CSYE 7374

Practice Exam with Solutions

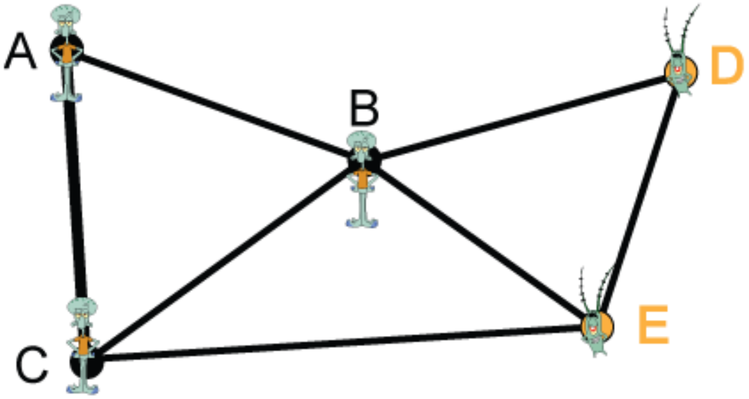
Student Name: ____

Professor: Nik Bear Brown

Rules:

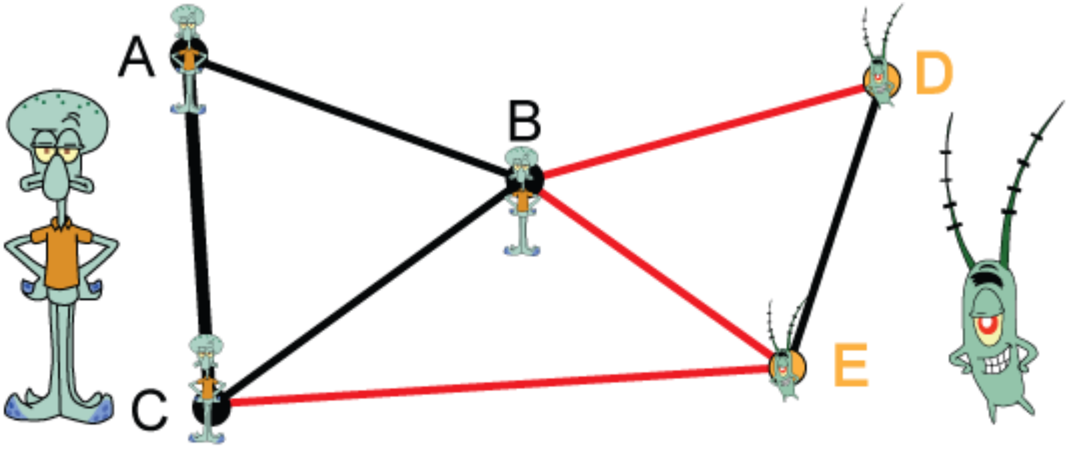
1. HackerRank Exam
2. It is a closed-book quiz.
3. Do not open another tab or a window. You'll lose points for that.
4. You'll be provided with the quiz link and password 5 mins before the start time.

Q1 (5 Points) The graph below is composed of two independent sets (A,B,C) (i.e. those vertices with Squidward) and (D,E) (i.e. those with Sheldon James Plankton). What is the cut set of the graph?



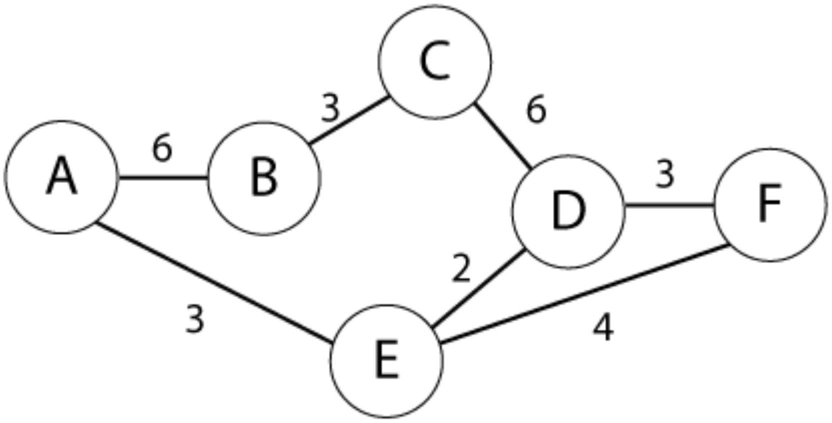
Solution:

The cut set is (B-D, B-E, C-E)



A cut set is a set of edges whose end points are in different subset of the partition. Edges are said to be crossing the cut if they are in its cut-set. A set of edges of a graph, which, if removed (or "cut"), disconnects the graph (i.e., forms a disconnected graph).

