
بسم الله الرحمن الرحیم

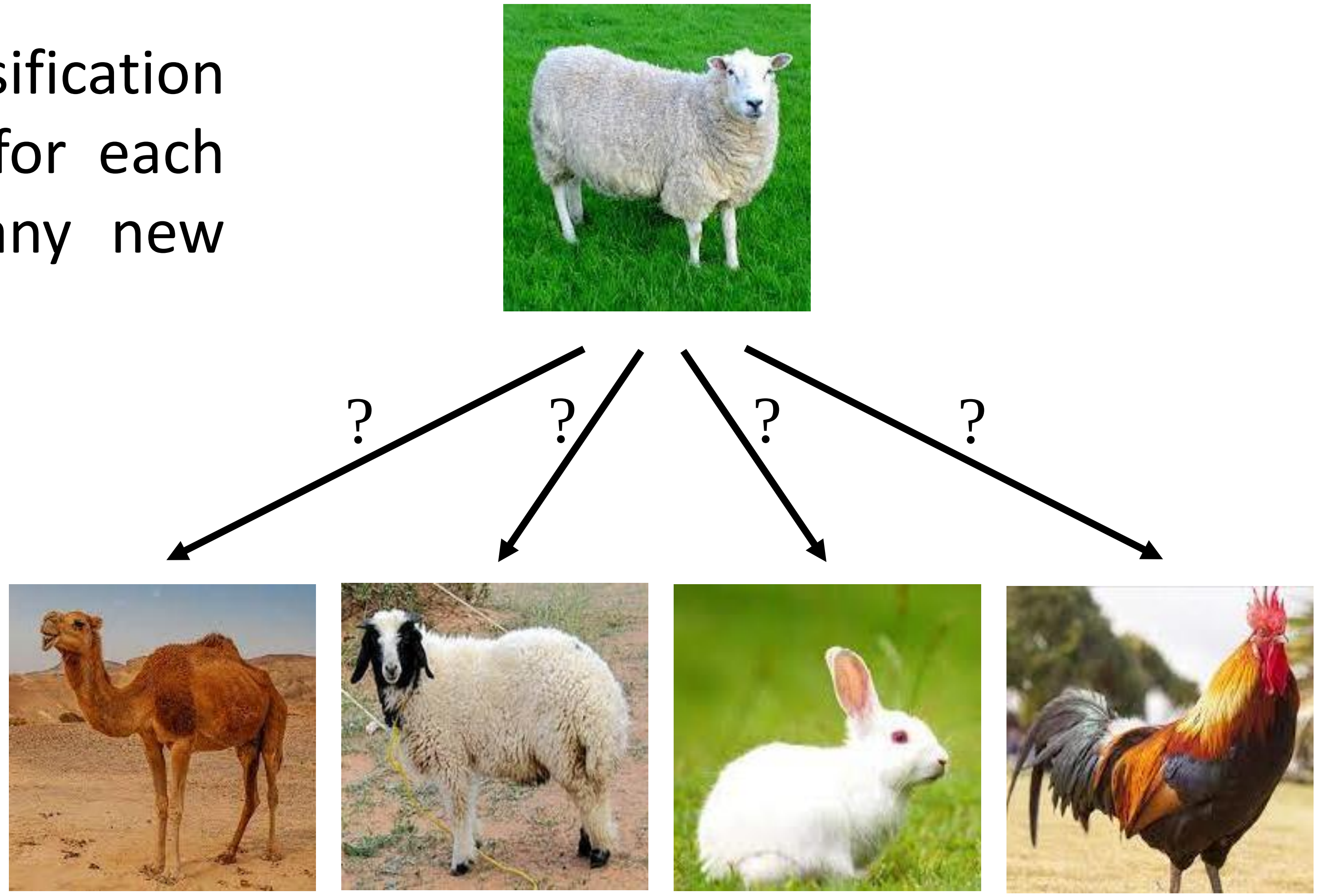
Deep Learning

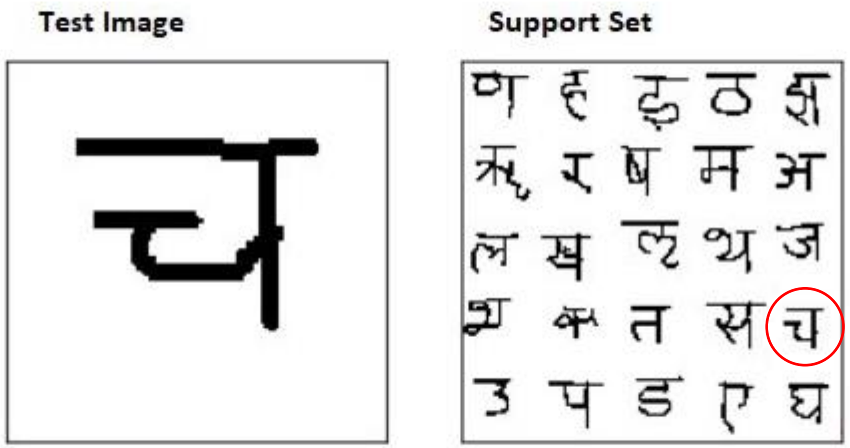
Mohammad Reza Mohammadi

2021

One shot learning

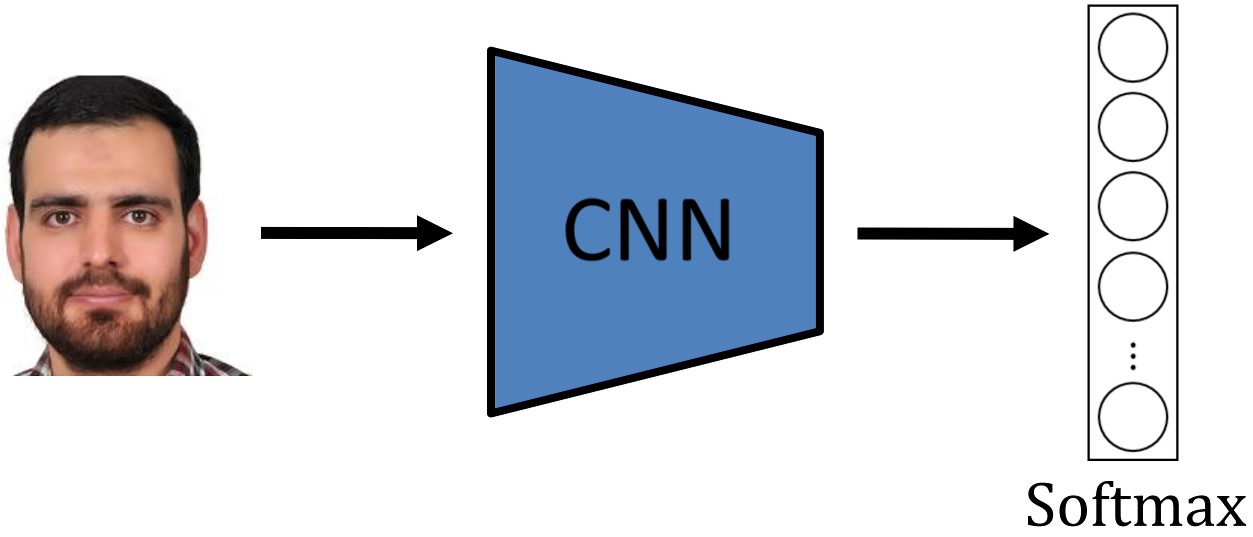
- One-shot learning is a classification task in which one example for each class is used to classify many new examples





