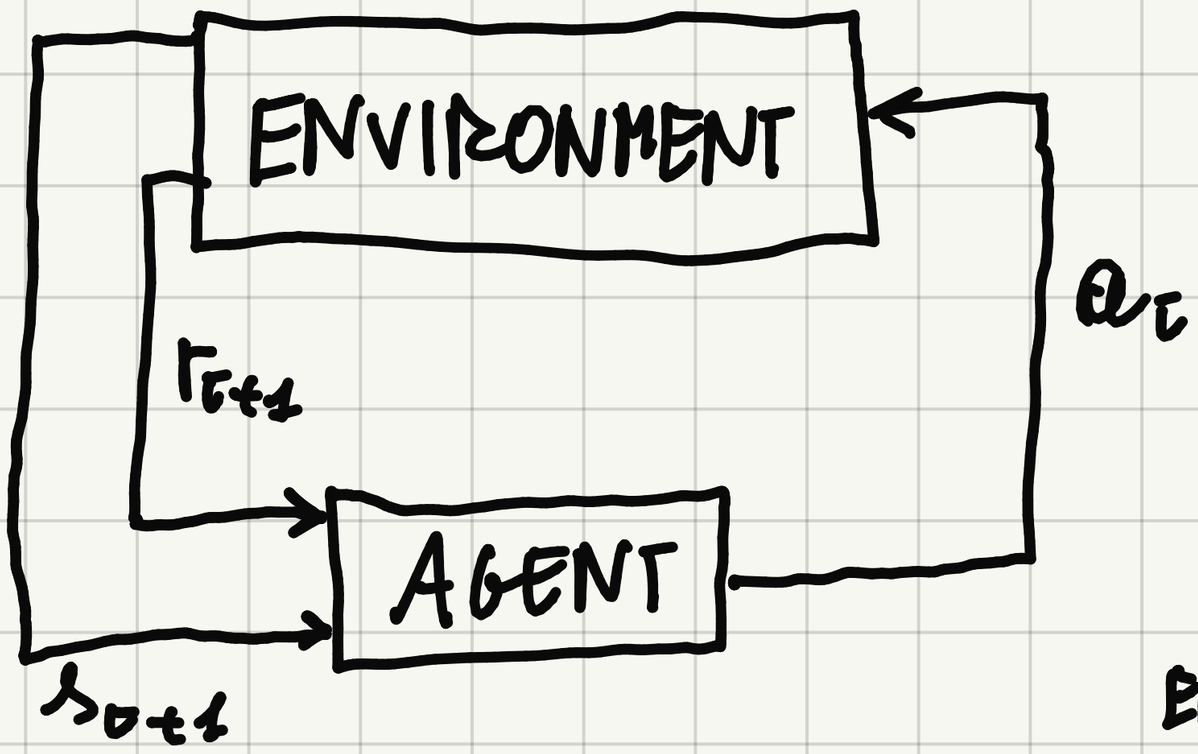
RECAP

UNSUPERVISED LEARNING: $\vec{x}$

SUPERVISED LEARNING: $\vec{x}, \vec{y}$

REINFORCEMENT LEARNING: INPUT $\vec{x}$, STATE-ACTION (s, a)

$\Rightarrow$ REWARD r (SCORE, FEEDBACK...)



CUMULATIVE REWARD = RETURN: $G_t = \sum_{k=0}^{N} \gamma^k r_{t+k+1}$

CHANGES IN A NEW STATE

γ: DISCOUNT FACTOR

GOAL: MAXIMIZE G

POLICY: $\pi(a|s)$

VALUE FUNCTION: $V^{\pi}(s) = E(G_t | S_t = s)$

$\pi(a|s) \mid \max E_{\pi}[\sum_{t=0}^{\infty} \gamma[t]], \gamma \in [0,1]$

"AS LONG AS WE GET TO THE RESULT, WE'RE GOOD"

ACTION-VALUE FUNCTION: $Q^{\pi}(s) = E[G_t | S_t = s, A_t = a]$

$a' = s_{t+1}$

BELLMAN EQUATIONS: $V^{\pi}(s) = \sum_a \pi(s,a) \sum_{s'} P^a_{ss'} [R^a_{ss'} + \gamma V^{\pi}(s')]$

$Q^{\pi}(s,a) = \sum_a \pi(s,a) \sum_{s'} P^a_{ss'} [R^a_{ss'} + \gamma Q^{\pi}(s')]$

Reinforcement Learning Worksheet (Part 1)

Answer each of the following questions. If you can't answer a question, then raise your hand and a TA will assist you. At the end of the session, the TA will solve the problems on the board. If you finish quickly, feel free to look over your answers with the TA and leave early.

