

new approach

state size: 2
reward size: 6
action size: 2

Initialize environment
(FTN)

Repeat for M
episodes

state →
1. define random w
2. state, w >> model_ >> Q for
each action
3. wQ (scalarize Q)
4. chose the action that
maximizes wQ
→ action

action >> env.step >> next_state,
reward, terminal

memorize them in a buffer and assign
probability for each trajectory
(experienced prioritized replay)

generate random weights for each objective with
mean 0 and 1 variance

$$act = \max_a(tmpQ.w)$$

$$HQ = \arg_{DQ}(act)$$

The DC of an action that maximized

$$tmpQ.w$$

$$wTQ = w.TauQ$$

$$loss = \beta MSE(wQ, wTQ) = (1 - \beta)MSE(Q, TauQ)$$

The loss is used to train the model which outputs
tmpQ, and then after 100 updates, target model is
update:

target model = model

Proposed approach

state size: 2
reward size: 6
action size: 2

Initialize environment
(FTN)

Repeat for M
episodes

state →
1. define random w
2. state, w >> model_ >> Q for
each action
3. wQ (scalarize Q)
4. chose the action that
maximizes wQ
→ action

action >> env.step >> next_state,
reward, terminal

memorize them in a buffer and assign
probability for each trajectory
(experienced prioritized replay)

Repeat for N
iterations

update-mu = 0

N: inside loops

M: number of objectives

losslist = []

if update-mu = 0: {
generate random weights for each objective
with mean mu}

if update-mu = 1: {
mutemp = np.mean(losslist) >>> to find mean of
the list of the past loss values in N past iterations.
for each objective :

$$mu[i] = \frac{\exp(\beta mutemp[i])}{\sum_{j=0}^M \exp(\beta mutemp[j])}$$

define random weight for each objective
with mu[i] as mean for the random
normal.}

$$act = \max_a(tmpQ.w)$$

$$HQ = \arg_{DQ}(act)$$

The DC of an action that maximized

$$tmpQ.w$$

$$wTQ = w.TauQ$$

$$loss = \alpha MSE(wQ, wTQ) + (1 - \alpha)MSE(Q, TauQ)$$

save the loss function (the second segment with
no w) in a list called losslist.

after N iterations of the list update-mu = 1

The loss is used to train the model which outputs tmpQ ,
and then after 100 updates, target model is updated:

target model = model