One shot face recognition

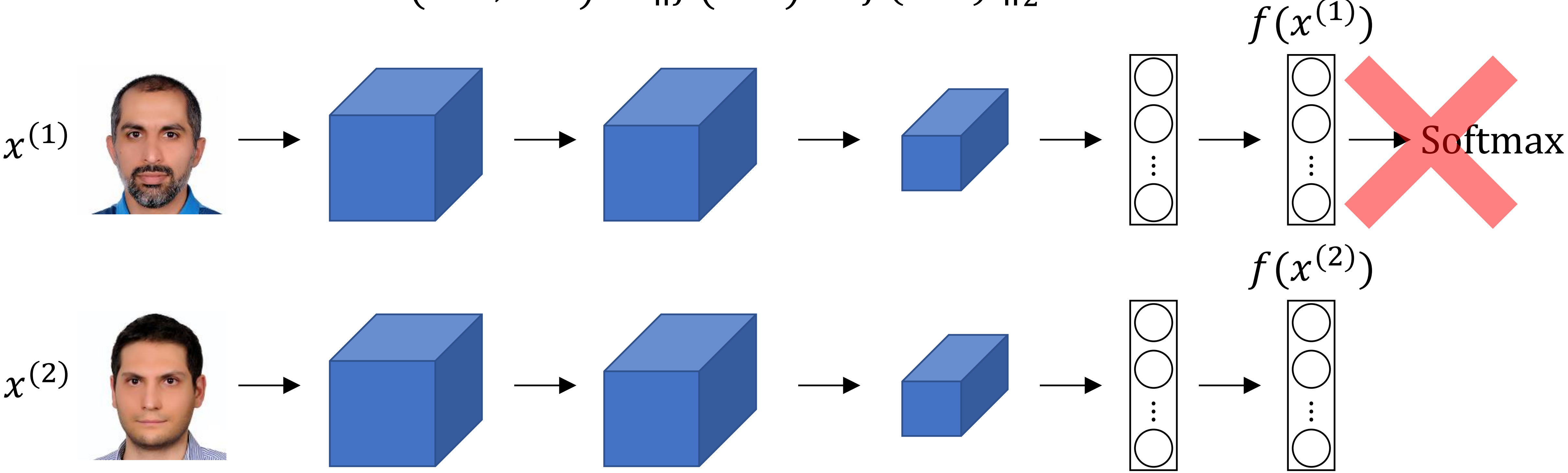
- Learning from example to recognize the person again
  - Is a ConvNet + Softmax well for one shot face recognition?
    - Not enough data to train a robust neural network
    - What if a new person joins our team?
  - Instead, we are going to learn a “similarity” function