We recommend that you consult the second edition of *Reinforcement Learning: An Introduction* by Richard S. Sutton and Andrew G. Barto for help solving these questions. The book is freely available in full at http://incompleteideas.net/book/RLbook2018.pdf

1. Supervised learning and reinforcement learning exhibit a number of critical differences. *(5 points total)*
   - 1.1. Describe three of these differences. *(3 points)*
   - 1.2. Describe one way in which the problem of learning an optimal policy in a Markov decision process with fixed length episodes can be modeled as a supervised learning problem. *(2 points)*

2. The Markov property lets us assume that knowing the current state of the MDP is as useful as knowing the full history for choosing the next action. *(6 points total)*
   - 2.1. Write a mathematical equation that describes this property. *(1 point)*
   - 2.2. In which of the following scenarios would the Markov property hold? Why?
     - 2.2.1. Playing a game of Go where the state is the colours of all the pieces on the board and their respective locations. Assume that black always moves first. *(1 point)*
     - 2.2.2. Navigating a car from your place of residence to the university where the state is the current position, velocity, and acceleration of the car. *(1 point)*
     - 2.2.3. Buying and selling stocks where the state is the current value of all the stocks, the number of each stock you own, and the amount of each stock available for sale with their respective prices. *(1 point)*
   - 2.3. In non-Markovian problems, we can frequently use a different definition for the state to make the problem Markovian. Do this for each of the scenarios in 1.2. that you said were non-Markovian (there should be at least one). *(2 points)*

3. You're a contestant on a game show and you're asked to pick one of three doors. Behind one of the doors is a car, behind the other two are goats. You don't like goats. After you pick a door, the presenter, Mr. Hall, will select one of the two doors that you didn't pick that has a goat behind it, open it, and show it to you. He'll then ask if you want to change your guess. Assume that each door is equally likely to have had the car behind it when the game starts. *(10 points total)*

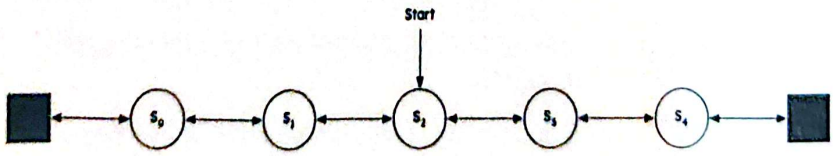
   3.1. What is the probability that if you change your guess to the door you didn't pick and which Mr. Hall didn't open then you will find a car behind it? *(1 point)*

   3.2. Model this as a reinforcement learning problem and give the state set, the action set(s), and the reward function. Remember that the action set can be the same or different for each state. *(5 points)*

   3.3. Using your formulation of the problem from 3.1., pick a policy and give the state-action values for each state-action pair. *(2 points)*

   3.4. Give a set of mathematical equations defining a policy that would be greedy with respect to the optimal value function. Is it an optimal policy? Why? *(2 points)*

4. Consider the slippy random walk shown below. At each step, an agent is in one of the seven states and—while in a non-terminal state—can choose to either move one state to the left or one state to the right. However, there is some probability α that the action taken by the agent ends up having the opposite effect. If the agent reaches the left-hand terminal state, then it gets a reward of -1, and if it reaches the right-hand, it gets a reward of 1. Ending a step in any other state means the agent gets a reward of 0 and the discount factor at each timestep is 0.9. *(7 points total)*



   4.1. Assume that the initial estimate of all the state values is 0 and the agent's policy is to always try and go right. What are the state value estimates after one step of policy evaluation? After two? After three? *(4 points)*