Q2 (5 Points) Find shortest path from A to F in the graph below using Dijkstra's algorithm. *Show your steps.*



Solution:

```
Given a graph, G, with edges E of the form (v1, v2) and vertices V, and a
source vertex, s

dist : array of distances from the source to each vertex
prev : array of pointers to preceding vertices
i    : loop index
F    : list of finished vertices
U    : list or heap unfinished vertices

/* Initialization: set every distance to INFINITY until we discover a path */
for i = 0 to |V| - 1
   dist[i] = INFINITY
   prev[i] = NULL
end
```

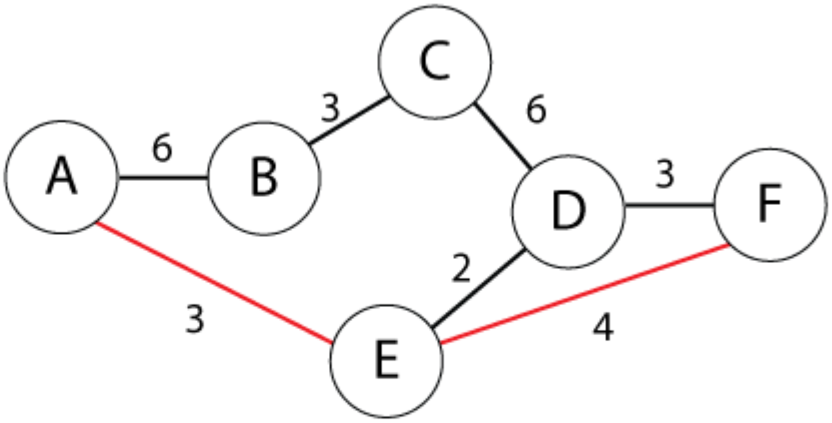
```
while(F is missing a vertex)
  pick the vertex, v, in U with the shortest path to s
  add v to F
  for each edge of v, (v1, v2)
      /* The next step is sometimes given the confusing name "relaxation"
      if(dist[v1] + length(v1, v2) < dist[v2])
         dist[v2] = dist[v1] + length(v1, v2)
         prev[v2] = v1
         possibly update U, depending on implementation
      end if
   end for
end while
```

Source A

		A	B	C	D	E	F
1: A {A}	A	0	(6,A)	INF	INF	(3, A)**	INF
2: E {A,E}	E	0	(6,A)	(9,B)	(5,E) **	(3, A)	(7,E)
3: D {A,E,D}	D	0	(6,A) **	(9,B)	(5,E)	(3, A)	(7,E)
4: B {A,E,D B}	B	0	(6,A)	(9,B)	(5,E)	(3, A)	(7,E) **
5: F { A,E,D B,F}	F	0	(6,A)	(9,B) **	(5,E)	(3, A)	(7,E)
4: C { A,E,D B,F,C}	C	0	(6,A)	(9,B)	(5,E)	(3, A)	(7,E)