Learning a similarity function

- $d(img1, img2)$ = degree of difference between images

$$d(x^{(1)}, x^{(2)}) = \| f(x^{(1)}) - f(x^{(2)}) \|_2^2$$

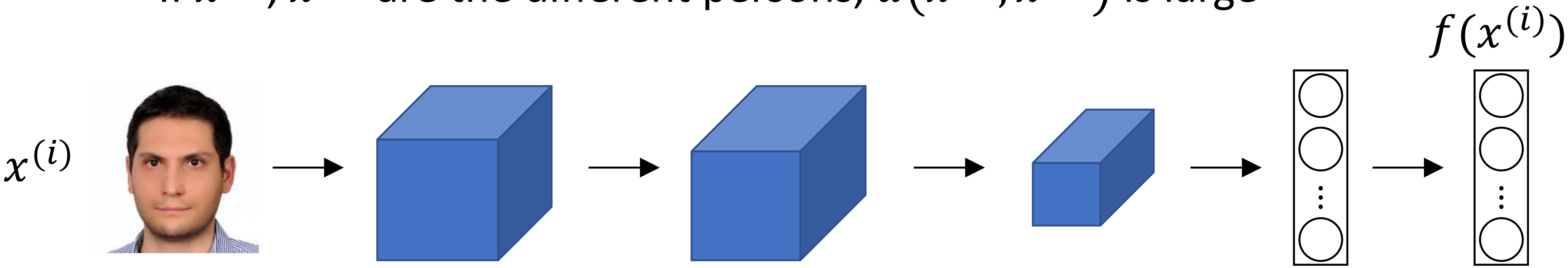


Siamese network

- $d(img1, img2)$ = degree of difference between images

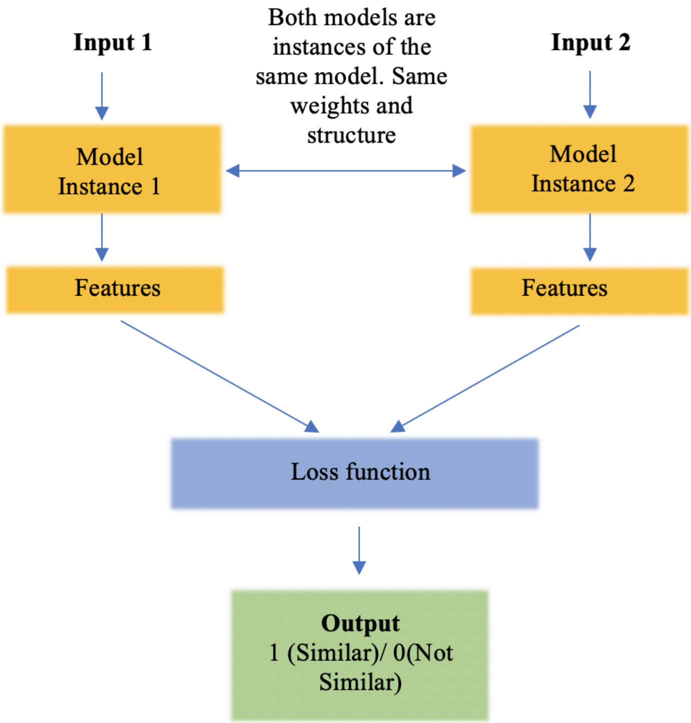
$$d\left(x^{(1)}, x^{(2)}\right) = \| f\left(x^{(1)}\right) - f\left(x^{(2)}\right) \|_2^2$$

- Learn network parameters so that:
  - If $x^{(i)}$, $x^{(j)}$ are the same person, $d\left(x^{(i)}, x^{(j)}\right)$ is small
  - If $x^{(i)}$, $x^{(j)}$ are the different persons, $d\left(x^{(i)}, x^{(j)}\right)$ is large



Siamese network

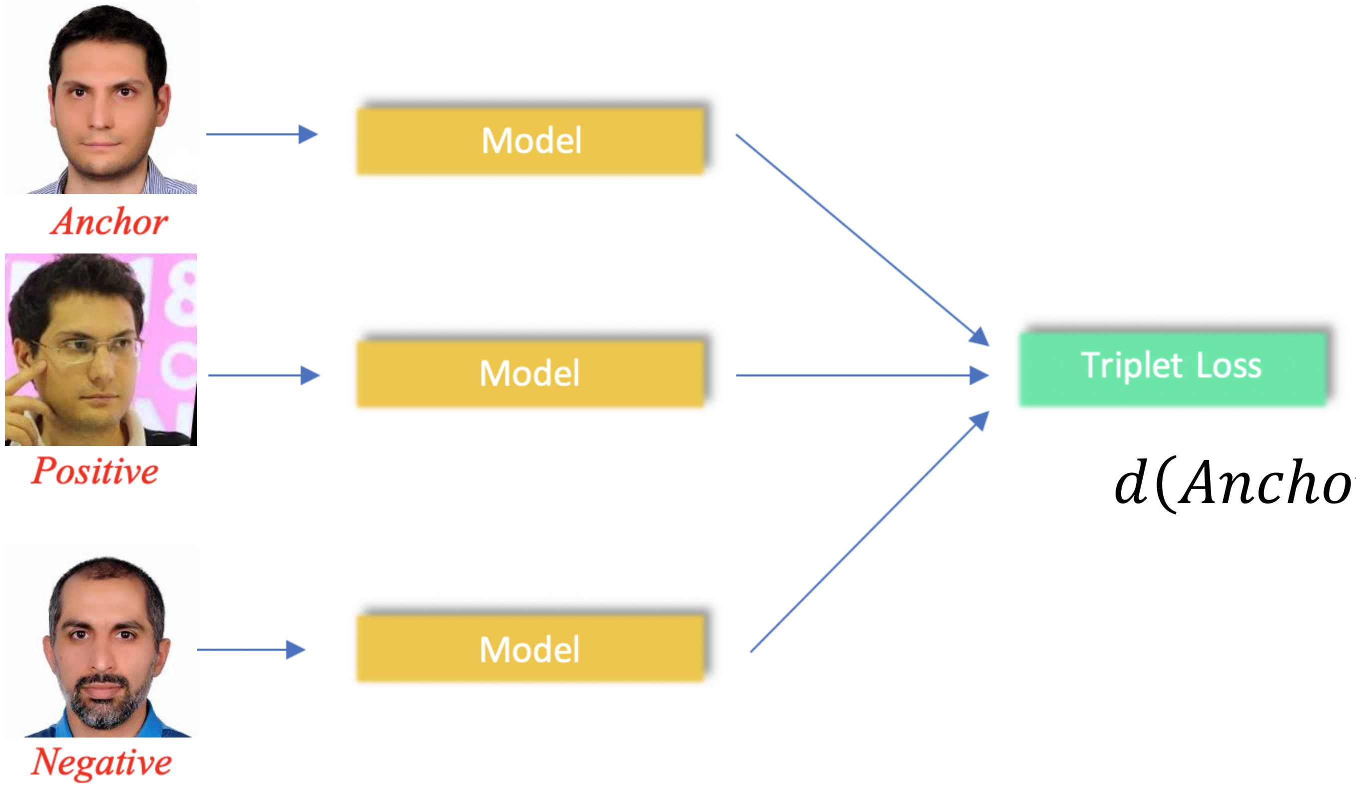
- A Siamese network is a class of neural networks that contains one or more identical networks



$x^{(1)}$	$x^{(2)}$	y
		0
		1
		0
		1

Triplet loss

- The model takes three inputs:
  - anchor, positive, and negative



$x^{(1)}$	$x^{(2)}$	y
		0
		1

$$d(Anchor, Positive) + \underbrace{\alpha}_{\text{margin}} \leq d(Anchor, Negative)$$