   4.2. Give a set of linear equations that, when solved, give you the value of each state when the agent's policy is to always try and go right. *(3 points)*

1

1.1

a) SUPERVISED LEARNING → DATA CONTAINS THE DESIRED SOLUTION

REINFORCEMENT LEARNING → AGENT AUTONOMOUSLY LEARN HOW TO GET THE DESIRED RESULT/REWARD

b) SUPERVISED LEARNING → AIMS TO REDUCE THE ERROR, DEFINED AS THE DIFFENCE BETWEEN THE MODEL PREDICTION AND THE DATASET RESULT

REINFORCEMENT LEARNING → FIND A CORRECT POLICY TO MAXIMIZE THE REWARD

c) SUPERVISED LEARNING → FIXED DATASET

REINFORCEMENT LEARNING → DYNAMIC LEARNING AND "REAL-TIME" UPDATES BASED ON FEEDBACKS

1.2

MDP CAN BE DESCRIBED AS:

- FINITE SET OF STATES S AND ACTIONS A FROM STATES AND ACTION a
- TRANSITION FUNCTION $P(s'|s,a)$ = PROBABILITY TO GET TO s' STARTING
- REWARD FUNCTION $R(s,a)$
- FINITE TIMESET T

AN EXAMPLE: MONTECARLO SIMULATION

SIMULATE EPISODES AND GET G_t FOR EACH (s,a). THE OPTIMAL SOLUTION WILL BE THE ONE THAT MAXIMIZES G_t

$$G_t = \sum_{k=0}^{T-t} \gamma^k r_{t+k}, \quad 0 \leq \gamma \leq 1$$

$$\text{ESTIMATED VALUE } Q(s,a) = \frac{\sum G_t}{\#\text{VISITS TO } (s,a)}$$

AVERAGE OF THE RETURNS OBSERVED FOR THE SAME TUPLE (s,a) IN ALL THE EPISODES

$$E = \frac{1}{N}\sum_{i=1}^{N}\left(Q_{PRED}(s_i,a_i) - Q_{TARGET}(s_i,a_i)\right)^2 \qquad \pi^*(s) = \underset{a}{\operatorname{argmax}}\, Q(s,a)$$

(2)

2.1

$$P(S_{t+1}=s' \mid S_t) = P(S_{t+1}=s' \mid S_t, S_{t-1}, \ldots, S_1)$$

2.2

a) YES — THE OPTIMAL ACTION DEPENDS ONLY ON THE CURRENT STEP

b) NO — EXTERNAL CONDITIONS (TRAFFIC, WEITHER...) COULD AFFECT THE SYSTEM

c) NO — "THE TREND" DEPENDS NOT ONLY FROM 1 STATE BUT FROM ALL

2.3

b) IT SUFFICES TO ADD "CAR CONDITION" IN s

c) CONSIDER "ALL THE PAST" AS A UNIQUE STATE

! IN ALMOST ALL THE REAL-WORLD APPLICATIONS, MDP IS NOT USABLE

BUT, BY MAKING SOME ASSUMPTIONS, WE CAN APPROXIMATE THE PROBLEM

③

3.1

AT THE BEGINNING, $P(G) = 2/3$ AND $P(C) = 1/3$

$$P(C|G) = \frac{P(C)P(G)}{P(C)} = 2/3$$

"IF WE CHANGE, WE ASSUME THAT C IS IN THE OTHER DOOR"

MONTY-HALL PARADOX

3.2

$S = \{$NO DOOR PICKED, DOOR 1 PICKED DOOR 2 OPENED, DOOR 1 PICKED DOOR 3 OPENED,

DOOR 2 PICKED DOOR 1 OPENED, DOOR 2 PICKED DOOR 3 OPENED, DOOR 3 PICKED DOOR 1 OPENED,

DOOR 3 PICKED DOOR 2 OPENED, GOT CAR, GOT GOAT$\}$

$A = \{$DOOR 1, DOOR 2, DOOR 3$\} + \{$STAY, CHANGE$\}$

$$R = \begin{cases} 1 & S = \text{GOT CAR} \\ 0 & S = \text{GOT GOAT} \end{cases}$$

