April 12, 2023

DSML: Computer Vision.

Siamese Networks: One-shot learning.

Class starts @ 9:05 pm.

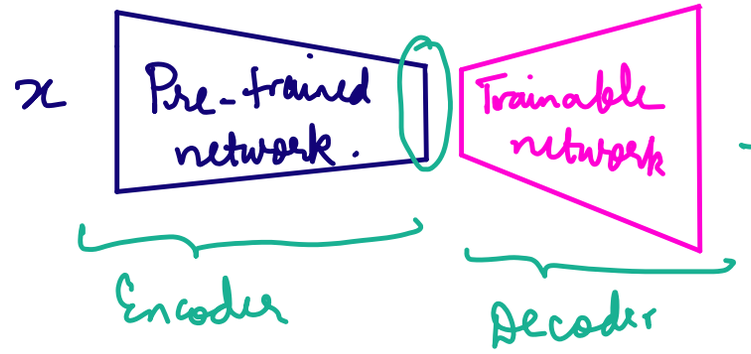
Recap:

* Image Segmentation

(a) Differences between Segmentation & Detection.

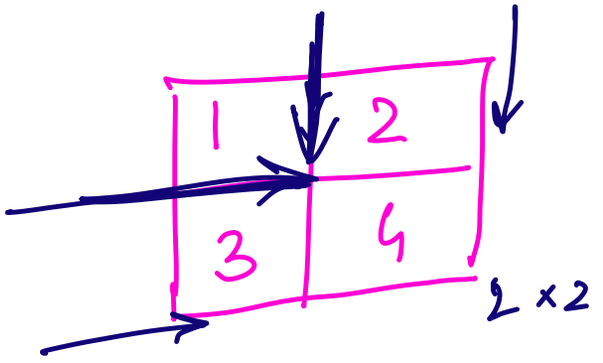
* CNN architecture (Output layer): 2 heads → classification, Regression
  ↳ same size masks as the input image. ≈ 5.
* Evaluation metric: IoU. DICE coefficient.
* Loss: Combination of classification & ~~regression~~.
  CCE Loss

(b) FCN 8 architecture.



$\hat{x}$ (segmentation maps, 1 per class).

$\hat{y}$ (labels per class)



Upsampling →

1	0	2	0
0	0	0	0
3	0	4	0
0	0	0	0

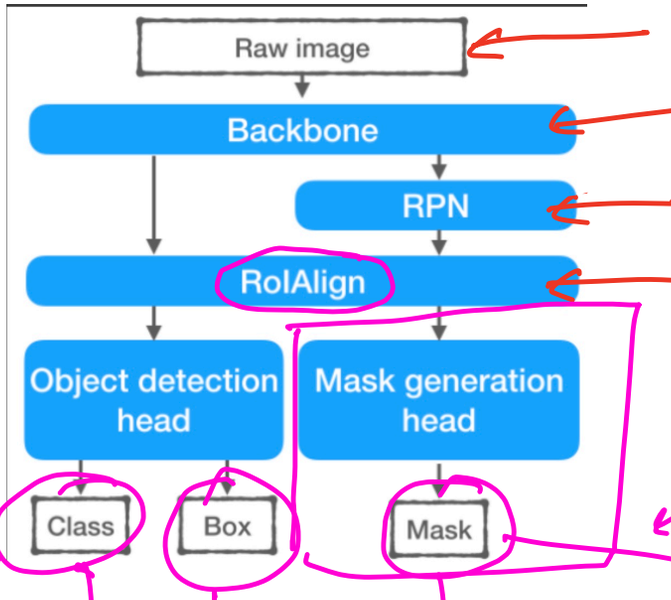
4 x

batch x h x w x channels

h x w: no change

channels: ↑

Faster R-CNN.



Region proposal.

