REVISION, RL

MARKOV = STATE TRANSITIONS DEPEND ONLY ON CURRENT STATE, NOT PAST

STATIONARY = TRANSITION PROBABILITIES DON'T CHANGE OVER TIME

MARKOV REWARD PROCESS = MRP = (S, P (TRANSITION PROBABILITY), R, γ)

RETURN = TOTAL DISCOUNTED REWARD FROM TIME, $t = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$

STATE VALUE FUNCTION, $V^\pi(s)$ = EXPECTED RETURN FROM S, $V = R_s + \gamma \sum P_{ss'} V(s')$

DETERMINISTIC POLICY = ONLY ONE ACTION POSSIBLE

MARKOV DECISION PROCESS = MDP = $(S, A, P_{ss'}, \gamma, R_{ss'}, \overset{\text{POLICY}}{\pi})$

STATE-ACTION VALUE FUNCTION = $Q^\pi(s,a) = \mathbb{E}[R_t | s, A]$ $\quad V^\pi(s) = \sum_a \pi(s,a) Q^\pi(s,a)$

BELLMAN OPTIMALITY EQUATION → $V^*(s) = \max_a \sum_{s'} P_{ss'}(R^a_{ss'} + \gamma V^*(s'))$

DYNAMIC PROGRAMMING = COMPUTE OPTIMAL POLICY GIVEN A MODEL
- BREAK UP THE PROBLEM INTO SMALLER IDENTICAL PROBLEMS

POLICY ITERATION = EVALUATE (VALUE OF) POLICY UNTIL CONVERGENCE ⇄ UPDATE POLICY
- MULTIPLE EVALUATIONS ARE EXPENSIVE

VALUE ITERATION = EVALUATE POLICY ONCE ⇄ UPDATE POLICY
- CAN BE VIEWED AS EVALUATION W/O POLICY, UPDATE WITH OPTIMAL OR GREEDY ACTION

SYNCHRONOUS BACKUPS = ALL STATES DONE AT ONCE, 2 COPIES OF VAL FUNC. = EXPENSIVE

ASYNCHRONOUS BACKUPS = DO SELECTED STATES, 1 COPY OF VAL FUNC = CHEAPER

BOOTSTRAPPING = UPDATE ESTIMATES BASED ON OTHER ESTIMATE — { LESS VARIANCE, MORE BIAS

MODEL-FREE LEARNING

MONTE CARLO - GET FULL TRACE OF EXPERIENCES, THEN UPDATE
- NO BOOTSTRAPPING
- MUST BE EPISODIC WITH TERMINAL STATES
- CAN AVERAGE EVERY VISIT OR FIRST VISIT (PER TRACE)
- DOES NOT EXPLOIT MARKOV PROPERTY ⇒ MORE EFFECTIVE IN NON-MARKOV ENV.
- MC CAN UPDATE AFTER EACH TRACE OR MC CAN BATCH THE TRACES

CAN AVERAGE ALL VISITS EQUALLY $V(s) \leftarrow V(s) + \frac{1}{N}(R - V(s))$

OR GRADUALLY FORGET OLD EPISODES $V(s) \leftarrow V(s) + \alpha(R - V(s))$ GOOD FOR NON-STATIONARY ENV

TEMPORAL DIFFERENCE = UPDATE AT EACH STEP, BOOTSTRAPPING { LESS VARIANCE, MORE BIAS

$$V(s_t) \leftarrow V(s_t) + \alpha(\underbrace{\underbrace{r_{t+1} + \gamma V(s_{t+1})}_{\text{TD TARGET}} - V(s_t)}_{\text{TD ERROR}})$$

SOFT POLICY = FINITE PROBABILITY TO EXPLORE ALL STATES

EG. ϵ-GREEDY = RANDOM ϵ% OF THE TIME, OTHERWISE OPTIMAL

ϵ-GREEDY IS GLIE IF ϵ REDUCES TO ZERO $\epsilon_k = \frac{1}{k}$

GLIE = GREEDY IN THE LIMIT W/ INFINITE EXPLORATION
- ALL STATE/ACTION PAIRS EXPLORED INFINITELY MANY TIMES
- CONVERGES TO GREEDY POLICY

EXPLORING STARTS = START EPISODES WITH RANDOM STATE/ACTION, EXPLORE ALL

ON-POLICY = OPTIMIZE THE POLICY YOU ARE FOLLOWING

OFF-POLICY = OPTIMIZE A POLICY WHILE FOLLOWING A DIFFERENT ONE

π = TARGET POLICY $\quad \pi'$ = BEHAVIOUR POLICY

$Q(s,a) \leftarrow Q(s,a) + \alpha(r + \gamma Q(s',a') - Q(s,a))$

SARSA = ON-POLICY TD

SARSA CONVERGES TO OPTIMAL IF 1) POLICY, π_t IS GLIE

2) ROBBINS-MONRO STEP-SIZES FOR α_t $\begin{cases} \sum^{\infty} \alpha_t = \infty \\ \sum^{\infty} \alpha_t^2 < \infty \end{cases}$, EG $\frac{1}{2^k}$, NOT $\frac{1}{k}$