3.3

POLICY = DOOR 1 - CHANGE

Q_π(NO DOOR PICKED, *) $= \gamma\, 2/3$

Q_π(DOOR # PICKED DOOR # OPENED, STAY) $= 1/3$

Q_π(DOOR # PICKED DOOR # OPENED, CHANGE) $= 2/3$

3.4

$\pi(\text{NO DOOR PICKED}, \text{DOOR 1}) = 1$

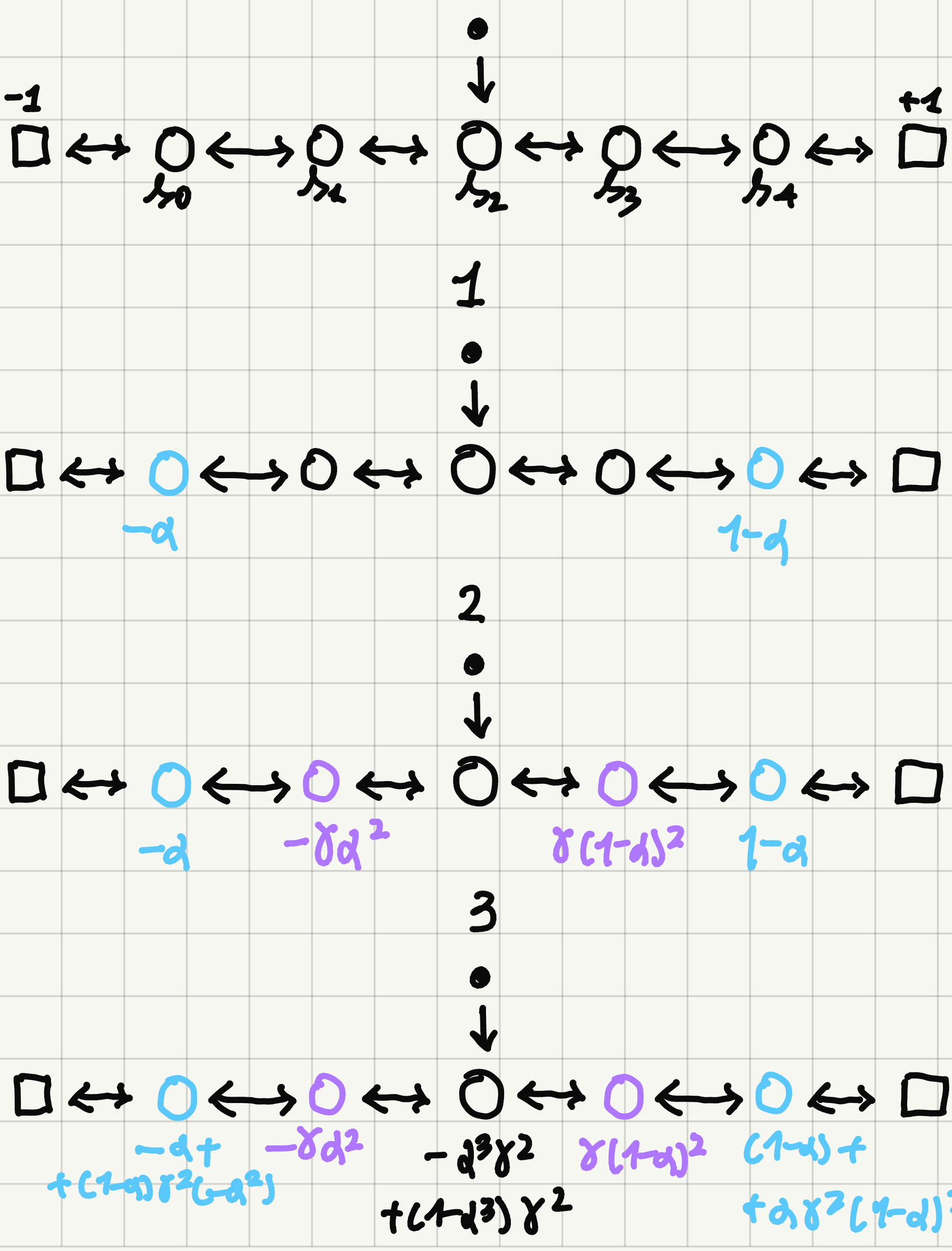
$\pi(\text{DOOR} * \text{PICKED DOOR} * \text{OPENED}, \text{CHANGE}) = 1$