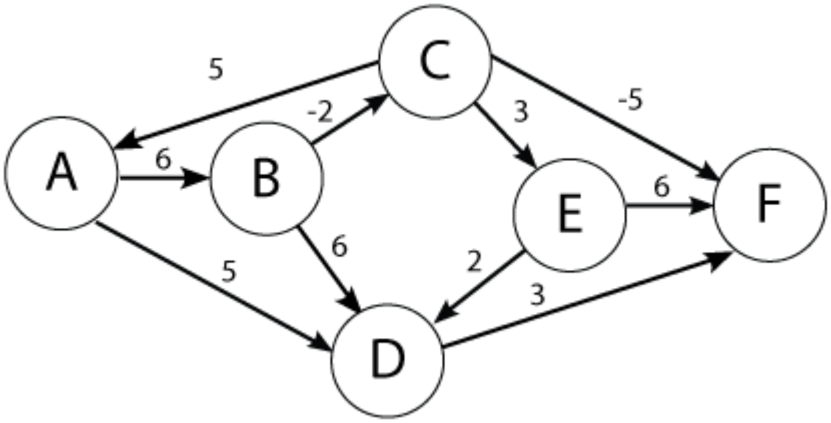
** U with the shortest path to s

Red indicates a vertex has been moved from U to F



Now we return our final shortest path, which is: A → E → F Cost (3+4 = 7)

Q3 (5 Points)



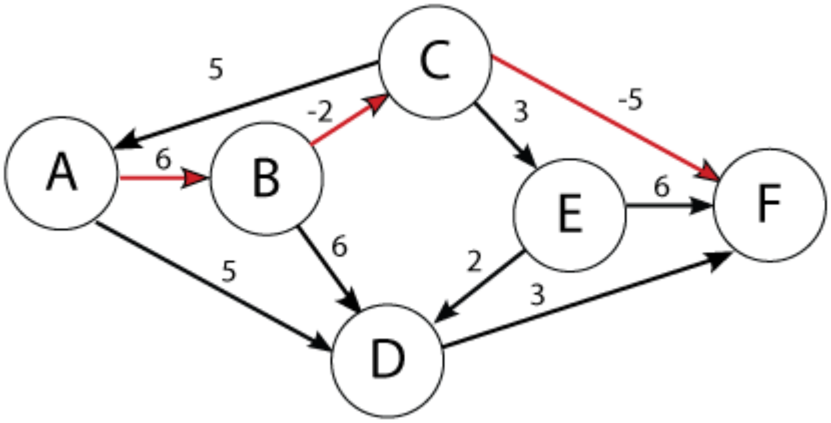
Use the Bellman-Ford algorithm to find the shortest path from node A to F in the weighted directed graph above. *Show your work.*

Solution:

Shortest path: A->B->C-->F at cost -1

	A	B	C	D	E	F
0	0	INF	INF	INF	INF	INF
1	0	6	INF	5	INF	INF
2	0	6	4	5	INF	8
3	0	6	4	5	7	-1
4	0	6	4	5	7	-1
5	0	6	4	5	7	-1
6	0	6	4	5	7	-1

Many students calculate from F to A which is fine but the numbers in the table will be different even though the final path will be the same.



Another Bellman-Ford algorithm example is worked detail at https://www.youtube.com/watch?v=hq3TZInZ5J4

Q4 (5 Points) Write the *Bellman optimality equation* used in reinforcement learning and explain its meaning.

Solution:

The equation for the optimal policy is referred to as the *Bellman optimality equation*:

$$V\pi*(s)=\mathrm{maxa}\{R(s,a)+\gamma\sum s'P(s'|s,a)V\pi*(s')\}$$

where $\pi*$is the optimal policy and $V\pi*$ refers to the value function of the optimal policy. The equation above describes the reward for taking the action giving the highest expected return.

Q5 (5 Points) Why were we using Q-values? What is the advantage of learning state-action values (Q) compared to state values (V)? ?(Hint: consider action selection)

Solution:

The advantage is in the action selection. With Q-values, we can directly see the value of each available action, and use these in e.g. -greedy or softmax action selection. On the contrary, when we only have V-estimates, we need to 1) do extra calculations, and 2) have a transition model, so at each decision moment we can calculate the Q(s, a). for all available actions a. Especially when controlling a real-time system, your decisions have to be fast.

Q6 (5 Points) What is q-learning? Define below term. Exploration rate, learning rate and Discounting rate?

Solution:

Q-learning is an off-policy reinforcement learning algorithm that seeks to find the best action to take given the current state. It's considered off-policy because the q-learning function learns from actions that are outside the current policy, like taking random actions, and therefore a policy isn't needed. More specifically, q-learning seeks to learn a policy that maximizes the total reward.

