Speeding up Reinforcement Learning-based Information Extraction Training using Asynchronous Methods : APPENDIX

Aditya Sharma
Indian Institute of Science
Bangalore, India
adisharma075@gmail.com

Zarana Parekh*
DA-IICT
Gandhinagar, India
zaranaparekh17@gmail.com

Partha Talukdar
Indian Institute of Science
Bangalore, India
ppt@iisc.ac.in

1 Experimental Setup

For RLIE-DQN and RLIE-A3C, all experiments were carried out on an 18 core Intel Xeon E5-2666 v3 (Haswell) CPU.

2 Hyperparameter Values

For RLIE-A3C, we set discount factor $\gamma = 0.8$, and the entropy related hyperparameters β_d and β_q to 0.05 and 0.01, respectively. The learning rates are sampled from $\mathrm{LogUniform}(10^{-4}, 10^{-2})$. The global network is updated after every 500 steps.

3 RLIE-A3C: Psuedocode for each parallel agent

The pseudocode for each parallel agent of RLIE-A3C is described in Algorithm 1.

Algorithm 1 Asynchronous advantage actor-critic - pseudocode for each parallel agent p

$x \in \{d, q\}$
// Assume global shared parameter vectors θ_x and θ_v and global shared counter $T = 0$
// Assume thread-specific parameter vectors θ_x^p and θ_v^p for parallel agent p
Initialize thread step counter $t \leftarrow 1$
repeat
  Reset gradients: $d\theta_x \leftarrow 0$ and $d\theta_v \leftarrow 0$.
  Synchronize thread-specific parameters $\theta_x^p = \theta_x$ and $\theta_v^p = \theta_v$
  $t_{start} = t$
  Get state s_t
  repeat
    Perform a_{dt} according to policy $\pi_d(a_{dt}|s_t; \theta_d^p)$
    Perform a_{qt} according to policy $\pi_q(a_{qt}|s_t; \theta_q^p)$
    Receive reward r_t and new state s_{t+1}
    $t \leftarrow t + 1$
    $T \leftarrow T + 1$
  until terminal s_t **or** $t - t_{start} == t_{max}$
  $R = \begin{cases} 0 & \text{for terminal } s_t \\ V(s_t, \theta_v^p) & \text{for non-terminal } s_t \text{//from last state} \end{cases}$
  for $i \in \{t-1, \ldots, t_{start}\}$ **do**
    $G_t \leftarrow r_i + \gamma V(s_t, \theta_v^p)$
    $A(s_t, a_t; \theta_v) = \sum_{j=0}^{t_{max}-1} \gamma^j R_{t+j} + \gamma^{t_{max}} V(s_{t+t_{max}}; \theta_v) - V(s_t; \theta_v)$
    Accumulate gradients wrt θ_x^p: $d\theta_x^p \leftarrow d\theta_x^p + \nabla_{\theta_x^p} \log \pi_d(a_{x_i}, s_i; \theta_x^p) A(s_i, a_{x_t}; \theta_v^p) + \beta_x \nabla_{\theta_x^p} H(\pi_x(s_t; \theta_x^p))$
    Accumulate gradients wrt $d\theta_v^p \leftarrow d\theta_v^p + \partial (G_t - V(s_t; \theta_v^p))^2 / \partial \theta_v^p$
  end for
  Perform asynchronous update of θ_d using $d\theta_d^p$, of θ_q using $d\theta_q^p$ and of θ_v using $d\theta_v^p$.
until $T > T_{max}$

*Research carried out during an internship at the Indian Institute of Science, Bangalore.