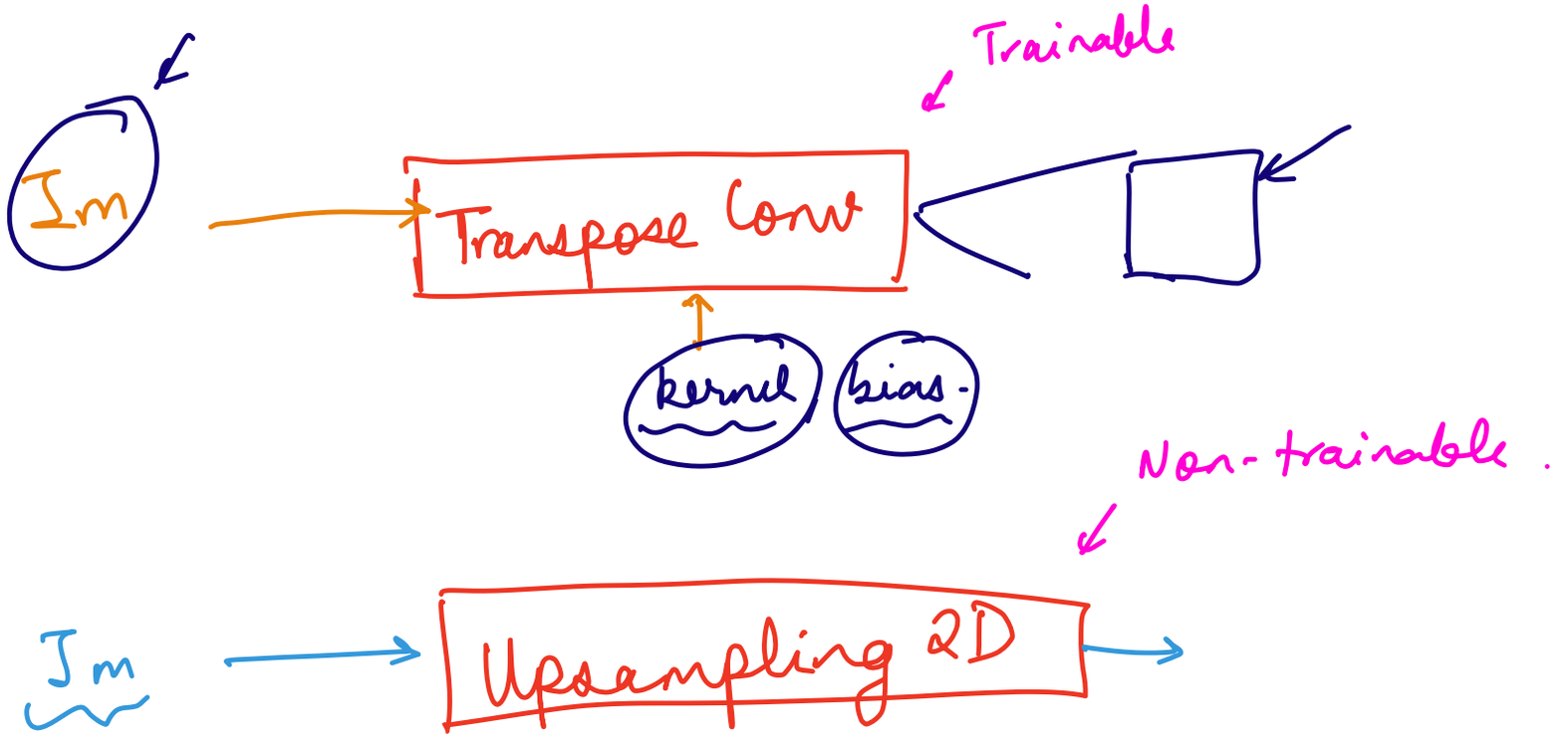
Region of interest alignment layer.

CCE.

Something like CCE.

MSE.

New component which allows us to do segmentation.


Trainable
Im
Transpose Conv
kernel
bias
Non-trainable.
Im
Upsampling 2D

Object Classification.

One-shot learning:

Are these the same object?

Business Case:

→ Start a startup with 26 employees

→ Facial recognition software

↳ Employee → Name of the person.

