WEEK 4:

Face Recognition: One Shot Learning.

Learning a similarity function: $d(img1, img2)$ = degree of diff b/w images.

$$\left.\begin{array}{l} \text{If } d(img1, img2) \leq \tau \text{ "same"} \\ \qquad\qquad\qquad\quad > \tau \text{ "different"} \end{array}\right\} \text{verification.}$$

Siamese network: (to learn func. 'd')

Triplet loss:

A, P & N images ; $f(x)$ - encoding of x.

Want: $\underbrace{||f(A)-f(P)||^2}_{d(A,P)} \leq \underbrace{||f(A)-f(N)||^2}_{d(A,N)}$

$$||f(A)-f(P)||^2 - ||f(A)-f(N)||^2 \leq 0$$

$0 - 0 \leq 0$ — satisfies ↑ this equation.

So, add margin α, ~~β~~

Anchor - Negative + $\alpha \leq 0$

α - pushes (A,P) & (A,N) pair further apart

If (A,P) = 0.5 & (A,N) = 0.51, the case will pass.

So, to ↑ (A,N) or ↓ (A,P) we use 'α'.

loss function:

Given: 3 images - A, P, N.

$$L(A,P,N) = \max(||f(A)-f(P)||^2 - ||f(A)-f(N)||^2 + \alpha, 0)$$

$$J = \sum_{i=1}^{m} L(A^{(i)}, P^{(i)}, N^{(i)})$$

need pairs of images to train triplets.

Choosing A, P, N: If A, P & N are chosen randomly, $d(A,P) + \alpha \le d(A,N)$ is easily satisfied.

So, choose triplets that are hard to train on: s.t. $d(A,P) \simeq d(A,N)$, so that Gr. D. does work to increase margin.

PAPER: FaceNet

Face Verification & Binary Classification:

$f(x^i)$ - encoding of image x^i

$f(x^j)$ - " " " x^j

$$\begin{matrix} x^i \searrow \\ x^j \nearrow \end{matrix} O \rightarrow \hat{y}$$

last layer computation:

$$\hat{y} = \sigma\left(\sum_k w_i \underbrace{|f(x^i_k) - f(x^j_k)|} + b\right)$$

$$\frac{(f(x^i_k) - f(x^j_k))^2}{f(x^i_k) + f(x^j_k)} = \chi^2$$

In siamese n/w, params to compute $f(x_i)$ & $f(x_j)$ are the same

The encoding is precomputed & stored for all employees to be verified while testing.

NEURAL STYLE TRANSFER :

1) What is style transfer? Painting an image with the style of another.

2) Visualization of deep layers (complex shapes in images) in convnet

3) Cost function: $\underbrace{C \quad S}_{G}$

min $J(G)$: measures how good is $J(G)$ & use G.D to minimize

$J(G) = \alpha J_{content}(C, G) + \beta J_{style}(S, G)$

paper: Gatys et al (A Neural Algo of artistic style)

Algo: i) Initialize G randomly.

ii) use G.D to min $J(G)$: $G := G - \frac{d}{dG} J(G)$

4) Content Cost Function

$J(G) = \alpha J_{content}(C, G) + \beta J_{style}(S, G)$

→ Use hidden layer 'l' to generate content cost.

If $l = 1$, it forces G to have same pixel values as C.

If l = deep, it makes sure if there is an object in C then the same obj. is in G'

$1 < l < L$

$J_{content}(C, G) = \frac{1}{\alpha} \| a^{[l](C)} - a^{[l](G)} \|^2$

$\downarrow$

α

Style cost function:

Style? Correlation between activations across channels.

Intuition: channel 1 captures horizontal ~~line~~ edges
channel 2 captures orange color
⇒ style = orange horizontal edges.

Let $a^{[l]}_{i,j,k}$ = activation at (i,j,k). $G^{[l]}$ is $n_c^{[l]} \times n_c^{[l]}$

$$G^{[l](S)}_{kk'} = \sum_{i=1}^{n_H^{[l]}} \sum_{j=1}^{n_W^{[l]}} a^{[l](S)}_{ijk} \, a^{[l](S)}_{ijk'}$$ → styled img.

$$G^{[l](G)}_{kk'} = \text{"}$$ → generated img.

$$J^{[l]}_{style}(S,G) = \| G^{[l](S)} - G^{[l](G)} \|_F^2$$

$$= \frac{1}{(2 n_H n_W n_c)^2} \sum_k \sum_{k'} \left(G^{[l](S)}_{kk'} - G^{[l](G)}_{kk'} \right)^2$$

$$J_{style}(S,G) = \sum_l \lambda^{[l]} J^{[l]}_{style}(S,G)$$

↳ weighted by λ; use diff weights for earlier simple layers & later layers (hi-level)

$$J(G) = \alpha J_{content}(C,G) + \beta J_{style}(S,G)$$