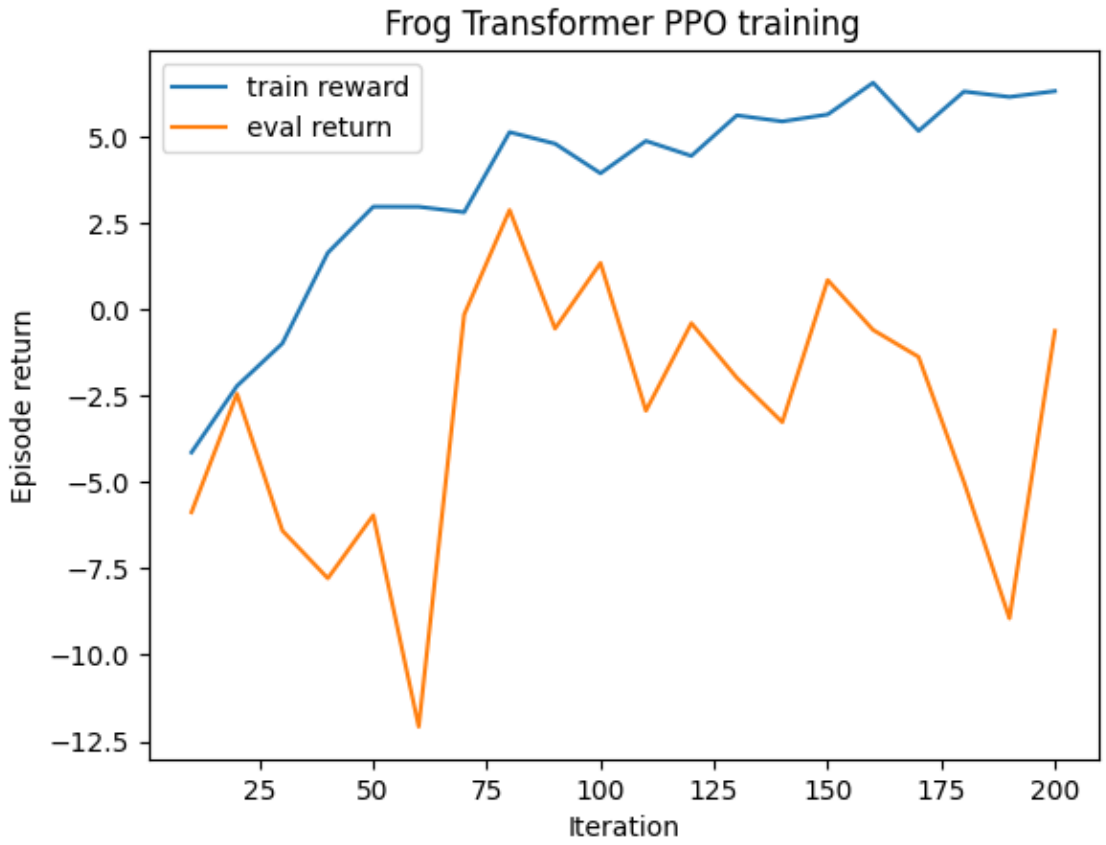
```
[Iter  168] train_reward=    5.89, train_len=  71.6
[Iter  169] train_reward=    5.84, train_len=  72.2
[Iter  170] train_reward=    5.16, train_len=  77.5 | eval_return=   -1.38,
eval_len= 200.0 | catch_rate= 20.0%
[Iter  171] train_reward=    5.85, train_len=  72.2
[Iter  172] train_reward=    6.49, train_len=  70.0
[Iter  173] train_reward=    6.33, train_len=  67.0
[Iter  174] train_reward=    6.10, train_len=  69.2
[Iter  175] train_reward=    5.99, train_len=  74.1
[Iter  176] train_reward=    6.05, train_len=  64.2
[Iter  177] train_reward=    6.47, train_len=  65.3
[Iter  178] train_reward=    6.48, train_len=  70.9
[Iter  179] train_reward=    6.26, train_len=  65.7
[Iter  180] train_reward=    6.30, train_len=  70.3 | eval_return=   -5.00,
eval_len= 200.0 | catch_rate=  5.0%
[Iter  181] train_reward=    5.96, train_len=  76.4
[Iter  182] train_reward=    5.80, train_len=  74.5
[Iter  183] train_reward=    6.23, train_len=  70.5
[Iter  184] train_reward=    6.00, train_len=  72.3
[Iter  185] train_reward=    6.04, train_len=  73.8
[Iter  186] train_reward=    5.95, train_len=  75.2
[Iter  187] train_reward=    6.04, train_len=  74.0
[Iter  188] train_reward=    5.93, train_len=  76.0
[Iter  189] train_reward=    6.13, train_len=  71.4
[Iter  190] train_reward=    6.14, train_len=  69.5 | eval_return=   -8.94,
eval_len= 200.0 | catch_rate=  0.0%
[Iter  191] train_reward=    5.85, train_len=  75.0
[Iter  192] train_reward=    5.49, train_len=  79.2
[Iter  193] train_reward=    6.04, train_len=  68.3
[Iter  194] train_reward=    5.77, train_len=  72.2
[Iter  195] train_reward=    5.35, train_len=  77.8
[Iter  196] train_reward=    6.11, train_len=  70.7
[Iter  197] train_reward=    6.08, train_len=  67.5
[Iter  198] train_reward=    6.31, train_len=  65.9
[Iter  199] train_reward=    6.48, train_len=  58.0
[Iter  200] train_reward=    6.31, train_len=  67.5 | eval_return=   -0.63,
eval_len= 197.8 | catch_rate=  0.0%
```

[12]:
```python
import pandas as pd
import matplotlib.pyplot as plt

df_hist = pd.DataFrame(train_history)

plt.figure()
plt.plot(df_hist["iter"], df_hist["train_reward"], label="train reward")
plt.plot(df_hist["iter"], df_hist["eval_return"], label="eval return")
plt.xlabel("Iteration")
```

```
plt.ylabel("Episode return")
plt.legend()
plt.title("Frog Transformer PPO training")
plt.show()
```



[ ]:
```
import numpy as np
import matplotlib.pyplot as plt
from matplotlib import animation
from mpl_toolkits.mplot3d import Axes3D
import imageio

# ------------------------------------------------------------
# 1. UNWRAP ENV (in case it's inside ObsHistoryWrapper)
# ------------------------------------------------------------
def unwrap_env(env):
    """Return underlying FrogFly3DEnv even if wrapped."""