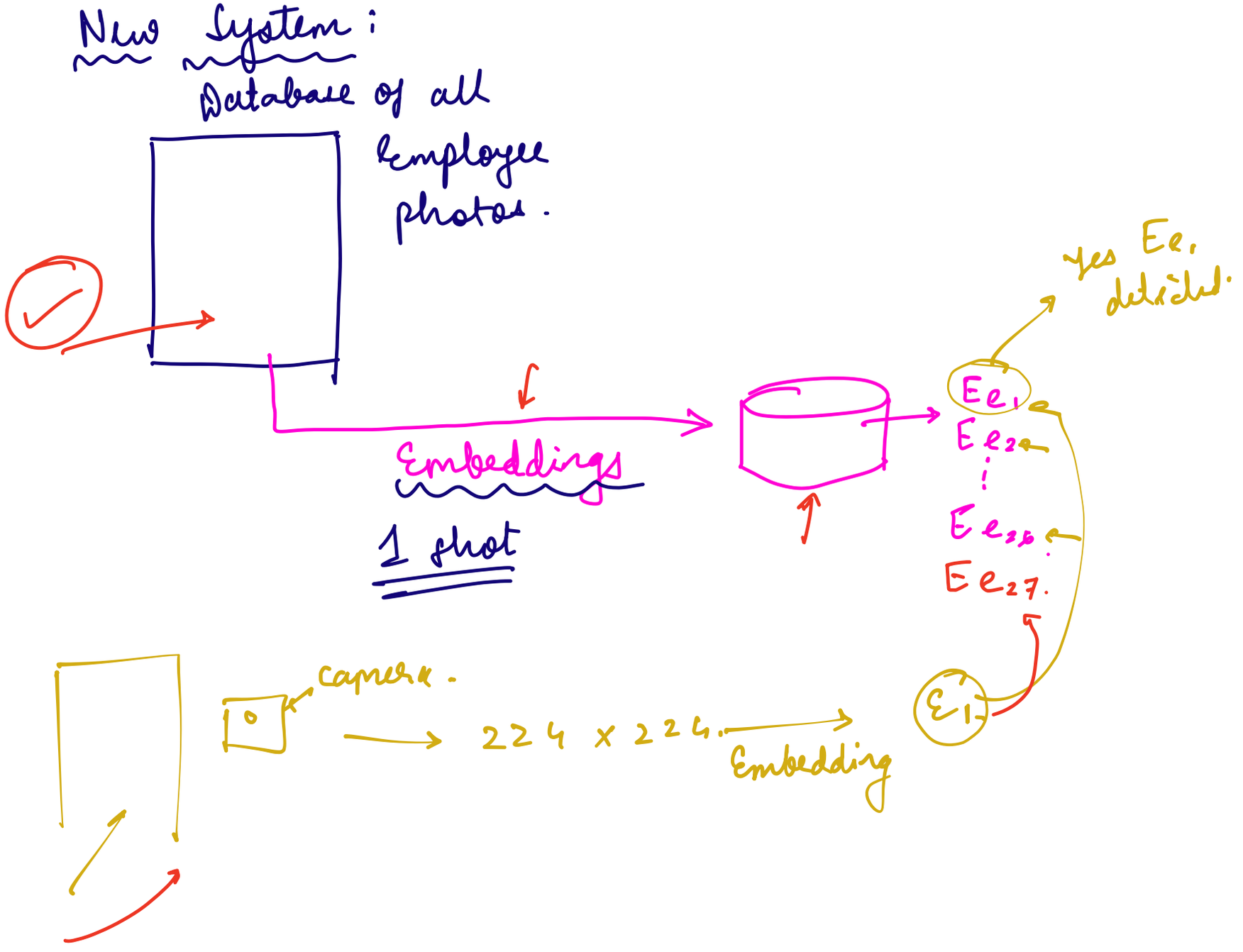
or

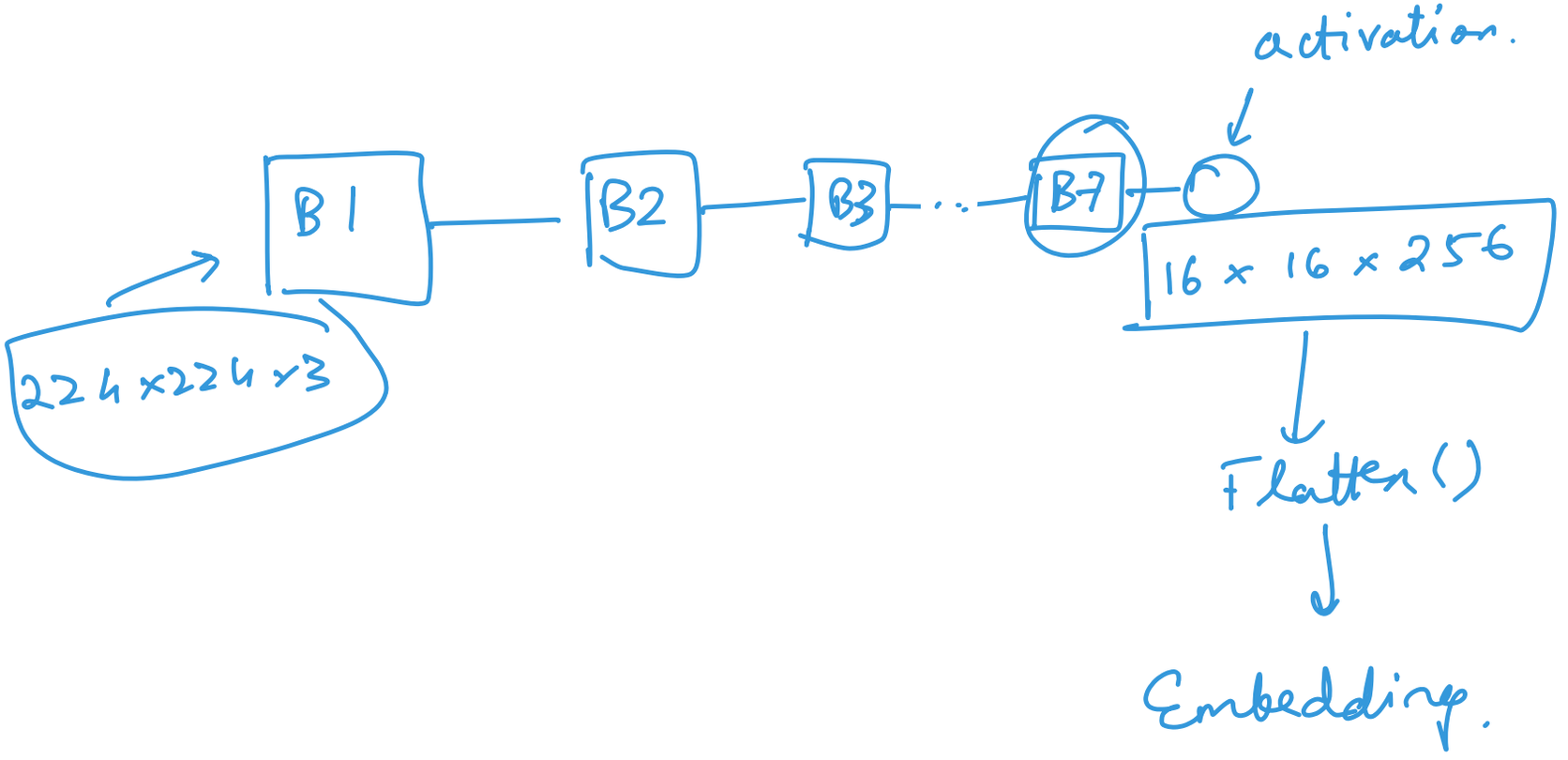
Not Employee.