Exploration rate: This exploration rate is the probability that our agent will explore the environment rather than exploit it

Learning Rate: learning rate, often referred to as alpha or α, can simply be defined as how much you accept the new value vs the old value.

Discount factor: It's used to balance immediate and future reward

Q7 (5 Points) What is an activation function? Give an example of activation function

Solution:
An activation function is something which helps us regulate the flow of information from the previous set of neurons to the next. It is a mathematical function which decides whether a particular neuron should be activated or not. Depending upon the type of activation values are shrunk in various ranges from -1 to +1 in case of tanh or 0 - 1 in case of sigmoid.

Example of activation function: Sigmoid $f(x) = 1 / (1 + e^{-x})$

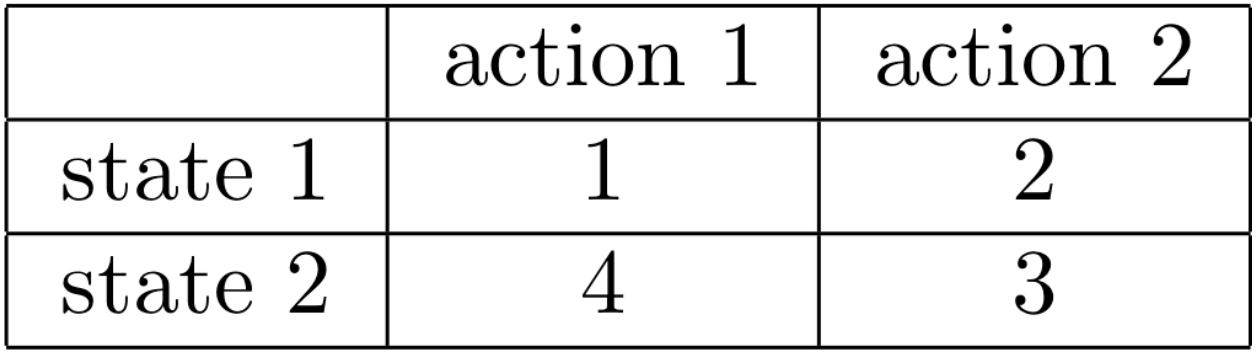
Q8 (5 Points) What is a policy in reinforcement learning? What is the difference between a deterministic policy and a stochastic policy?

Solution:

A policy, denoted as π (or sometimes π(a|s)), is a mapping from some state s to the probabilities of selecting each possible action given that state. For example, a greedy policy outputs for every state the action with the highest expected Q-Value.

- A **deterministic policy** is a mapping $\pi : \mathcal{S} \rightarrow \mathcal{A}$. For each state $s \in \mathcal{S}$, it yields the action $a \in \mathcal{A}$ that the agent will choose while in state s.
- A **stochastic policy** is a mapping $\pi : \mathcal{S} \times \mathcal{A} \rightarrow [0, 1]$. For each state $s \in \mathcal{S}$ and action $a \in \mathcal{A}$, it yields the probability $\pi(a|s)$ that the agent chooses action a while in state s.

Q9 (5 Points) Which of the following describes the policy that is greedy with respect to the Q-table? What action would a greedy policy take in states 1 and 2?



	action 1	action 2
state 1	1	2
state 2	4	3

Solution:

For state 1, action 2 has the highest estimated return (2>1). For state 2, action 1 has the highest estimated return (4>3).

Q10 (5 Points) What is the difference between a value function V and a Q function in reinforcement learning?

Solution:

Anything that says that one is a state-value function and the other is an action-value function is accepted for full credit.

$V\pi(s)$ is the state-value function. The V value function states what the expected overall value of a state s under the policy π.

$Q\pi(s,a)$ is the action-value function. The Q function states what the value of a state s and an action $a$under the policy π.

The relationship between $Q\pi$and $V\pi$ (the value of being in that state) is

$$V\pi(s)=\sum a\in A \quad \pi(a|s)*Q\pi(a,s)$$

Q11 (5 Points) Which of the values for epsilon yields an epsilon(ϵ)-greedy policy that is guaranteed to always select the greedy action? What is the range of values that epsilon(ϵ) can take? Which of the values for epsilon yields an epsilon(ϵ)-greedy policy that is guaranteed to always select the non-greedy action?