    while hasattr(env, "env"):
        env = env.env
    return env
```

```python
# ------------------------------------------------------------
# 2. ROLLOUT AND RECORD POSITIONS
# ------------------------------------------------------------
def rollout_trajectory(env, algo, max_steps=200):
    raw_env = unwrap_env(env)

    obs, info = env.reset()

    frog_traj = [raw_env.frog_pos.copy()]
    fly_traj  = [raw_env.fly_pos.copy()]

    for _ in range(max_steps):
        action, _, _ = algo.compute_single_action(obs, explore=False)
        obs, reward, terminated, truncated, info = env.step(action)

        frog_traj.append(raw_env.frog_pos.copy())
        fly_traj.append(raw_env.fly_pos.copy())

        if terminated or truncated:
            break

    return np.array(frog_traj), np.array(fly_traj)

# ------------------------------------------------------------
# 3. CREATE 3D ANIMATION
# ------------------------------------------------------------
def make_3d_gif(frog_traj, fly_traj, gif_path="transformer_original_3d.gif"):
    fig = plt.figure(figsize=(6, 6))
    ax = fig.add_subplot(111, projection='3d')

    # initial points
    frog_dot, = ax.plot([], [], [], 'go', markersize=8)
    fly_dot,  = ax.plot([], [], [], 'ro', markersize=8)

    # history lines
    frog_path, = ax.plot([], [], [], 'g-', linewidth=2, alpha=0.7)
    fly_path,  = ax.plot([], [], [], 'r-', linewidth=2, alpha=0.7)

    # axis limits
    ax.set_xlim([-1.0, 1.0])
    ax.set_ylim([-1.0, 1.0])
    ax.set_zlim([-1.0, 1.0])
    ax.set_title("Frog-Fly 3D Trajectory (Original Transformer PPO)")
    ax.set_xlabel("X")
    ax.set_ylabel("Y")
    ax.set_zlabel("Z")
```

```
    def init():
        frog_dot.set_data([], [])
        frog_dot.set_3d_properties([])
        fly_dot.set_data([], [])
        fly_dot.set_3d_properties([])
        frog_path.set_data([], [])
        frog_path.set_3d_properties([])
        fly_path.set_data([], [])
        fly_path.set_3d_properties([])
        return frog_dot, fly_dot, frog_path, fly_path

    def update(i):
        # --- frog point ---
        frog_dot.set_data([frog_traj[i, 0]], [frog_traj[i, 1]])
        frog_dot.set_3d_properties([frog_traj[i, 2]])

        # --- fly point ---
        fly_dot.set_data([fly_traj[i, 0]], [fly_traj[i, 1]])
        fly_dot.set_3d_properties([fly_traj[i, 2]])

        # --- frog path ---
        frog_path.set_data(frog_traj[:i+1, 0], frog_traj[:i+1, 1])
        frog_path.set_3d_properties(frog_traj[:i+1, 2])

        # --- fly path ---
        fly_path.set_data(fly_traj[:i+1, 0], fly_traj[:i+1, 1])
        fly_path.set_3d_properties(fly_traj[:i+1, 2])

        return frog_dot, fly_dot, frog_path, fly_path

    ani = animation.FuncAnimation(
        fig, update, init_func=init,
        frames=len(frog_traj),
        interval=60,
        blit=False  # IMPORTANT for 3D
    )

    ani.save(gif_path, writer="pillow", fps=20)

    plt.close(fig)
    print(f"Saved 3D GIF to: {gif_path}")

# ------------------------------------------------------------
# 4. RUN IT
# ------------------------------------------------------------
```

```
# STEP A - create a fresh evaluation env (unwrapped)
eval_env = FrogFly3DEnv(env_config)

# STEP B - rollout using the trained algorithm "algo"
frog_traj, fly_traj = rollout_trajectory(eval_env, algo)

# STEP C - render the GIF
make_3d_gif(frog_traj, fly_traj, gif_path="transformer_original_3d.gif")
```

```
Saved 3D GIF to: transformer_original_3d.gif
```

```
(raylet) [2025-12-02 15:59:21,336 E 2877486 2877486] (raylet)
node_manager.cc:3277: 2 Workers (tasks / actors) killed due to memory pressure
(OOM), 0 Workers crashed due to other reasons at node (ID:
bb845cb40520f1b50aaa49aaa0b1c0f7d0c4ee4cc6af62ca1b229b4f, IP: 192.168.86.52)
over the last time period. To see more information about the Workers killed on
this node, use `ray logs raylet.out -ip 192.168.86.52`
(raylet)
(raylet) Refer to the documentation on how to address the out of memory
issue: https://docs.ray.io/en/latest/ray-core/scheduling/ray-oom-
prevention.html. Consider provisioning more memory on this node or reducing task
parallelism by requesting more CPUs per task. To adjust the kill threshold, set
the environment variable `RAY_memory_usage_threshold` when starting Ray. To
disable worker killing, set the environment variable
`RAY_memory_monitor_refresh_ms` to zero.
```