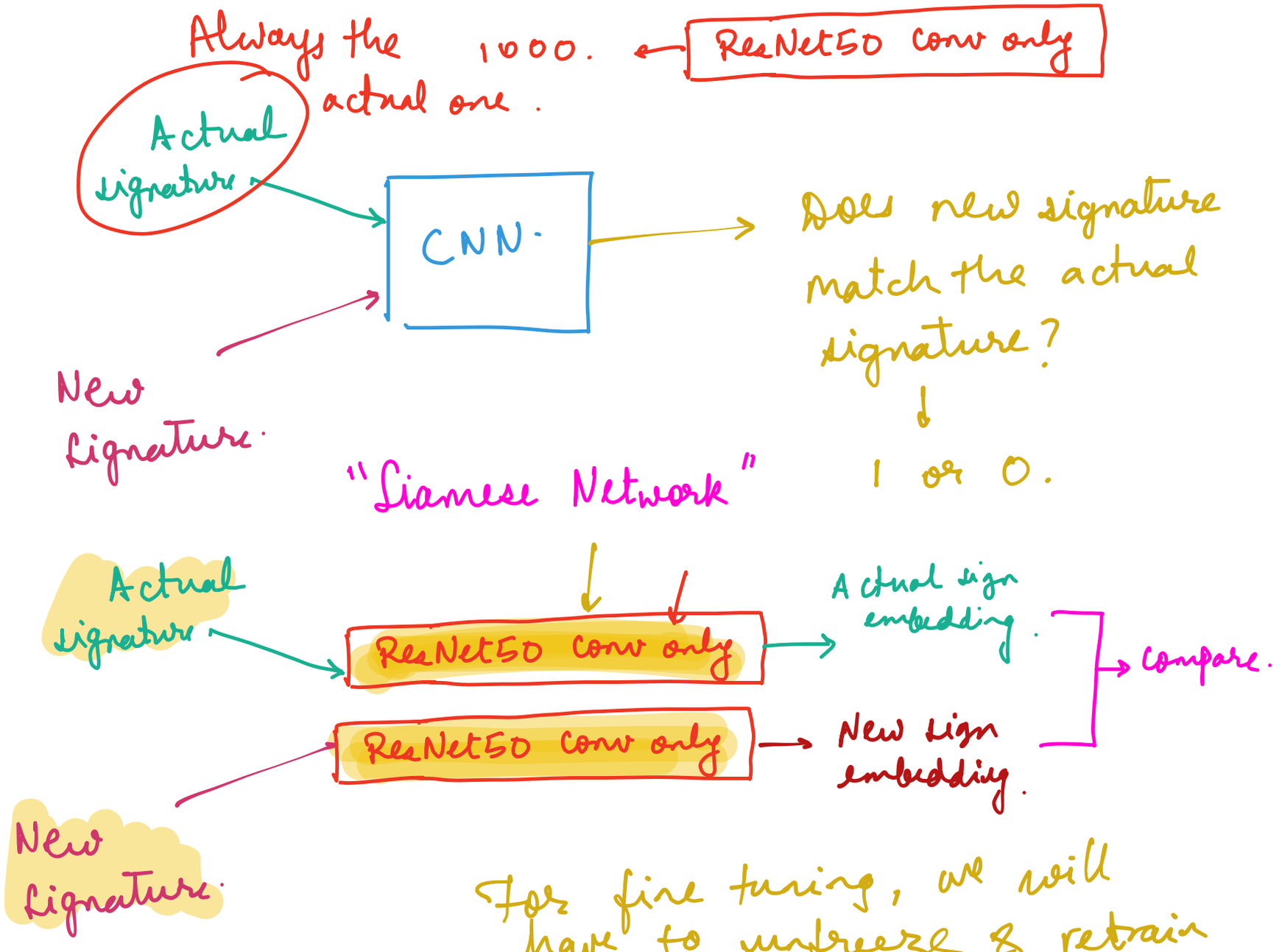
↕ retrain?

