PRIORITISED EXPERIENCE REPLAY

Observation: **on policy** learning means that data is discarded right after learning

This is **WASTEFUL**

Idea: moving to **off policy** learning:

keep data collected in a replay buffer

So it can be used several times

V0: EXPERIENCE REPLAY

use a data structure (a **replay buffer**) to store a fixed (large) number of transitions collected from the policy

→ older transitions get thrown away

→ learning: pick at random a batch of transitions

Issue: not all transitions are equally interesting !

Theorem: sampling uniformly from the buffer yields unbiased estimates for Q

V1: PRIORITISED SAMPLING

when adding a transition (s, a, r, s')

store its **priority**

$$\text{TD error} = r + \gamma \max_{a' \in A} Q(s', a') - Q(s, a)$$

Naive version:

$$\text{priority} = |\text{TD error}| + \varepsilon$$

Better version:

initially: $\text{priority} = \max_{\text{buffer}} \text{priority}$

to ensure using latest transitions once

Sampling: probability = normalised priorities

$$\text{proba}_i = \frac{\text{priority}_i}{\sum_j \text{priority}_j}$$

Remark: after $\text{proba}_i = \frac{\text{priority}_i^{\alpha}}{\sum_j \text{priority}_j^{\alpha}}$

$\alpha \in [0,1]$ hyperparameter

Learning

we sampled transitions $t_1, \dots, t_B$ (B = batch)

(1) calculate current TD error:

$$\text{TD error} = r + \gamma \max_{a' \in A} Q(s', a') - Q(s, a)$$

(2) compute the loss

$$\mathcal{L}(\theta) = \frac{1}{B} \sum_{i=1}^{B} (\text{TD error}_i)^2$$

(3) apply optimisation step (backward propagation)

(4) update priorities for $t_1, \dots, t_B$

$$\text{priority} = |\text{TD error}| + \varepsilon$$

Better: we use more interesting transitions

BUT: this introduces a bias which invalidates the original theorem:

Theorem: sampling uniformly from the buffer yields unbiased estimates for Q

V2 : PRIORITISED SAMPLING WITH IMPORTANCE SAMPLING

Uniform sampling → Unbiased estimate for Q

prioritised sampling → selection of good data but biased

Typical scenario for IMPORTANCE SAMPLING

APARTÉ : IMPORTANCE SAMPLING

target distribution $p(x)$ ← we are interested in p

proposal distribution $q(x)$ ← we can sample from q

Goal : $\mu = \mathop{\mathbb{E}}_{x \sim p}[f(x)]$

Fact : $\mu = \mathop{\mathbb{E}}_{x \sim q}\left[\frac{p(x)}{q(x)} f(x)\right]$

Concretely : sample $x_i \sim q(x)$ N times

and estimate :

$$\mu \simeq \frac{1}{N} \sum_{i=1}^{N} w_i f(x_i)$$

$w_i = \frac{p(x_i)}{q(x_i)}$ is called importance weight

Now, let's apply **IMPORTANCE SAMPLING**

target distribution: uniform (over buffer)

$$U_i = \frac{1}{N} \qquad N \text{ buffer size}$$

proposal distribution:

$$P_i = \frac{\text{priority}_i^{\alpha}}{\sum_j \text{priority}_j^{\alpha}}$$

Goal: $$\mathcal{L}(\theta) = \mathbb{E}\left[(\text{TD-error})^2\right] \simeq \frac{1}{B}\sum_{i=1}^{B} \delta_i^2$$

$$w_i = \frac{U_i}{P_i} = \frac{1}{N}\cdot\frac{1}{P_i}$$

$$w_i^{\text{normalised}} = \frac{w_i}{\max_k w_k}$$

New Loss: $$\frac{1}{B}\sum_{i=1}^{B} w_i^{\text{normalised}}\,\delta_i^2$$

Bonus: $$w_i = \left(\frac{1}{N}\frac{1}{P_i}\right)^{\beta}$$

$\beta \in [0,1]$ bias correction

β annealed from 0.4 to 1.0

FULL ALGORITHM

ADD TO THE BUFFER:

$$\text{priority} = \max_{\text{buffer}} \text{priority}$$

SAMPLING:

$$P_i = \frac{\text{priority}_i^{\alpha}}{\sum_j \text{priority}_j^{\alpha}}$$

$\alpha \in [0,1]$ hyperparameter

LEARNING:

we sampled transitions $t_1, \dots, t_B$ (B = batch)

(1) calculate current TD error:

$$\text{TD error} = r + \gamma \max_{a' \in A} Q(s', a') - Q(s, a)$$

(2) compute the importance weights

$$w_i = \left(\frac{1}{N} \cdot \frac{1}{P_i}\right)^{\beta} \qquad w_i^{\text{normalised}} = \frac{w_i}{\max_k w_k}$$

(3) compute the loss

$$\mathcal{L}(\theta) = \frac{1}{B} \sum_{i=1}^{B} w_i^{\text{normalised}} \, \delta_i^2$$

(4) apply optimisation step (backward propagation)

(5) update priorities for $t_1, \ldots, t_B$

$$\text{priority} = |\text{TD error}| + \varepsilon$$