IT IS AN OPTIONAL POLICY BY DEFINITION

4

4.1

POLICY EVALUATION $V_k(s) = \sum_a \pi(s,a) \sum_{s'} P^a_{ss'} (R_{ss'} + \gamma V_{k-1}(s'))$



4.2

$$V(s_0) = -\alpha + (1-\alpha)\gamma V(s_1)$$

$$V(s_1) = \alpha\gamma V(s_0) + (1-\alpha)\gamma V(s_2)$$

$$V(s_2) = \alpha\gamma V(s_1) + (1-\alpha)\gamma V(s_3)$$

$$V(s_3) = \alpha\gamma V(s_4) + (1-\alpha)V(s_4)$$

$$V(s_4) = \alpha\gamma V(s_3) + (1-\alpha)$$

Reinforcement Learning Worksheet (Part 2)

Answer each of the following questions. If you can't answer a question, then raise your hand and a TA will assist you. At the end of the session, the TA will solve the problems on the board. If you finish quickly, feel free to look over your answers with the TA and leave early.

We recommend that you consult the second edition of *Reinforcement Learning: An Introduction* by Richard S. Sutton and Andrew G. Barto for help solving these questions. The book is freely available in full at http://incompleteideas.net/book/RLbook2018.pdf

5. Draw backup diagrams for each of the following algorithms: *(6 points total)*
   - 5.1. Monte Carlo *(1 point)*
   - 5.2. TD(0) *(1 point)*
   - 5.3. TD with two-step rollout *(1 point)*
   - 5.4. TD(1) *(1 point)*
   - 5.5. SARSA *(1 point)*
   - 5.6. Q-learning *(1 point)*

6. Give an example of an environment where you would want to use *(4 points total)*
   - 6.1. dynamic programming over TD, *(1 point)*
   - 6.2. a large ϵ when performing ϵ-greedy exploration with Q-learning, *(1 point)*
   - 6.3. a small ϵ when performing ϵ-greedy exploration with Q-learning, and *(1 point)*
   - 6.4. something other than ϵ-greedy exploration. *(1 point)*

7. What is a good value for ϵ in Andrej Karpathy's Temporal Difference Learning Gridworld Demo? You can find it here: https://cs.stanford.edu/people/karpathy/reinforcejs/gridworld_td.html *(1 point)*

8. Which of the following algorithms are off-policy algorithms? *(6 points total)*

   8.1. Monte Carlo *(1 point)*

   8.2. TD(0) *(1 point)*

   8.3. Expected SARSA *(1 point)*

   8.4. Q-learning *(1 point)*

   8.5. REINFORCE *(1 point)*

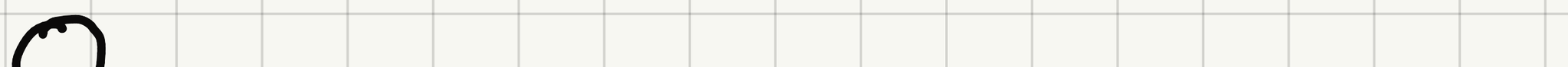
   8.6. DQN *(1 point)*

9. DQN was the first widely-successful architecture that used deep artificial neural networks to solve reinforcement learning problems. This success was due to it encompassing several key mechanisms that go above and beyond the basic Q-learning architecture. Draw a diagram showing these mechanisms in relation to the basic Q-learning architecture and describe their purpose. *(5 points)*

10. Exploration remains one of the great unsolved problems in reinforcement learning. However, there are obvious techniques we could use that are more principled than ϵ-greedy exploration. Try to think of two of these and describe why you think they would work better than ϵ-greedy exploration? *(4 points)*

