Face verification vs Face Recognition

verification -

- input image, name / ID
- output whether the input image is that of the claimed person

recognition -

- has a database of K persons
- get an input image
- output ID if the image is of the K persons (or "not recognized")

Learning a "similarity" function

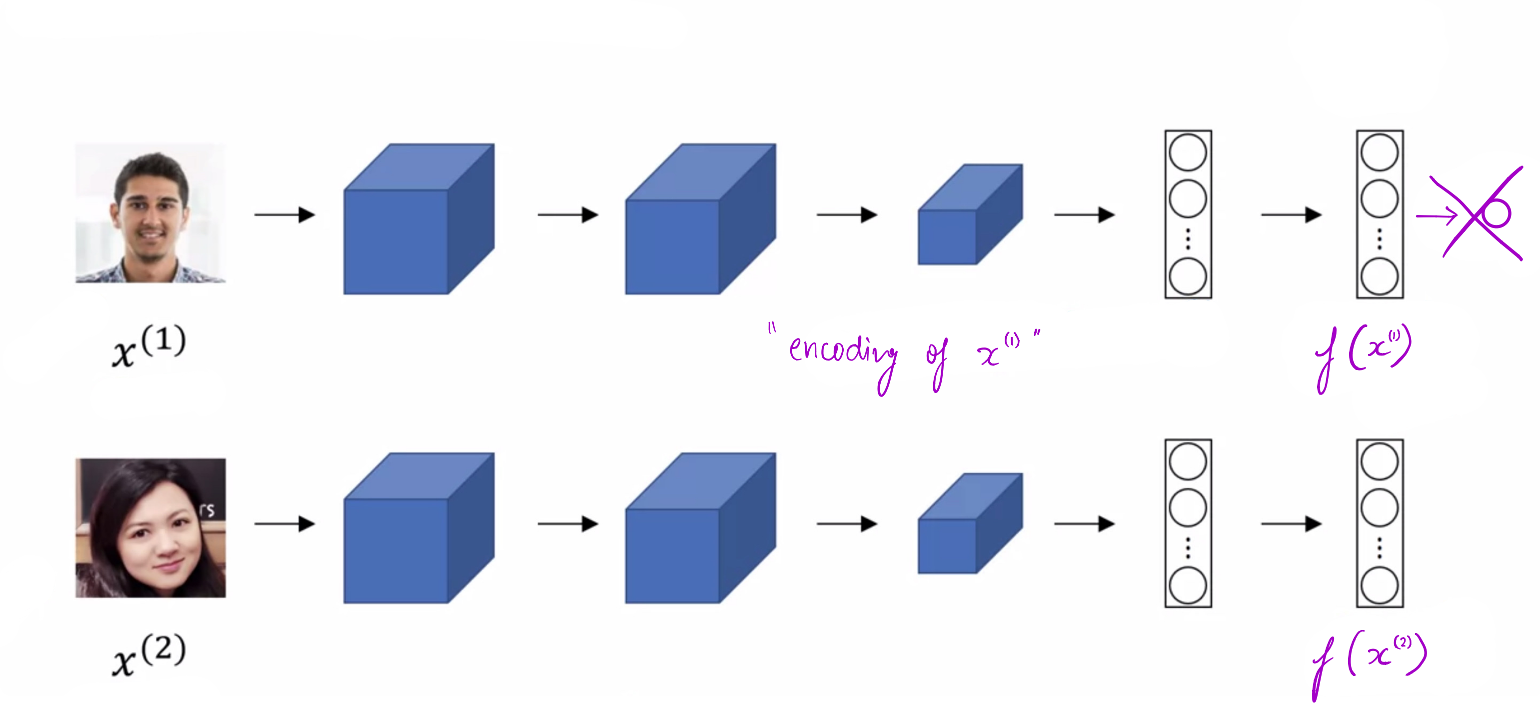
$d(\text{img } 1, \text{img } 2)$ = degree of difference between images

$$\left.\begin{array}{lll} \text{If } d(\text{img } 1, \text{img } 2) \leq \tau & & \text{"same"} \\ \qquad\qquad\qquad\qquad > \tau & & \text{"different"} \end{array}\right\} \text{verification}$$

some threshold

→ we try to predict if the two images have the same person by comparing it to others