IF SPARSE REWARDS USE SARSA-LAMBDA

IF SPARSE REWARDS USE SARSA-LAMBDA = AVERAGE REWARDS OF ALL TRACES, FROM LENGTH = 1, 2, 3 ... TERMINATION

Q-LEARNING = OFF-POLICY TD, BC π IS GREEDY, π' IS ϵ-GREEDY

$Q(s,a) \leftarrow Q(s,a) + \alpha(r + \gamma \max_{a'} Q(s',a') - Q(s,a))$

UPDATE EACH STEP

		FULL BACKUP (DP)	SAMPLE BACKUP (TD)
PREDICTION	BELLMAN EXPECT. EQ. $V_\pi(s)$	ITERATIVE POLICY EVALUATION $V(s) \leftarrow E[R + \gamma V(s') \mid s]$	TD LEARNING $V(s) \leftarrow V(s) + \alpha[R + \gamma V(s') - V(s)]$
CONTROL	BELLMAN EXPECT. EQ $Q_\pi(s,a)$	CONVERGE VALUE, THEN ADJUST POLICY Q-POLICY ITERATION $Q(s,a) \leftarrow E[R + \gamma Q(s',A') \mid s,a]$	SARSA $Q(S,A) \leftarrow Q(S,A) + \alpha[R + \gamma Q(S',A') - Q(S,A)]$
CONTROL	BELLMAN OPTIMALITY EQ $Q^*(s,a)$	ALTERNATE EVAL W/ POLICY IMPROVEMENT Q-VALUE ITERATION $Q(s,a) \leftarrow E[R + \gamma \max_{a'} Q(s',a') \mid s,a]$	Q-LEARNING $Q(S,A) \leftarrow Q(S,A) + \alpha[R + \gamma \max_{A'} Q(S',A') - Q(S,A)]$

FUNCTION APPROXIMATION - GOOD FOR LARGE OR CONTINUOUS STATE SPACE

$V^\pi(s), Q^\pi(s,a) \approx \hat{V}(s,w), \hat{Q}(s,a,w)$ W APPROXIMATES STATES $\Big\{$ UPDATE UNSEEN STATES FROM SEEN

USE DIFFERENTIABLE APPROX. UPDATE W WITH GRADIENT DESCENT

LIN APPROX. $\hat{V} = x^T w = \sum_j x_j(s) w_j \Rightarrow \Delta w = \alpha(V^\pi(s) - \hat{V}(s,w))\, x(s)$ = LR * ERROR * FEATURE VAL

COARSE CODING = REPRESENT A STATE W/ OVERLAPPING CIRCULAR FEATURES $= \begin{cases} 1 \text{ FEATURE IN STATE} \\ 0 \text{ IF NOT} \end{cases}$

TILE CODING = COARSE CODING WITH STATE SPACE PARTITIONED INTO OVERLAPPING RECTANGULAR TILES

RADIAL BASIS FUNCTIONS = RBFs = COARSE CODING FOR CONTINUOUS SPACES

- EACH STATE IS A LINEAR WEIGHTED COMBO OF THE RBFs

MC FUNCTION APPROX - CONVERGES TO LOCAL OPTIMUM, LINEAR OR NONLIN APPROX

TD FUNCTION APPROX - LINEAR APPROX CONVERGES CLOSE (BIASED), BUT NOT NON-LINEAR APPROX

DEEP RL TRIAD = OFF-POLICY, FUNC. APPROX., BOOT-STRAPPING TD

DQN FEATURES

EXPERIENCE REPLAY = SAVE EXPERIENCES IN BUFFER FOR BATCHES OF RANDOM SAMPLES

- MORE EFFICIENT USE OF DATA, DATA IS REUSED
- CAN MAKE USE OF OLDER EXPERIENCES, AVOID CATASTROPHIC FORGETTING
- REDUCES CORRELATION BETWEEN SAMPLES IN TRAINING BATCH
- REQUIRES THAT IMMEDIATE OUTCOMES FOR S/A ARE SOMEWHAT STABLE LOW VARIANCE, LOW ENTROPY

PROBLEM: UPDATES ARE TOO FAST, TOO NOISY, AND CAN LEAD TO RUNAWAY BIAS

SOL'N= TARGET NETWORK = WHEN UPDATING THE MAIN NETWORK, Q, CALCULATE THE ERRORS BASED ON TARGET NETWORK, Q', Q' IS UPDATED INFREQUENTLY TO MATCH Q ⇒ STABILITY

CLIPPING REWARDS - DIFFERENT SCENARIOS HAVE DIFFERENT SCALE OF REWARDS, SO CLIP THEM ALL $\{-1, 0, 1\}$

SKIPPING FRAMES - COMPUTER CAN REACT FASTER THAN PEOPLE

- SKIPPING TO EVERY 4^{TH} FRAME ⇒ LOWER COMPUTATION COST, FASTER TRAINING
- CAN STACK THE FOUR FRAMES AS A SINGLE INPUT ⇒ ⇒ MORE INFO ON CHANGES IN CURRENT STATE