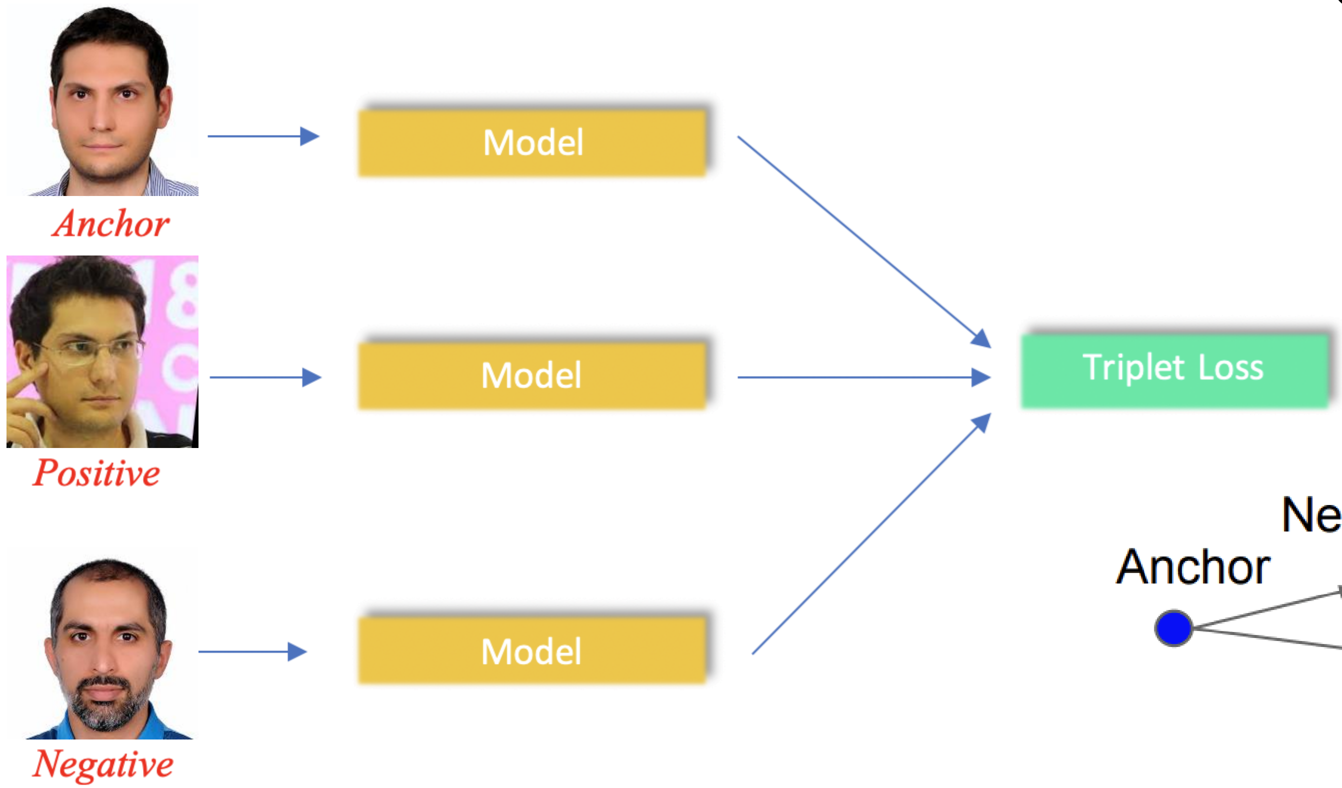
Triplet loss

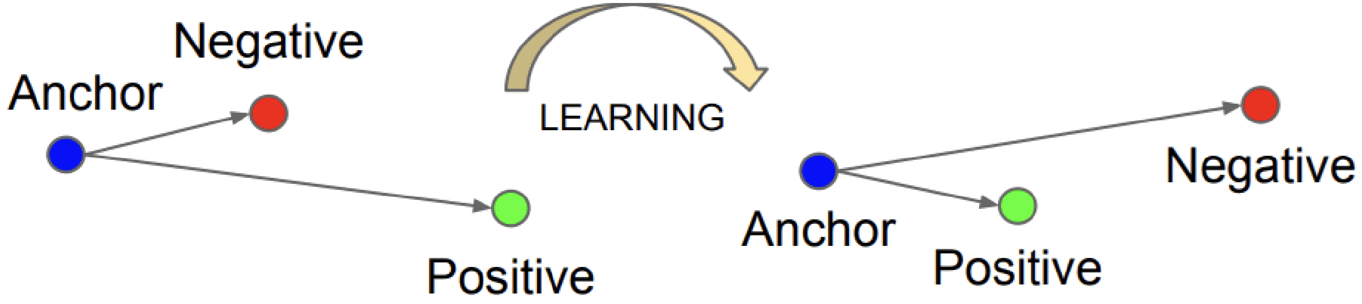
$$d(A,P) + \alpha \le d(A,N)$$

$$d(A,P) - d(A,N) + \alpha \le 0$$

$$\mathcal{L}(A,P,N) = \max(d(A,P) - d(A,N) + \alpha, 0\ )$$

$$J = \sum_{i=1}^{M} \mathcal{L}\left(A^{(i)}, P^{(i)}, N^{(i)}\right)$$





Triplet loss

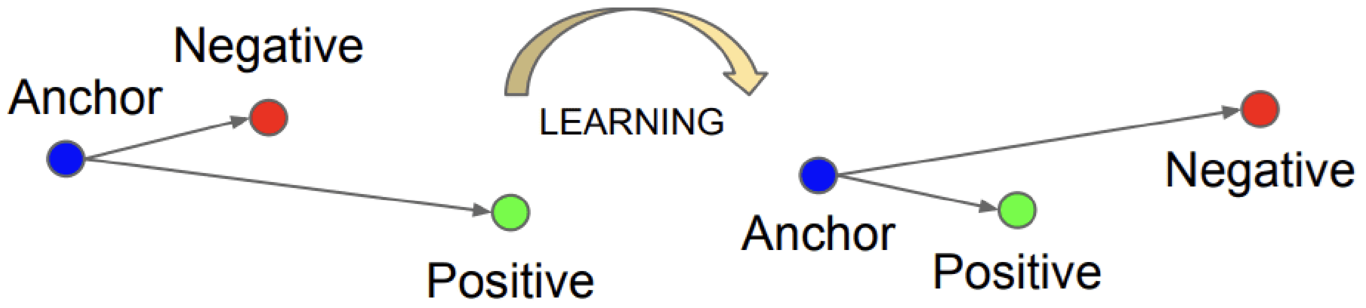
- Similar samples produce embeddings that have small distances (e.g. can be clustered)
- The system directly learns a mapping from face images to a compact Euclidean space where distances directly correspond to a measure of face similarity

