Optimal decision making for complex problems

Transfer learning in reinforcement learning problems

Benoît Umé

University of Liège

Problem

- Set of m similar environment $E_1, E_2, ..., E_m$
- E_i defined by deterministic discrete-time system:

$$\begin{aligned} x_{t+1} &= f_i(x_t, u_t)\ , \\ r_t &= r_i(x_t, u_t)\ , \qquad t = 0, 1, 2, 3, \ldots \end{aligned} \tag{1}$$

  where $1 \leq i \leq m$.
- Each E_i has a pre-trained Q functions $Q_i(x, u)\ \ \forall i \in [1, m-1]$
- We can perform a most d_m interactions with E_m

Problem

- Using $Q_1, ..., Q_{m-1}$ and d_m interactions with E_m can we find a fusion operator $\oplus$ such that:

$$Q_m = \oplus(Q_1, ..., Q_{m-1}, E_m, d_m) \tag{2}$$

  and

$$J_m(Q_m) \geq J_m(Q_0), J_m(Q_1), ..., J_m(Q_{m-1}) \tag{3}$$

- Q_0 is a Q function trained from scratch on E_m using d_m interactions
- $J_m(Q)$ is the expected return, on E_m, of the policy derived from Q.

Solution

- Transfer learning using deep neural network
- $Q_1, ..., Q_m$ can be represented by neural networks ($N_1, ..., N_m$).

First idea

- Concatenate all models with an output interface then train this new model on E_m using d_m interactions.

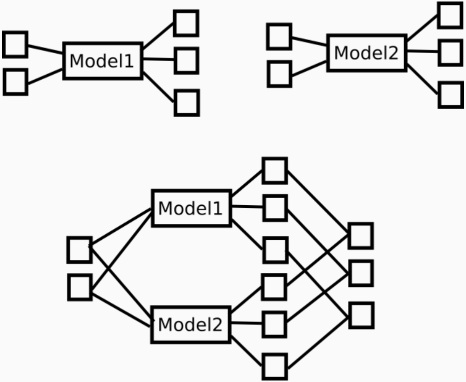

Figure 1: Fusion schematic

Protocol

- N_m is the result model.
- N_0 is a neural network initialized with random values and trained on E_m with d_m interaction, using standard Q learning algorithm.
- For each N_i, we play 100 times on E_m and compute the mean cumulative reward.
- If mean cumulative reward from N_m is greater than all other, the fusion operator is accepted.

Preliminary results

- OpenAI, Mountain Car environment:

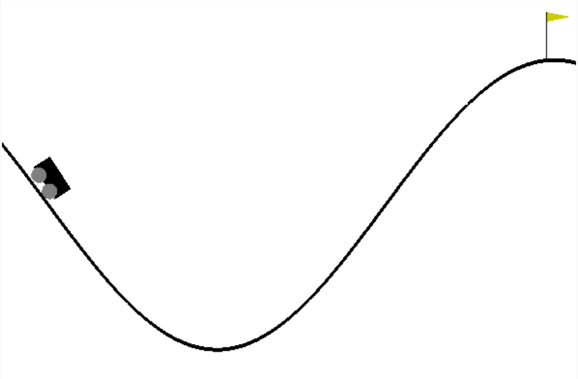

Figure 2: Environment view

Preliminary results

- Using $d_m = 10000$
- N_1: Smaller mountain (Score -200)
- N_2: Lighter car (Score -156)
- N_0: (Score -200)
- N_3 with N_1 and N_2 freezed (Score -200)
- N_3 without freezing (Score -125)
- N_3 full connected output (Score -200)
- N_3 full connected input/output (Score -200)