PROBLEM DQN NETWORKS OVERESTIMATE BECAUSE THEY TAKE MAX Q

$$Q(s,a) \leftarrow Q(s,a) + \alpha\left[r + \gamma \max_{a} Q'(s_{t+1}, a') - Q(s,a)\right]$$

Q' = TARGET NETWORK

DDQN - LET THE MAIN NETWORK, Q, CHOOSE THE ACTION TO QUERY, BUT GET THE VALUE FROM THE TARGET NETWORK, Q'

- DECORELLATES THE ERRORS BETWEEN CHOOSING ACTIONS & EVALUATING
- LOWERS THE BIAS, MORE STABLE

ACTOR/CRITIC METHODS

CRITIC = VALUE BASED = APPROXIMATE VALUE FUNC. ⇒ SAMPLE EFFICIENT, STEADY
EXAMPLES = Q-LEARNING, DQN, DDQN

ACTOR = POLICY BASED = FIND POLICY W/O COMPUTING Q OR V ⇒ FASTER CONVERGENCE ⇒ BETTER FOR CONTINUOUS AND STOCHASTIC ENVIRONS.
EXAMPLES = POLICY GRADIENTS, REINFORCE

POLICY GRADIENTS

= LEARN A PARAMETERIZED POLICY BASED ON POLICY WEIGHTS, θ AND OPTIMIZE

OPTIMAL POLICY $\pi^*(a|s,\theta)$ = PROBABILITY OF ACTION a IN STATE S

MAXIMIZE OUR PERFORMANCE MEASURE $J(\theta)$ W/ GRADIENT ASCENT

REINFORCE = 3-STEP POLICY GRADIENT IMPLEMENTATION

1) RUN THE POLICY, SAMPLE $\{\tau^i\}$ FROM $\pi_\theta(a_t|s_t)$ — N = NUMBER OF TRACES

2) $\nabla_\theta J(\theta) = \frac{1}{N}\sum_i^N \left(\sum_t \nabla_\theta \log \pi_\theta(a_t^i|s_t^i)\right)\underbrace{\left(\sum_t r(s_t^i, a_t^i)\right)}_{\text{RETURN}}$

3) $\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$

- REINFORCE HAS NO BIAS BUT HIGH VARIANCE

GAUSSIAN POLICY DISTRIBUTIONS FOR π = GOOD FOR ROBOTICS, REAL WORLD APPLICATIONS

PROBLEM: POLICY GRADIENT TAKES MANY ACTIONS DURING AN EPISODE
- HARD TO ASSIGN CREDIT TO SPECIFIC ACTIONS THAT LED TO REWARDS SO MAY TAKE MANY UPDATES TO CONVERGE
- POLICY GRAD ONLY VALID IN EPISODIC ENVIRONMENT

SOL'N = A2C = ADVANTAGE ACTOR CRITIC = ACTOR IS POLICY BASED, CRITIC IS VALUE BASED

1) RUN AND SAMPLE $\{s_i, a_i\}$ FROM $\pi_\theta(s,a)$

2) CRITIC FITS $\hat{V}_\theta^\pi(s)$ TO SAMPLED REWARDS

3) EVALUATE $\hat{A}^\pi(s_i,a_i) = r(s_i,a_i) + \hat{V}_\theta(s_i') - \hat{V}_\theta^\pi(s_i)$

4) $\nabla_\theta J(\theta) \approx \sum_i \nabla_\theta \log \pi_\theta(a_i|s_i)\hat{A}^\pi(s_i,a_i)$

5) UPDATE POLICY $\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$

AGENT LEARNS ADVANTAGE, $A(s,a) = Q(s,a) - V(s) = r + \gamma V(s') - V(s)$

- BOOTSTRAPPING ⇒ LOWER VARIANCE W/O ADDING BIAS
- FASTER POLICY CONVERGENCE
- CAN BE USED IN NON-EPISODIC DOMAINS

A3C = ASYNCHRONOUS ADVANTAGE ACTOR-CRITIC

= MULTIPLE INDEPENDENT AGENTS INTERACTING IN PARALLEL W/ DIFFERENT COPIES OF THE ENVIRONMENT

⇒ EXPLORE MORE OF THE STATE-SPACE QUICKER

- EACH AGENT PERIODICALLY UPDATES A GLOBAL NETWORK (NOT SYNCHRONIZED W/ OTHER AGENTS)

- AT EACH UPDATE, ALL AGENTS RESET THEIR PARAMETERS TO THE GLOBAL ONES

TRICKS OF THE TRADE

FRAME STACKING = STACK MULTIPLE FRAMES AS ONE INPUT TO SEE {POSITION, VELOCITY, ACCELERATION}

INPUT NORMALIZATION = REDUCE COMPLEXITY OF INPUT (B/W VS COLOR?)

FRAME OF REFERENCE = CHANGE COORDINATE SYSTEM TO GENERALIZE BETTER

GRADIENT CLIPPING = SCALE THE EUCLIDEAN NORM OF MODEL GRADIENT UNDER THRESHOLD

LARGE MINI-BATCH SIZE = BIAS-FREE METHOD OF REDUCING VARIANCE