Hardmaru:

| imesteps\_per\_actorbatch=4096, clip\_param=0.2, entcoeff=0.0, optim\_epochs=10,  optim\_stepsize=3e-4,  optim\_batchsize=64,  gamma=0.99,  lam=0.95,  schedule='linear',  verbose=2 |  |
| --- | --- |
|  |  |

**learning\_rate** = trial.suggest\_categorical("learning\_rate", [0.0003, 0.0007, 0.00096])

**lr\_schedule** = trial.suggest\_categorical('lr\_schedule', ['linear', 'constant'])

batch\_size = trial.suggest\_categorical("batch\_size", [32, 64, 128])

n\_steps = trial.suggest\_categorical("n\_steps", [512, 2048, 4096])

gamma = trial.suggest\_categorical("gamma", [0.95, 0.99, 0.999])

ortho\_init = True

**learning\_rate** = trial.suggest\_categorical("learning\_rate", [0.0003, 0.0007, 0.00096])

**lr\_schedule** = trial.suggest\_categorical('lr\_schedule', ['linear', 'constant'])

ent\_coef = trial.suggest\_categorical("ent\_coef", [0.0001, 0.0])

clip\_range = trial.suggest\_categorical("clip\_range", [0.1, 0.2])

n\_epochs = trial.suggest\_categorical("n\_epochs", [5, 10, 20])

gae\_lambda = trial.suggest\_categorical("gae\_lambda", [0.92, 0.95, 0.98])

ortho\_init = True

**learning\_rate** = trial.suggest\_categorical("learning\_rate", [0.0003, 0.0007, 0.00096])

**lr\_schedule** = trial.suggest\_categorical('lr\_schedule', ['linear', 'constant'])

vf\_coef = trial.suggest\_categorical("vf\_coef", [0, 0.5, 1])

activation\_fn = trial.suggest\_categorical("activation\_fn", ["tanh", "relu"])

ortho\_init = True

net\_arch = trial.suggest\_categorical("net\_arch", ["small", "medium"])

# lr\_schedule = "constant"

# Uncomment to enable learning rate schedule

# max\_grad\_norm = trial.suggest\_categorical("max\_grad\_norm", [0.5])

# Uncomment for gSDE (continuous actions)

# log\_std\_init = trial.suggest\_uniform("log\_std\_init", -4, 1)

# Uncomment for gSDE (continuous action)

# sde\_sample\_freq = trial.suggest\_categorical("sde\_sample\_freq", [-1, 8, 16, 32, 64, 128, 256])

# Orthogonal initialization

# ortho\_init = trial.suggest\_categorical('ortho\_init', [False, True])

# activation\_fn = trial.suggest\_categorical('activation\_fn', ['tanh', 'relu', 'elu', 'leaky\_relu'])