Solution:

You can think of the agent who follows an ϵ-greedy policy as always having a (potentially unfair) coin at its disposal, with probability ϵ of landing heads.

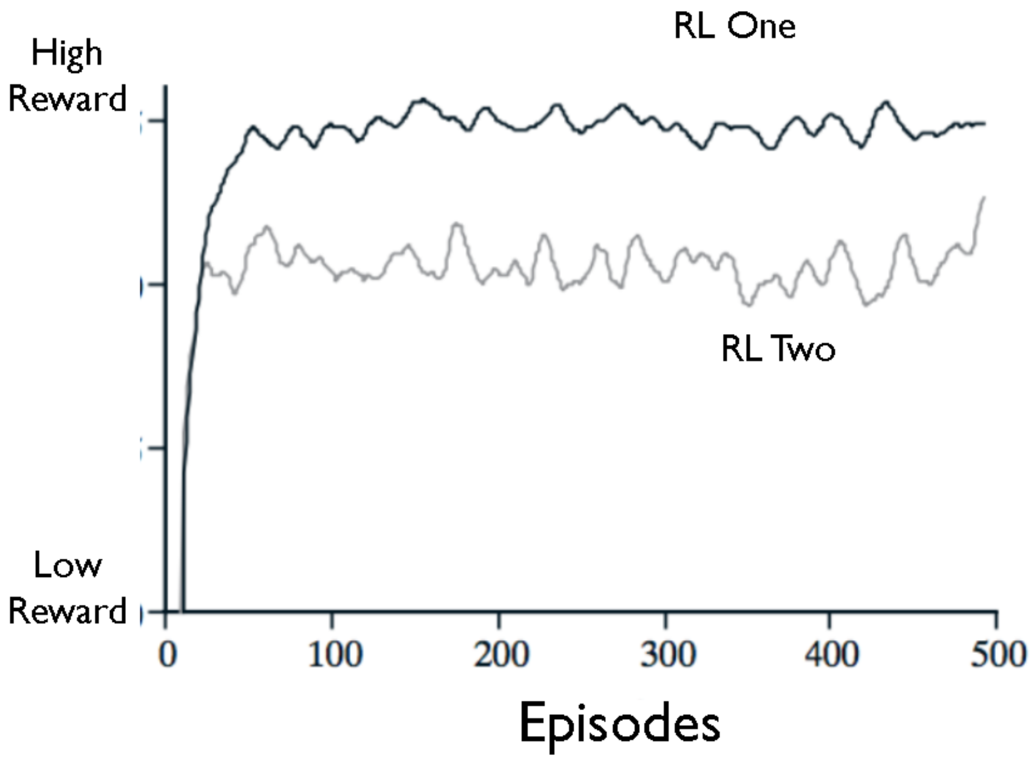
If the coin is greater or equal to epsilon(ϵ) it will take the max (greedy) q-value so if the ϵ is 0 the agent will always take the max (greedy) q-value.

There is NO VALUE of epsilon yields an epsilon(ϵ)-greedy policy that is guaranteed to **always** select the non-greedy action as there is always the possibility that the coin returns a 1.

Note that ϵ must always be a value between 0 and 1, inclusive (that is, $\epsilon\in[0,1]$).

Q12 (5 Points) Which reinforcement learning algorithm is doing better? RL One (top) or RL Two (bottom)? Why? Do the algorithms plateau? If so, where?

Solution:



Q13 (5 Points)) In OpenAi- Gyms Taxi-v2 environment there are 4 locations (labeled by different letters) and your job is to pick up the passenger at one location and drop him off in another. You receive +20 points for a successful drop-off and lose 1 point for every timestep it takes. There is also a 10-point penalty for illegal pick-up and drop-off actions. Assume the environment is on a 5x5 grid and the taxi can move up, down, right, left, wait, pick up passenger or drop passenger. How many states are there? How many actions are there? What is the size of the Q-table?

Solution:

How many states are there?

There are 5x5 or 25 states.