5

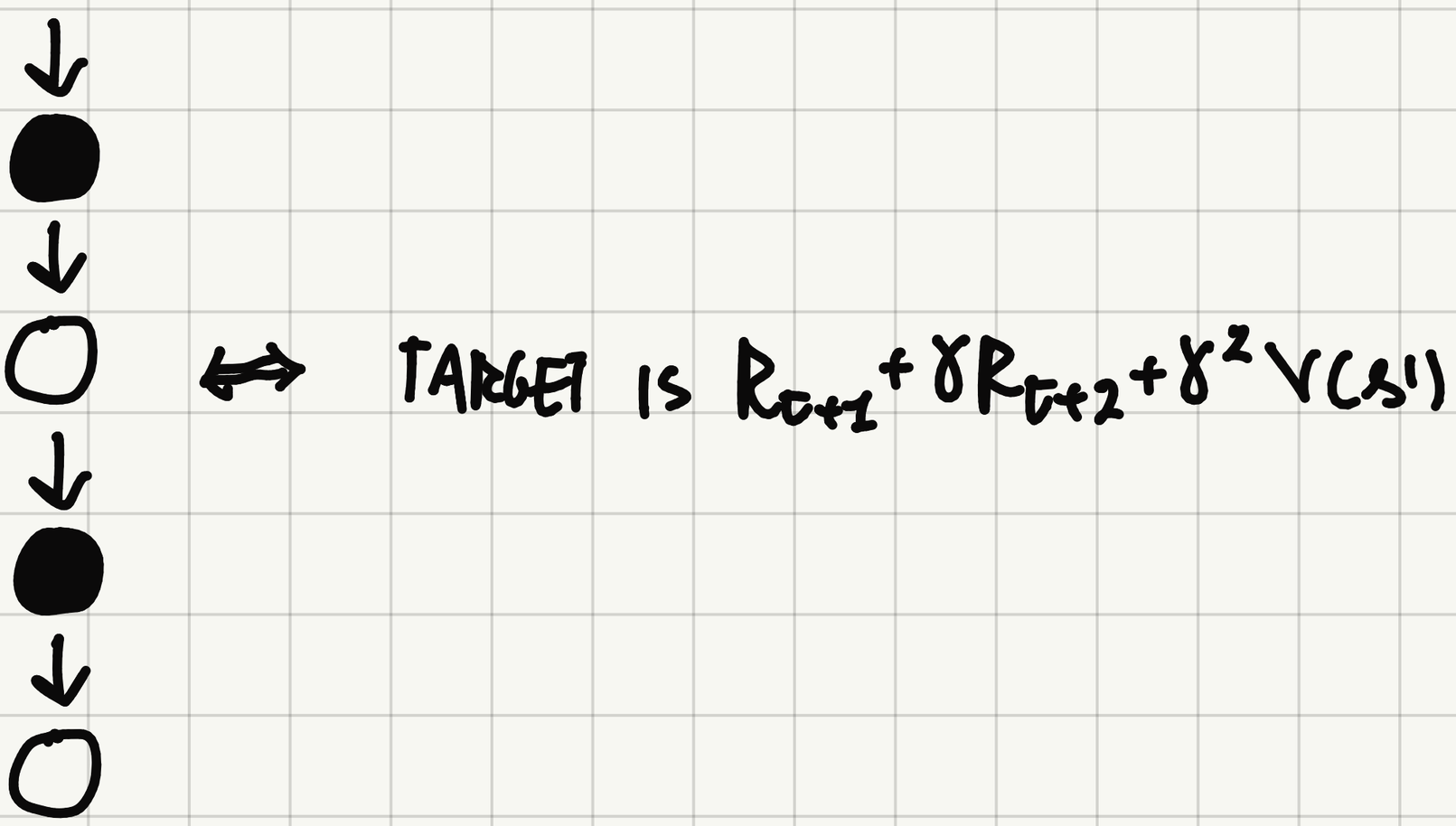
5.1


TARGET IS G_t

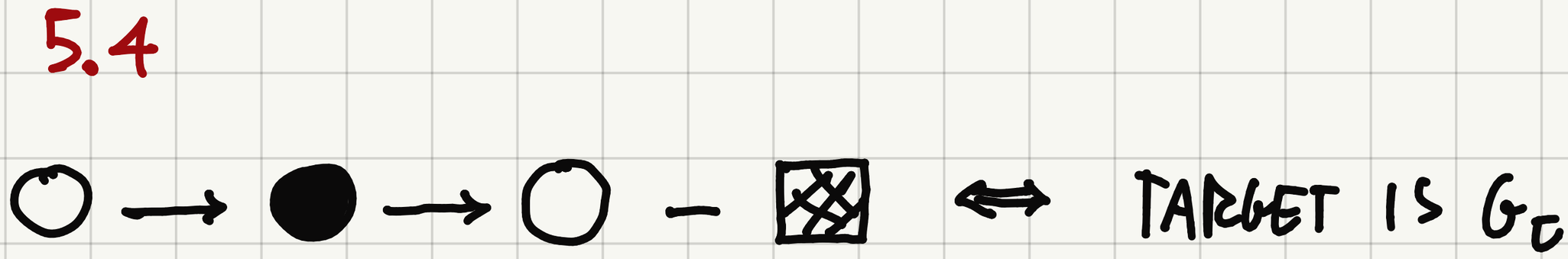
5.2

TARGET IS $R_{t+1} + \gamma V(S')$

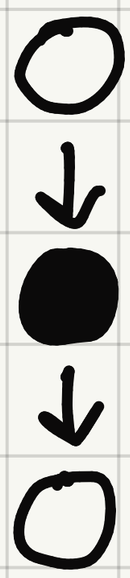
5.3


TARGET IS $R_{t+1} + \gamma R_{t+2} + \gamma^2 V(S')$

5.4

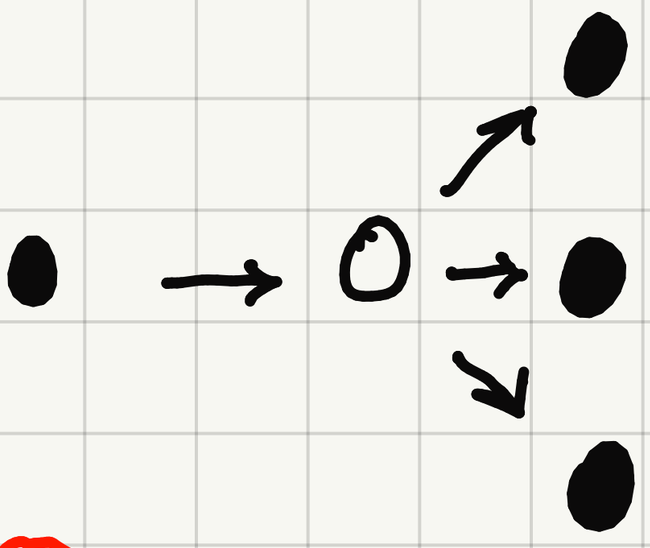

TARGET IS G_t

5.5



$\Leftrightarrow$ TARGET IS $R_{t+1} + \gamma Q(s,a)$ $a \approx \pi$

5.6 MOST COMMON ARCHITECTURES



$\Leftrightarrow$ TARGET IS $R_{t+1} + \max_Q \gamma Q(s,a)$

(6)

6.1

DYNAMIC PROGRAMMING: WE HAVE A MODEL OF THE ENVIRONMENT

TEMPORAL DIFFENCE : WE DONT HAVE A MODEL OF THE ENVIRONMENT

EXAMPLE) SHOP MANAGEMENT, WHERE WE MIGHT HAVE BOTH TRENDS AND HISTORICAL VALUES TO ANALYZE

6.2

HIGH ε => EXPLORATION-LIKE MODEL. EASY TO REVEAL A LOCAL MINIMUM

EXAMPLE) A DYNAMIC AND NOT-KNOWN ENVIRONMENT, LIKE A DRONE TESTED IN A NEW SPACE

6.3

LOW ε => EXPLOIT THE "BEST ACTION" ALREADY TAKEN.