$$d(A,P) + \alpha \leq d(A,N)$$

$$d(A,P) - d(A,N) + \alpha \leq 0$$

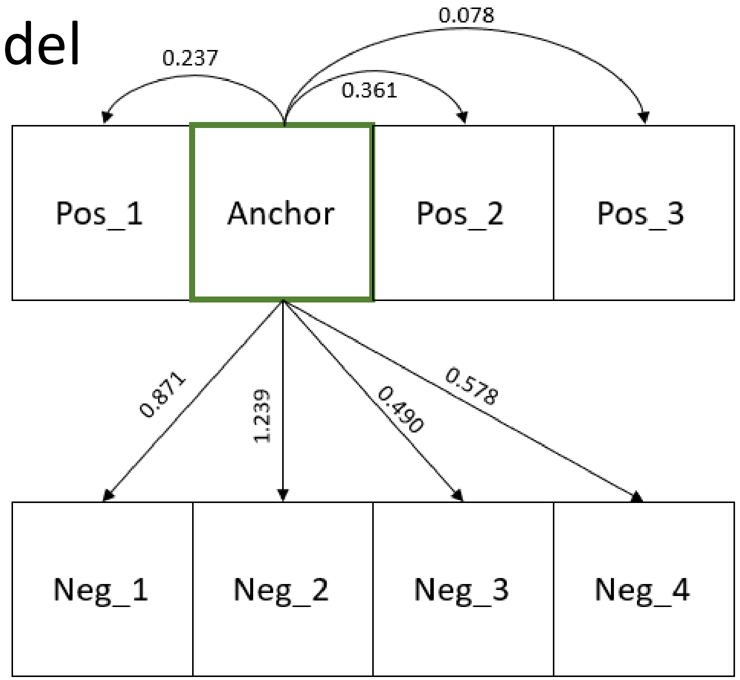
$$\mathcal{L}(A,P,N) = \max(d(A,P) - d(A,N) + \alpha, 0\ )$$

$$J = \sum_{i=1}^{M} \mathcal{L}\left(A^{(i)}, P^{(i)}, N^{(i)}\right)$$



Choosing the triplets

- The triplets that are used to train the model should carefully chosen
- If triplets are chosen randomly
  - $d(A,P) + \alpha \leq d(A,N)$ is easily satisfied!
  - result in a small loss, and are not effective at updating the model
- Triplets are generated in an online manner, and so-called hard positive (matching) and hard negative (non-matching) cases are found and used in the estimate of the loss for the batch



FaceNet

- CASIA-Webface (10K ids/0.5M images)
- VGG2 (9K ids/3.31M images)
- Glint360K (360K ids/17M images)

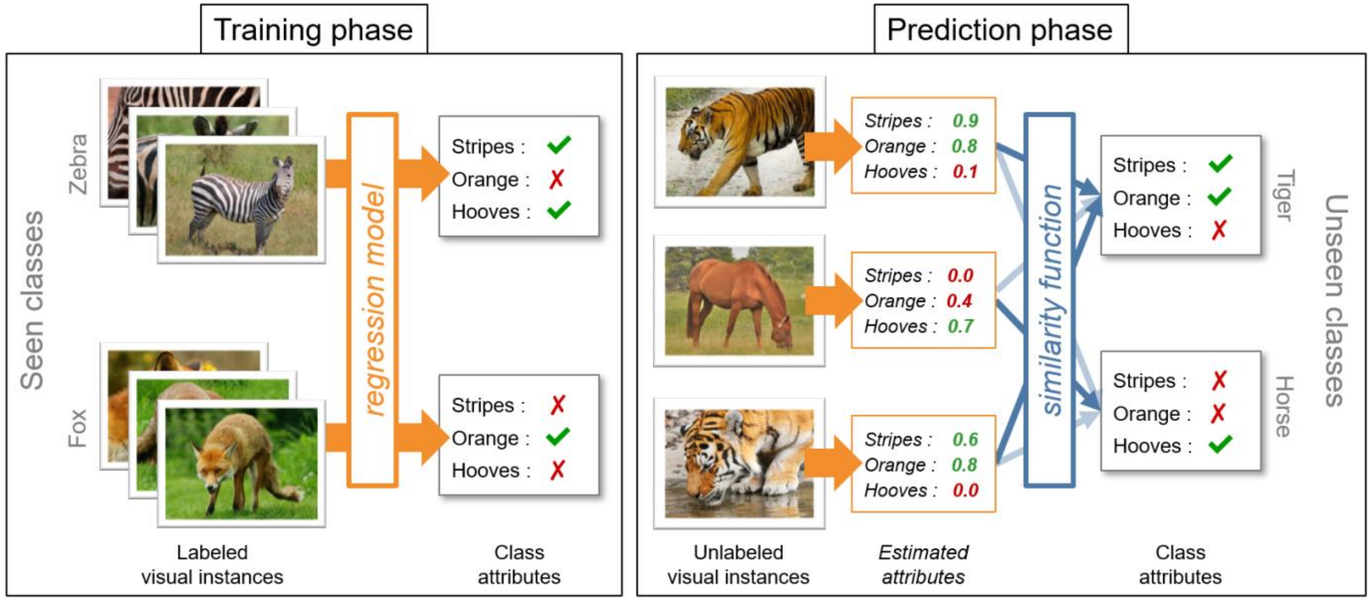
- LFW (5749 ids/13233 images/6K pairs)