How many actions are there?

There are 7 actions, move up, down, right, left, wait, pick up passenger or drop passenger.

The Q-table is 25x7 or 175.

Q14 (5 Points) Which of the following state spaces are continuous and which are discrete?

A. Playing cards with a standard 52-card deck.
B. GPS coordinates for self-driving cars.
C. Force applied to the arm of an industrial robot.
D. Board positions in a standard tic-tack-toe game.
E. Keys to play on a keyboard.

Solution:

Discrete (A, D, E)

Playing cards with a standard 52-card deck.
Board positions in a standard tic-tack-toe game.
Keys to play on a keyboard.

Continuous (B, C)

GPS coordinates for self-driving cars.
Force applied to the arm of an industrial robot.

Q15 (5 Points) Describe the structure of a Convolutional neural network (CNN). Is it supervised or unsupervised? How is subsampling done in CNNs?

Solution:

A Convolutional neural network (CNN) is supervised (the output is labeled). Subsampling is usually done by pooling.

Input->Conv->RELU (or another non-linear activation function) -> Pool (usually Max-pool)
the above sequence can be repeated for a while until it goes to a MLP structure

fc->RELU (or another non-linear activation function) for each hidden layer then possibly a normalizing activation like softmax then output layer.

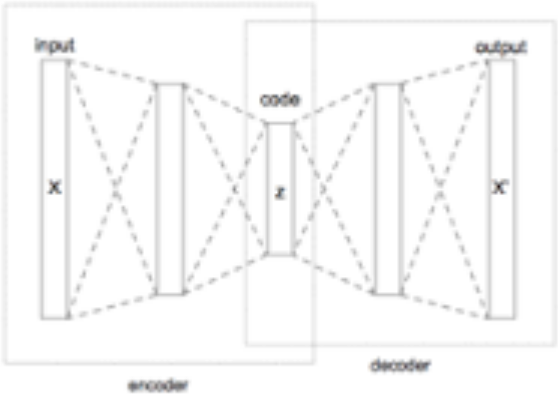
Note some consider the activation function to be part of fc, so something like fc-> fc-> fc->Softmax (maybe)->output is fine for full credit.

Q16 (5 Points) Describe the structure of an Autoencoder (AE). Is it supervised or unsupervised?

Solution:

An Autoencoder (AE) is unsupervised.

An **autoencoder** consists of two parts, the encoder and the decoder, which can be defined as transitions and such that: In the simplest case, given one hidden layer, the encoder stage of an **autoencoder** takes the input and maps it to : This image is usually referred to as code, latent variables, or latent representation. The latent layer is the basis of the latent representation.

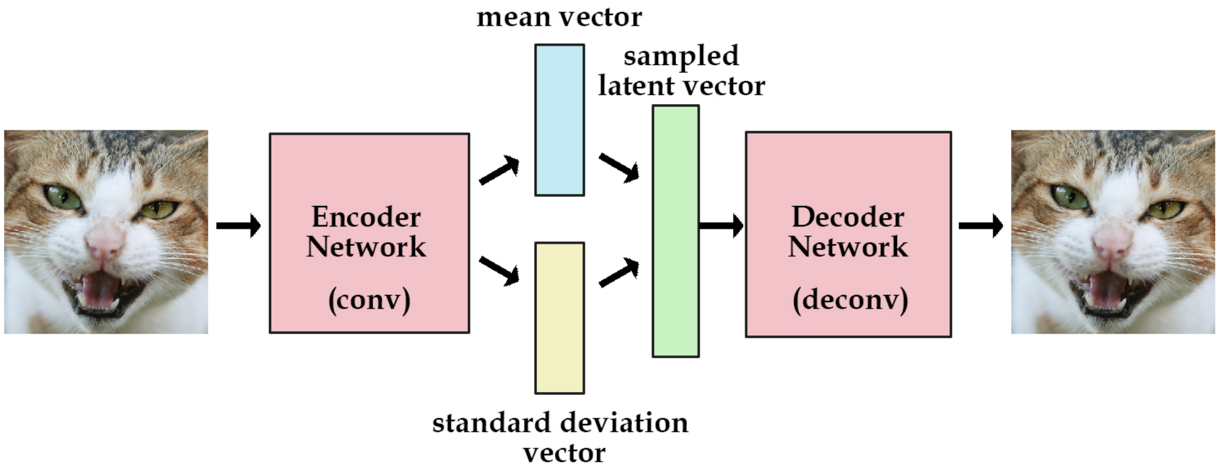


Q17 (5 Points) Describe the structure of a Variational Autoencoder (VAE). Is it supervised or unsupervised? How are the parameters of any probabilistic distributions represented in its structure?