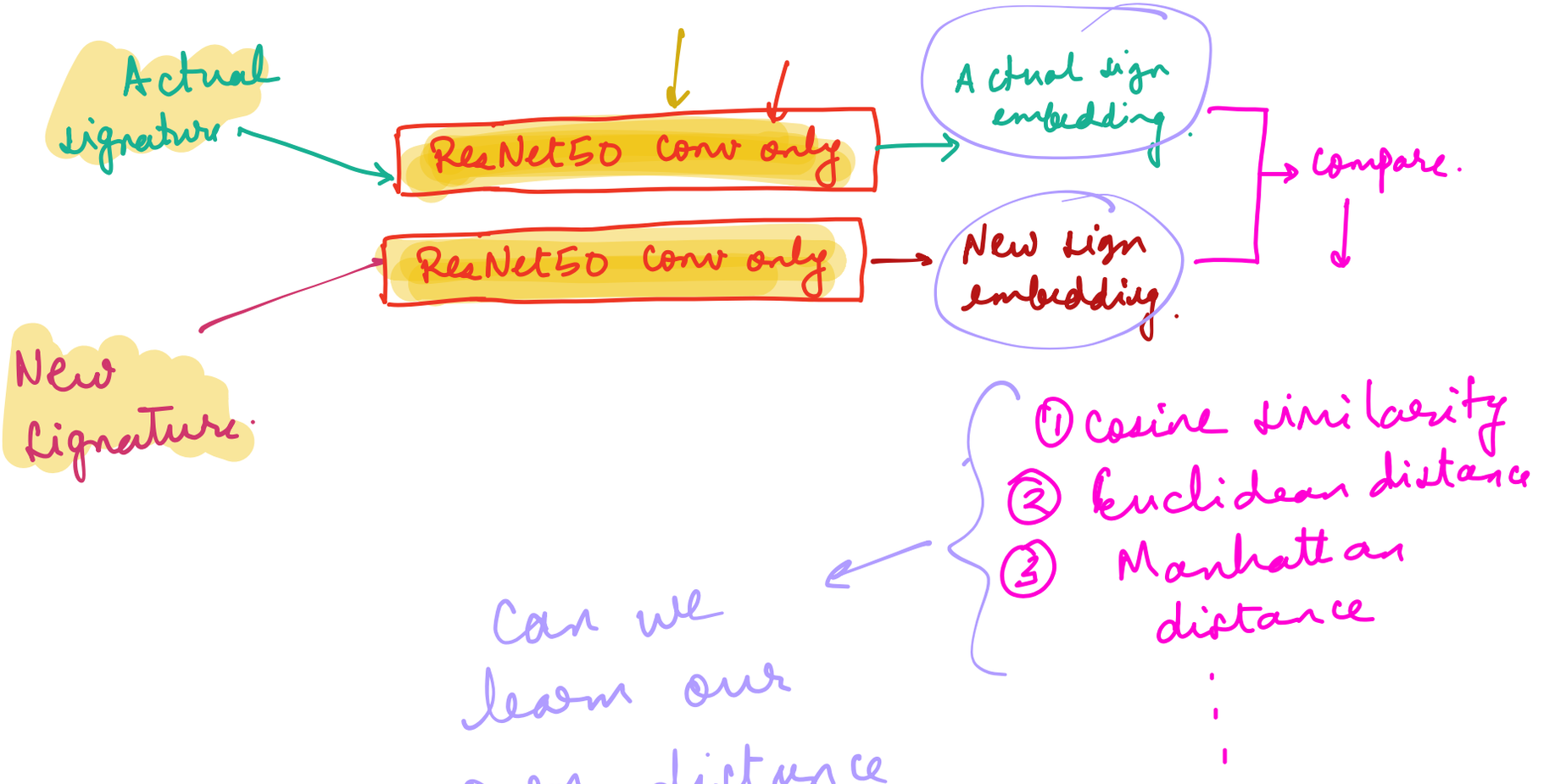
→ 1 new employee.

