## Abstract


- First DL model to learn control policies directly from high-D sensory i/p using RL.
- CNN trained with variant of Q-learning.
- Atari 2600 computer games.
- Only raw pixels are input.


---

## 1) Introduction.

- DL generally require large hand-labelled data.
- RL algorithm should learn from scalar reward signals that are very sparse / noisy / delayed.
- DL requires data samples to be independent. RL have correlated data.
- In RL, distribution changes as the algorithm learns new behaviors. DL assumes a fixed distribution.
- Tested on Atari 2600 - (210 x 160 RGB video at 60Hz)
- Good is to create single NN agent that learns to play as many games as possible.

---

## 2) Background.

$A = \{1 \dots K\}$ actions at, environment $\mathcal{E}$.
↓
may be stochastic

$a_t \longrightarrow \mathcal{E} \Rightarrow r_t$ (reward), $x_t$ (image)

$s_t = \{x_1, a_1, x_2, \dots a_{t-1}, x_t\}$ a sequence is a distinct state. This sequence is assumed to terminate

$$R_t = \sum_{t'=t}^{T} \gamma^{t'-1} r_{t'}$$ (goes from $t$ to completion)

$$Q^*(s,a) = \max_{\pi} E\left[R_t \mid s_t = s, a_t = a, \pi\right]$$

$Q^*$: optimal action value function

$$y_i = E\left[r + \gamma \max_{\theta_{i-1}} Q(s', a')\right] \quad \text{- target}$$

$$L_i(\theta_i) = E\left[(y_i - Q_{\theta_i}(s,a))^2\right]$$

↓ the parameters from the previous iteration is held fixed while optimising the loss function.

$$\nabla L_i(\theta_i) = E\left[\left(r + \gamma \max_{\theta_{i-1}} Q_{\theta_{i-1}}(s', a') - Q_{\theta_i}(s,a)\right) \nabla_{\theta_i} Q_{\theta_i}(s,a)\right]$$

$\theta_{i-1}$: target; $\theta_i$: policy

model-free: directly samples from $\mathcal{E}$, without explicitly estimating $\mathcal{E}$.

off-policy: learns greedy strategy $\max_a Q(s,a)$ with $\epsilon$-greedy exploration.

---

3.) Related work

→ TD-gammon - model free RL - Q learning with 1 hidden layer NN. Later chess / Go / checkers using the same approach was unsuccessful - [may be it worked for backgammon because the stochasticity induced by dice rolls helped in exploration and made the value function smooth.]

→ Non-linear function approximator + model-free (Q-learning) ⇒ Q network diverged. ⇒ Majority of the work focused on linear-fn-approximators.

→ Neural Fitted Q (NFQ) - Solo author paper

NFQ vs DQL ?

---

4) Deep Reinforcement learning

- Goal: connect RL algorithm to a DNN which operates directly on images.
- Experience-replay - Simple author paper

At each time step $e_t = (s_t, a_t, r_t, s_{t+1})$

$D = [e_1, e_2 \dots e_N]$

pooled over many episodes