Model name	LFW accuracy	Training dataset	Architecture
20180408-102900	0.9905	CASIA-WebFace	Inception ResNet v1
20180402-114759	0.9965	VGGFace2	Inception ResNet v1

https://github.com/davidsandberg/facenet

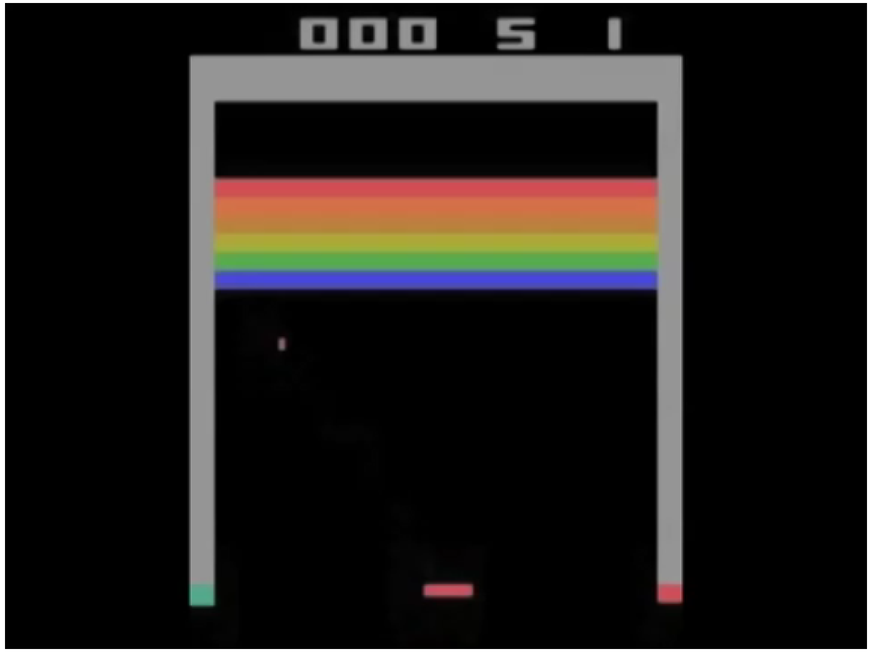
Zero-shot learning

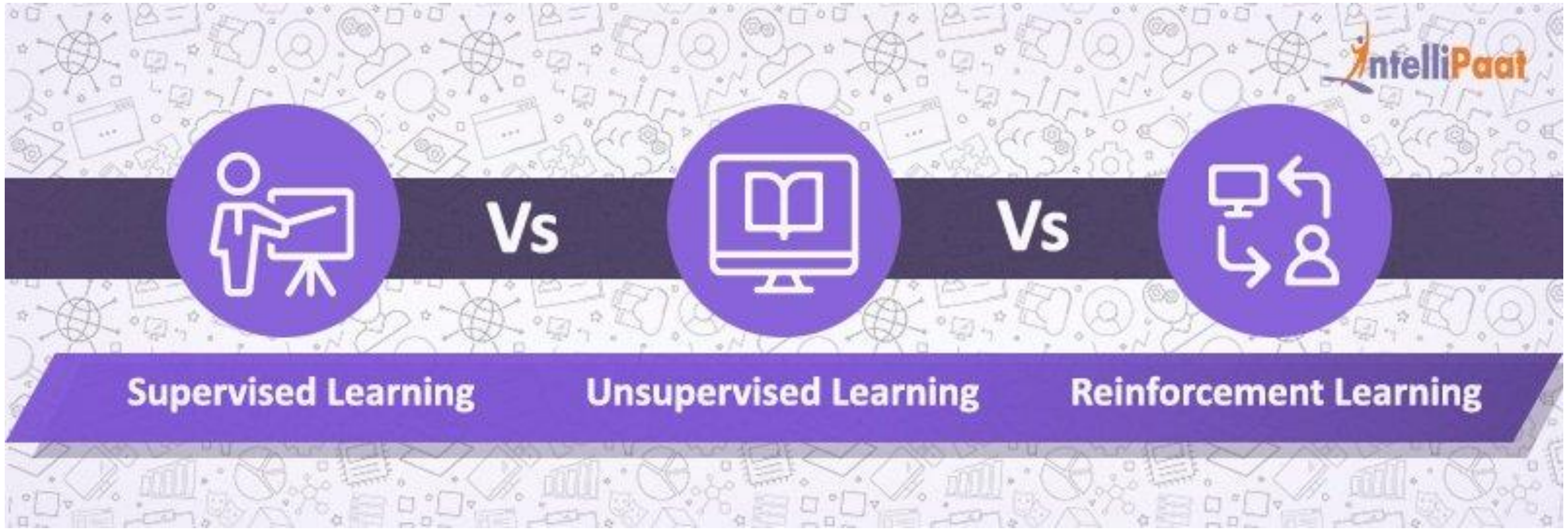
- No labeled examples are given at all!
- As an example, consider the problem of having a learner read a large collection of text and then solve object recognition problems
- For example, having read that “cats have four legs” or “cats have pointy ears”