Problem: We will have to retrain the network every time we get new employees.


New System :
Database of all employee photos.
Embeddings
1 shot
yes Ee, detected.
Ee1
Ee2
Ee25
Ee27.
camera.
224 x 224.
Embedding
E1


activation.
B1
B2
B3
B7
224 x 224 x 3
16 x 16 x 256
Flatten()
Embedding.


Always the actual one.
1000.
ResNet50 conv only
Actual signature
CNN.
New signature.
Does new signature match the actual signature?
1 or 0.
"Siamese Network"
Actual signature
ResNet50 conv only
Actual sign embedding.
Compare.
ResNet50 conv only
New sign embedding.
New signature.
For fine tuning, we will have to unfreeze & retrain the network!!


Actual signature
ResNet50 Conv only
Actual sign embedding
Compare.
ResNet50 Conv only
New sign embedding
New signature.
① cosine similarity
② Euclidean distance
③ Manhattan distance
Can we learn our own distance metric?

Cases for One-shot learning:

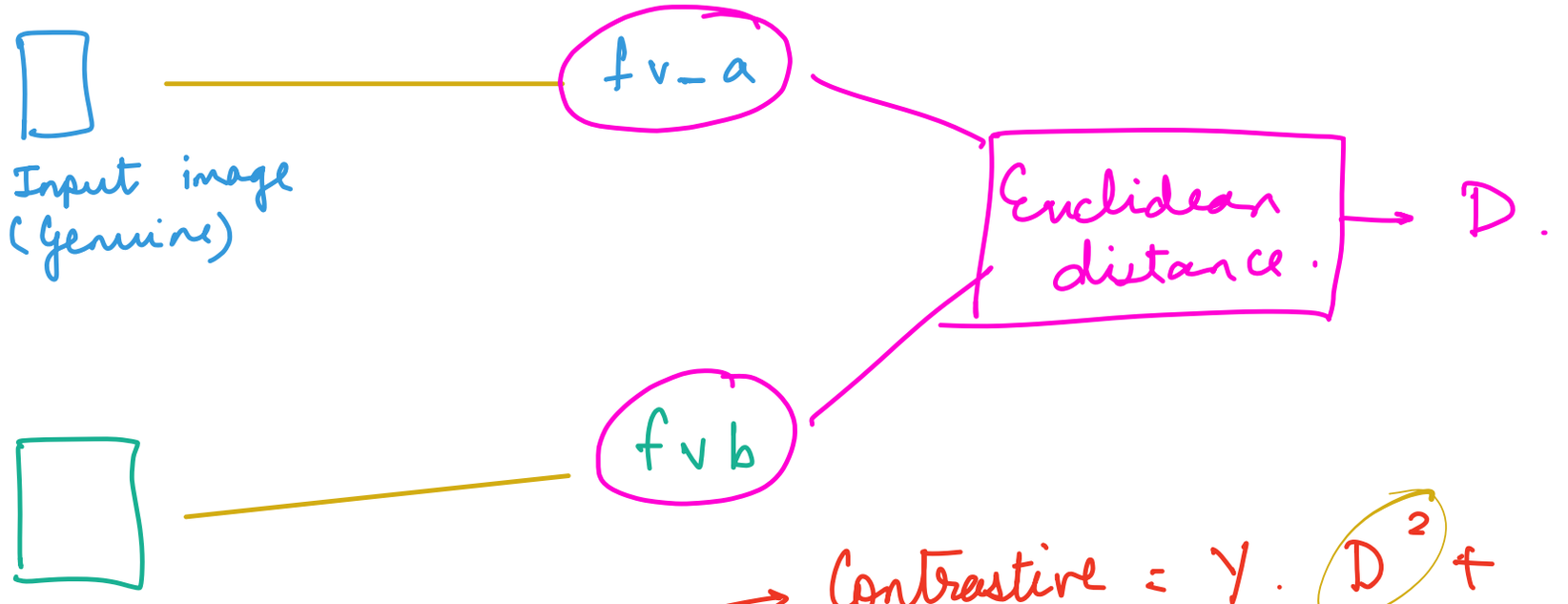
Case 1: Genuine vs. Genuine. } We want to keep these close together