Solution:

A Variational Autoencoder (VAE) is unsupervised the input layer is the same data as the output layer.

From the input layer there is an encoder network of one or more layers and before the output layer there is a corresponding decoder network. The latent layer is the basis of the latent representation. Unlike a deterministic autoencoder. additional layers which allow for the estimation of probability distributions are added. For example, if a Gaussian distribution is used layers that represent the parameters of a Gaussian distribution, the mean and standard deviation would be used.



Q18 (5 Points) Describe the structure of a Generative Adversarial Network (GAN). Is it supervised or unsupervised?

Solution:

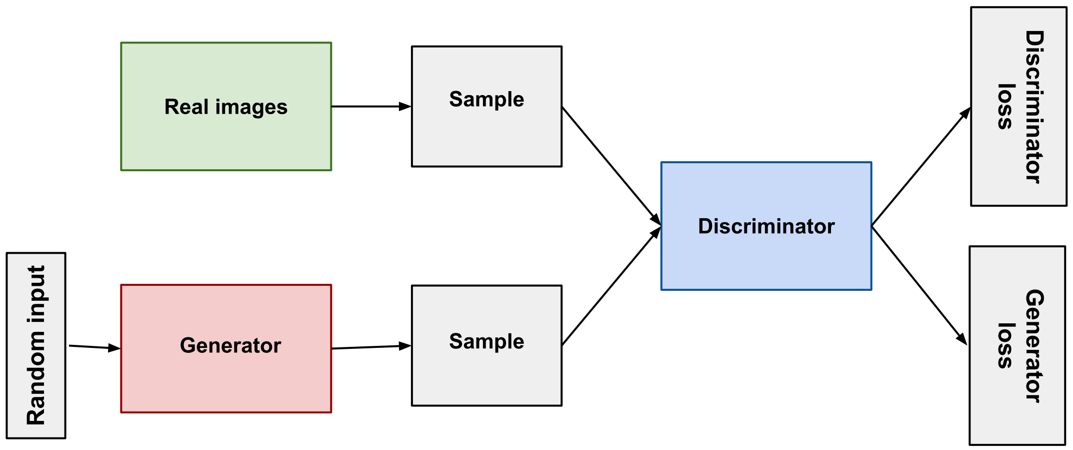
GANs are unsupervised learning algorithms that use a supervised loss as part of the training.

A generative adversarial network (GAN) has two parts:

- The **generator** learns to generate plausible data. The generated instances become negative training examples for the discriminator.
- The **discriminator** learns to distinguish the generator's fake data from real data. The discriminator penalizes the generator for producing implausible results.

When training begins, the generator produces obviously fake data, and the discriminator quickly learns to tell that it's fake. As training progresses, the generator gets closer to producing output that can fool the discriminator: Finally, if generator training goes well, the discriminator gets worse at telling the difference between real and fake. It starts to classify fake data as real, and its accuracy decreases.

Here's a picture of the whole system:



Both the generator and the discriminator are neural networks. The generator output is connected directly to the discriminator input.

Through backpropagation, the discriminator's classification provides a signal that the generator uses to update its weights.

See https://developers.google.com/machine-learning/gan/gan_structure

Q20 (5 Points) pixels on a single screen capture are being used as input and an MLP for the deep learning. How could one input states that captured the motion? How could one input states identified the things in the screen rather than individual pixels?

Solution:

To capture motion, one would have to input several frames rather than a single frame. Say the last few frames.

To identify the things on the screen one would need to use more sophisticated deep learning that detected and localized the elements in an image such as deep learning based semantic segmentation.