Siamese Network



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

Parameters of NN define an encoding $f(x^{(i)})$

Learn parameters so that:

if $x^{(i)}, x^{(j)}$ are the same person, $\| f(x^{(i)}) - f(x^{(j)}) \|^2$ is small

if $x^{(i)}, x^{(j)}$ are different persons, $\| f(x^{(i)}) - f(x^{(j)}) \|^2$ is large

Triplet Loss

- comparing images to anchors
- we want -

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

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

margin

- Loss function -

given 3 images A, P, N :

$$\mathcal{L}(A, P, N) = \max\left(\| f(A) - f(P) \|^2 - \| f(A) - f(N) \|^2 + \alpha, 0\right)$$

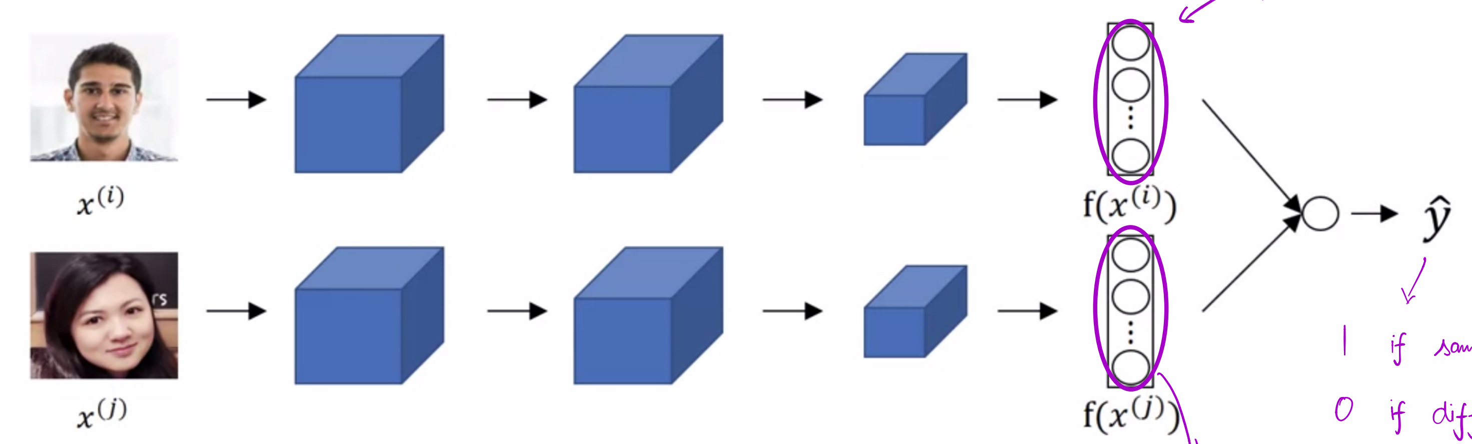
$$J = \sum_{i=1}^{m} \mathcal{L}(A^{(i)}, P^{(i)}, N^{(i)})$$

we want to make this zero or less than zero so that we train $f(A)$, $f(P)$, $f(N)$ to make the similarity function + margin equal to zero so that the distances are atleast α apart from each other

→ we try to choose $d(A,P)$ and $d(A,N)$ such that they are close and the learning algorithm works hard to push them apart