WHEN IT IS HARD TO REVEAL A LOCAL MINIMUM

EXAMPLE) TRAFFIC MANAGEMENT

6.4

VCB: EXPLORE ACTIONS WITH HIGH UNCERTAINTY. WHEN DEALING WITH

ENVIRONMENTS WITH A BOTTLENECK STATE

EXAMPLE) ONLINE SPOTS

⑦

START WITH HIGH VALUES (0.2-0.5), REDUCE ($\simeq$ 0.1) AND TERMINATE WITH LOW VALUES ($\simeq$ 0.01-0.05). A GOOD VALUE MIGHT BE 0.25

⑧

TARGET

8.1) NO LEARNS BY USING THE SAME POLICY USED TO EXPLORE => BEHAVIOUR =

8.2) NO DATA ARE UPDATED BASED ON THE CURRENT POLICY USED TO CHOOSE ACTIONS

8.3) NO EXPECTED VALUES ARE CALCULATED W.R.T. EXPLOIT FUNCTION

8.4) YES BEHAVIOUR POLICY = TARGET POLICY

8.5) NO EXPLOIT POLICY ITSELF IS OPTIMIZED

8.6) YES LEARN FROM THE PAST INDEPENDENTLY FROM THE EXPLOIT POLICY

9

TD

BOOTSTRAPPED PREDICTION → Q-LEARNING

OFF-POLICY TD(0)-BASED CONTROL → TARGET NETWORK

STABILIZED TARGET FOR LEARNING

ANN

LEARN KEY ABSTRACT FEATURES OF OBSERVATION

ENV → RL PROBLEM ITSELF

E-GREEDY → NAIVE BUT SUFFICIENT EXPLORATION

EXPERIENCE REPLAY → REDUCED CATASTROPHIC FORGETTING AND APPROXIMATES IID SAMPLES

10

- VCB
  - PRIORITIZES LESS EXPLORED ACTIONS
  - AVOIDS USELESS/INEFFICIENT RANDOM EXPLORATIONS

- INTRINSIC MOTIVATION
  - MORE FOCUS TO USEFUL STATES
  - IMPROVED EFFICIENCY IN COMPLEX ENVIRONMENTS

  $R_t \leftarrow R_t + \frac{\gamma}{\varphi(s')}$ → NUMBER OF TIMES s' HAS BEEN SEEN

- ANNEALING EPSILON
  - REDUCE ε OVER TIME $\varepsilon = 0.99^t$

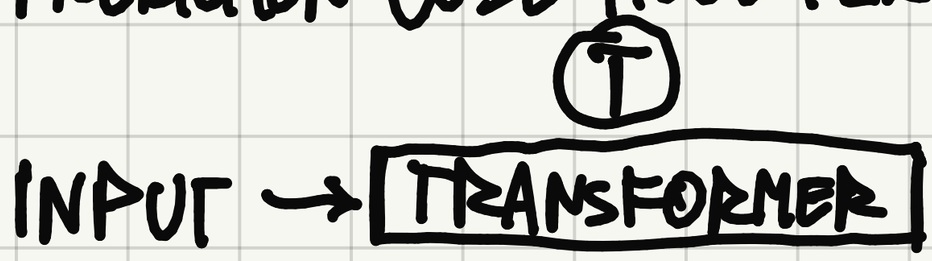
- ε-SOFTMAX-GREEDY
  - WITH $P=0.9$, SAME GREEDY ACTION

    × $P=0.09$ => ACTION ACCORDING TO SOFTMAX

    × $P=0.01$ => RANDOM ACTION

EXTRA: CREATE YOUR OWN CHATGPT

1) TRAIN A 100B PARAMETER TRANSFORMER TO DO NEXT-WORD PREDICTION (USE HUGE TEXT CORPUS)



(GPT-3)

2) MAKE T GENERATE 2 MOST LIKELY OUTPUTS FOR EACH OF THE INPUTS

3) HAVE A HUMAN SCORE EACH PAIR $X \to (A,B)$ "A:7; B=4..."

4) COPY T AND MAKE IT TAKE ITS OUTPUTS AS ADDITIONAL INPUTS AND MAKE IT OUTPUTS SCALAR SCORES

5) TRAIN T_{SCORE} TO PREDICT HUMAN SCORES USING THE 100000 INPUTS

6) GENERATE 100000000 NEW OUTPUTS AND SCORE THEM WITH T_{SCORE}

7) USE THEM TO TRAIN T TO PREDICT HIGH-SCORING EXAMPLES R.L.

8) GIVE IT A WEBSITE CHAT-GPT