1.7, 2.2, 3.8, 3.6, ...

Case 2: Genuine vs. Fake. } Keep these far apart.

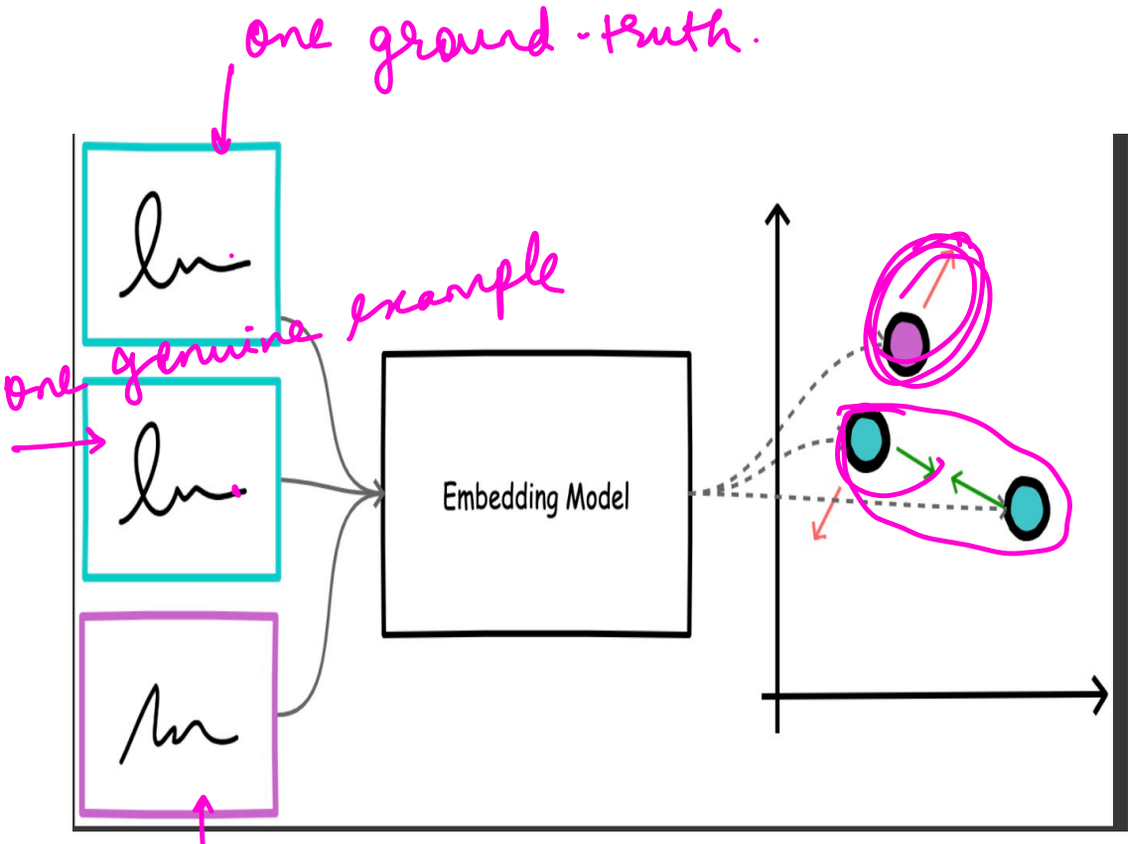
.... 22.7, 36.5, 44.8, 50.3...