Deep Reinforcement Learning

Reinforcement learning





Reinforcement learning

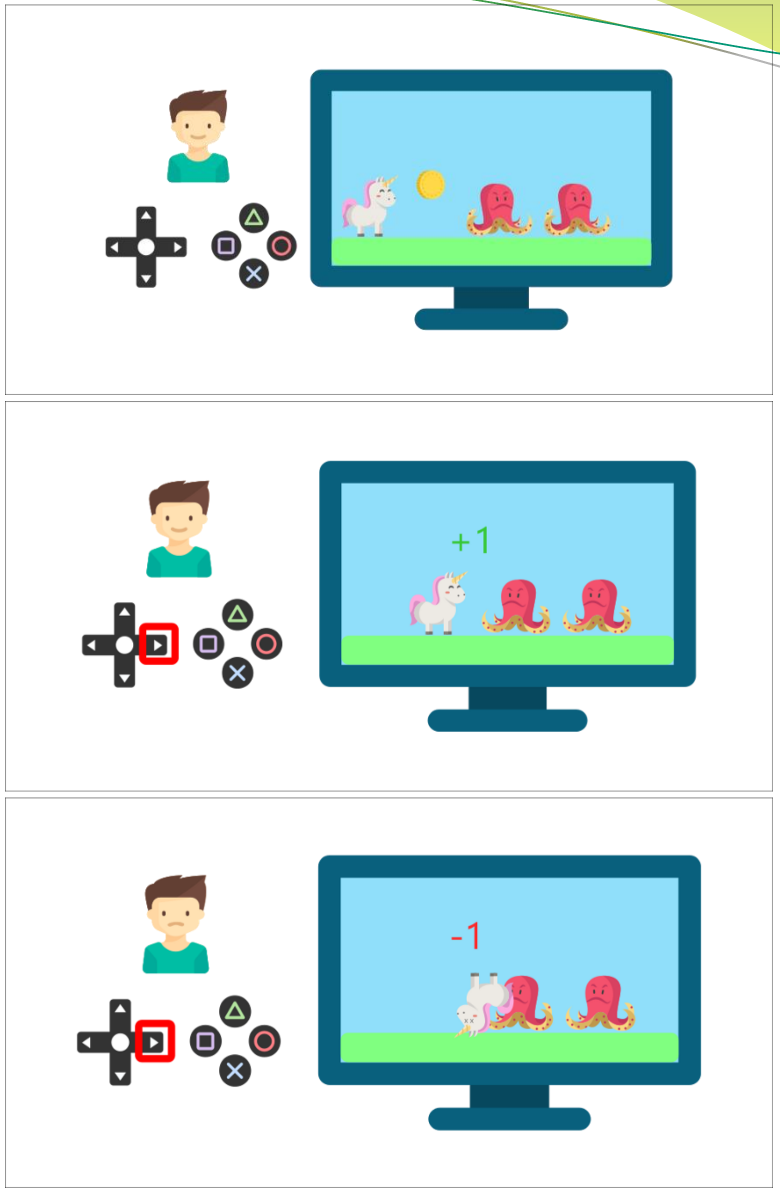
- Deep RL is a type of Machine Learning where an agent learns how to behave in an environment by performing actions and seeing the results
- The idea behind Reinforcement Learning is that an agent (an AI) will learn from the environment by interacting with it (through trial and error) and receiving rewards (negative or positive) as feedback for performing actions

https://thomassimonini.medium.com/an-introduction-to-deep-reinforcement-learning-17a565999c0c

Reinforcement learning

- For instance, imagine you put your little brother in front of a video game he never played, a controller in his hands, and let him alone
  - He presses the right button (action)
  - He got a coin, that’s a +1 reward
  - He must get the coins!

  - He presses the right button again
  - He touches an enemy, he just died -1 reward
  - He must avoid the enemies!

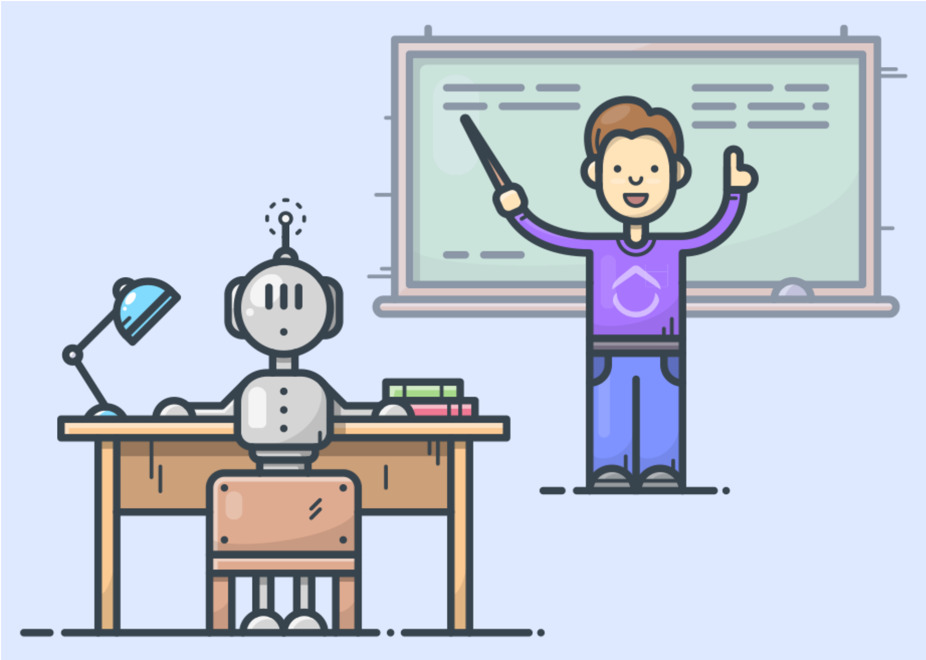


Reinforcement learning

- RL is learning what to do so as to maximize a numerical reward signal
- The learner is not told which actions to take
- It must discover which actions yield the most reward by trying them
- Two most important distinguishing features of reinforcement learning are:
  - Trial-and-error search
  - Delayed reward

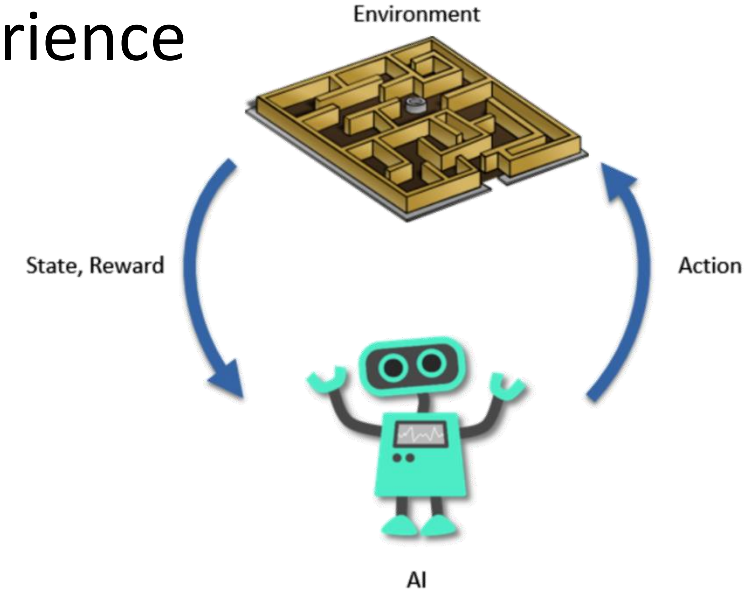
Reinforcement Learning vs Supervised Learning