Face Verification & Binary Classification

input and test



1 if same

0 if different

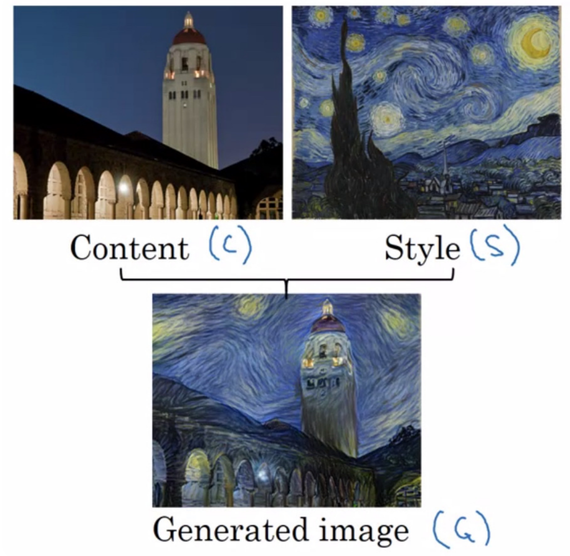
precompute

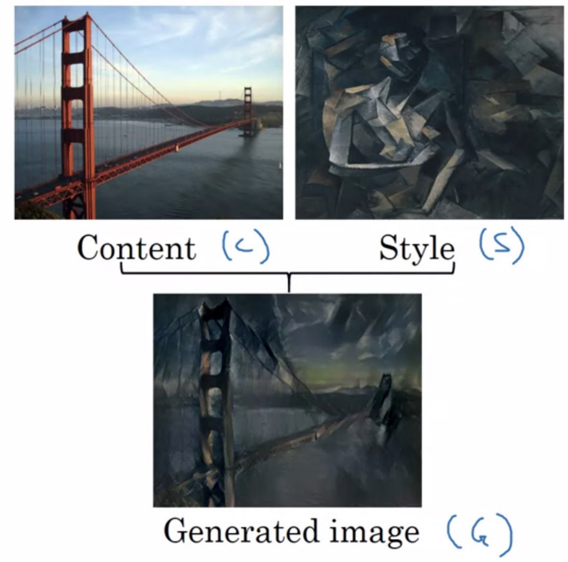
$$\hat{y} = \sigma\left(\sum_{k=1}^{128} \left| f(x^{(i)})_k - f(x^{(j)})_k \right| + b\right)$$

$$\frac{\left(f(x^{(i)})_k - f(x^{(j)})_k\right)^2}{f(x^{(i)})_k + f(x^{(j)})_k} \quad \chi^2 \text{ similarity}$$

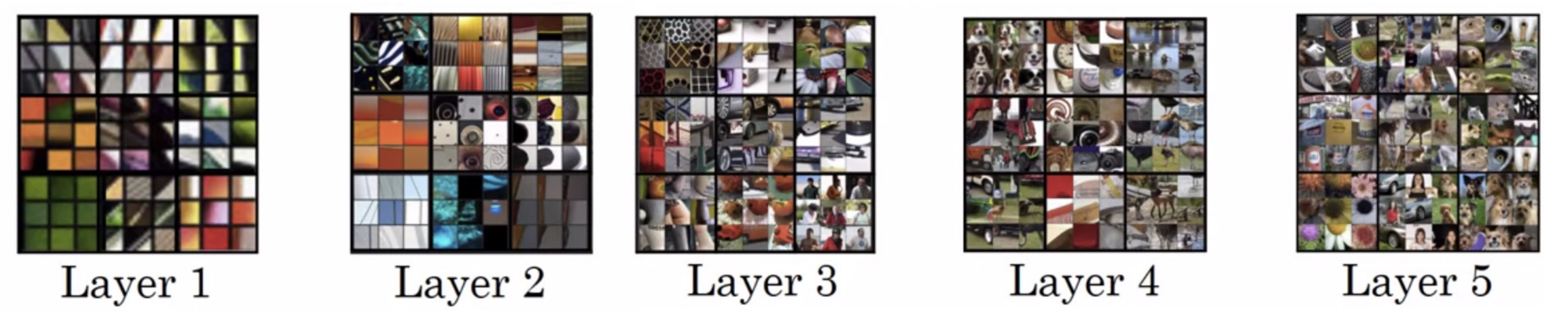
→ here we just compare 2 images and output the result

Neural Style Transfer





Visualizing Deep Layers



$\longrightarrow$

complexity

C S → G

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

→ initialize G randomly

$G: 100 \times 100 \times 3$

→ use gradient descent to minimize $J(G)$

$$G := G - \frac{\partial J(G)}{\partial G}$$

Content Cost Function

- → say you use layer l to compute content cost
- → use pre-trained ConvNet (VGG)
- → let $a^{[l](C)}$ and $a^{[l](G)}$ be the activation of layer l on the images
- → if $a^{[l](C)}$ and $a^{[l](G)}$ are similar, both images have similar content

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

learn so overall cost is low
to find an image G so the
hidden layer activations are
to the content image

Style Cost Function

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

Style Matrix :
"gram"

$$G^{[l](S)}_{kk'} = \sum_i \sum_j a^{[l](S)}_{ijk} \cdot a^{[l](S)}_{ijk'}$$

$$G^{[l](G)}_{kk'} = \sum_i \sum_j a^{[l](G)}_{ijk} \cdot a^{[l](G)}_{ijk'}$$

{ k and k' are correlated if $G_{kk'}$ is large
k and k' are un-correlated if $G_{kk'}$ is small

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

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

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

finally,

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