Contrastive loss.



min. $D^2 \rightarrow (0.02)^2 \; (0.04)^2 \; (0.06)^2$

min. $(1-D)^2 \rightarrow (0.98)^2 \quad (0.96)^2 \; (0.94)^2$



(Anchor)

→ Ground truth image im_gt.

→ Genuine image. im_g

→ Forged image. im_f

Distance (im_gt, im_g) $\leq$ Distance (im_gt, im_f)

max (Distance (im_gt, im_g) − Distance (im_gt, im_f), 0) $\neq 0$

Triplet Loss:

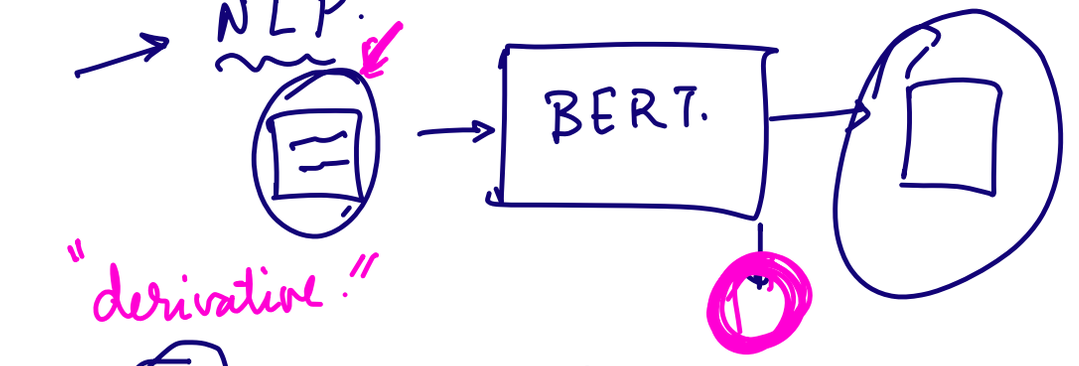
- A → Anchor embedding, Ground truth.
- P → Positive embedding, Genuine image.
- N → Negative embedding, Forged image.
- D(x, y) → Euclidean distance between x & y.

Loss: $\max\left(D(A, P) - D(A, N), 0 \right)$

negative - Good, do nothing.

positive - Bad, do an update.

Zero-Shot learning → 9 may see classes not encountered before, and must classify them anyway.

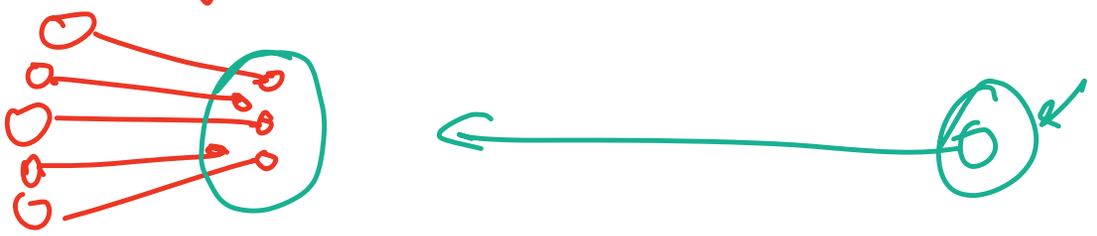


K-Shot learning.

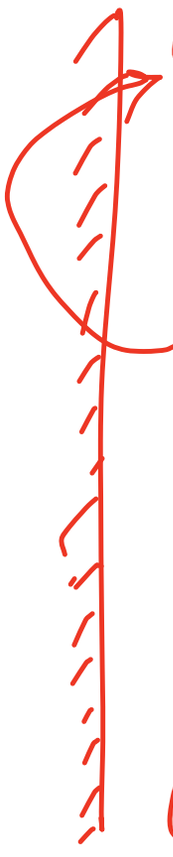
K → No. of training examples of the positive class.

K = 5

(5) photos of cats, dogs, zebras.



Discussion Requests:



(a) Chat GPT.

(b) Image + Text → Multimodal. (Diffusion, Dall-e)

(c) Lang Chain

(d) RL HF
↓
Re-inforcement learning through Human feedback.

(f) 3D related networks