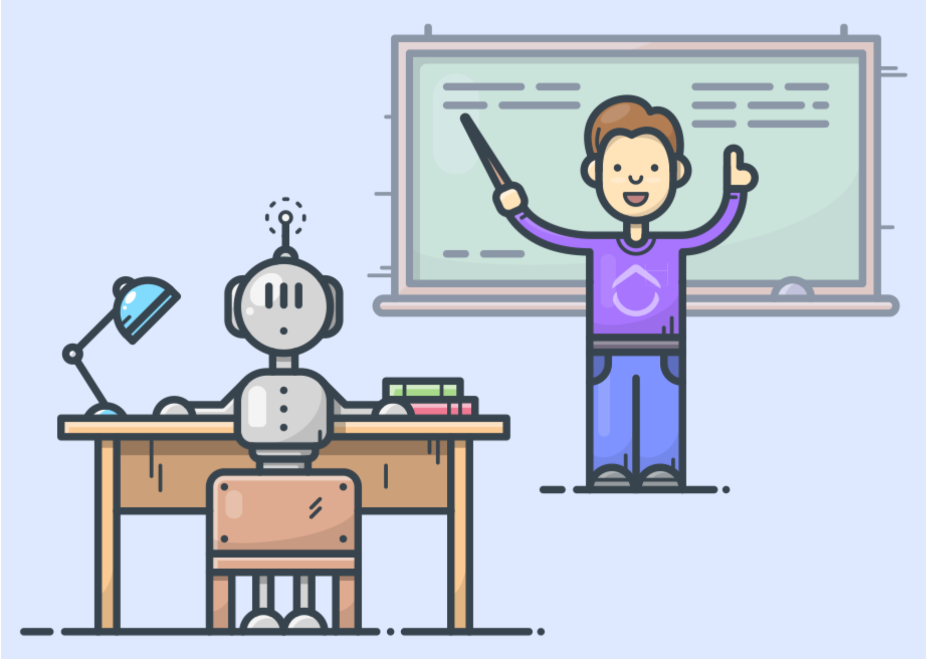
- Supervised learning is learning from a training set of labeled examples provided by a knowledgeable external supervisor
- Each example is a description of a situation together with a specification (label) of the correct action
- The learner should extrapolate, or generalize, its responses so that it acts correctly in situations not present in the training set



Reinforcement Learning vs Supervised Learning

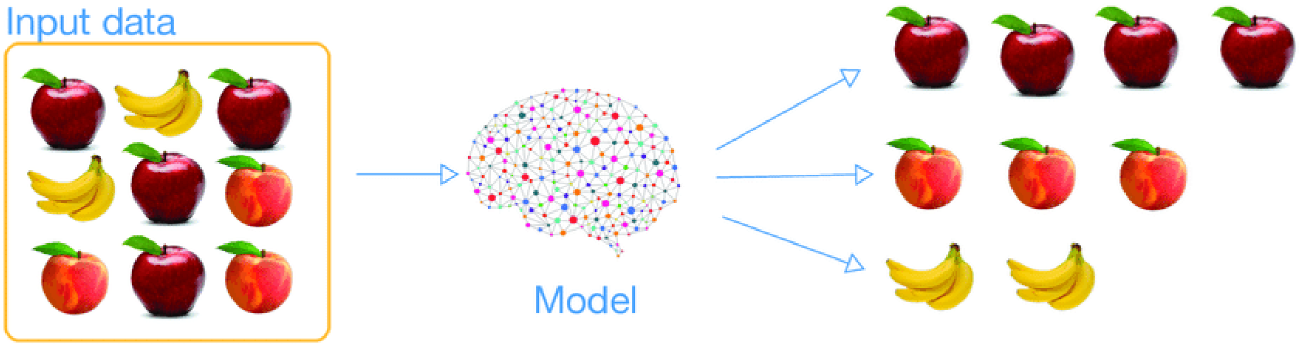
- In interactive problems it is often impractical to obtain examples of desired behavior that are both correct and representative of all the situations in which the agent has to act
- In uncharted territory, an agent must be able to learn from its own experience

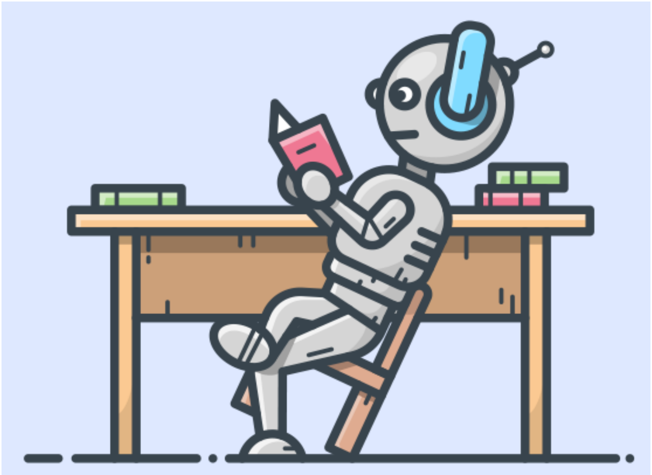




Reinforcement Learning vs Unsupervised Learning

- Unsupervised learning is typically about finding structure hidden in collections of unlabeled data
- One might be tempted to think of reinforcement learning as a kind of unsupervised learning
- Reinforcement learning is trying to maximize a reward signal instead of trying to find hidden structure





RL process

- Our agent receives state S0 from the environment
- Based on that state S0, the agent takes an action A0
- Environment transitions to a new state S1
- Environment gives some reward R1 to the agent

- The goal of the agent is to maximize its cumulative reward
  - called the expected return