C4W4

Face verification vs face verification:-

Verification	recognition
input:- image + name/ID	Database of k person
match image with the named person image	input: image only.
(1:1) Problem	Input image is matched with k stored image
	(1:k) problem
	probability of having error is k times more than verification

One shot Learning:-

In practice to implement facial recognition we might not have a lot of data that we need for Deep learning.

Also if a new person join in a team its not feasible to retrain the whole network for one person.

Learning a "similarity" function:-

$d(img1, img2)$ = Degree of difference between images $\leq \tau$ "same"

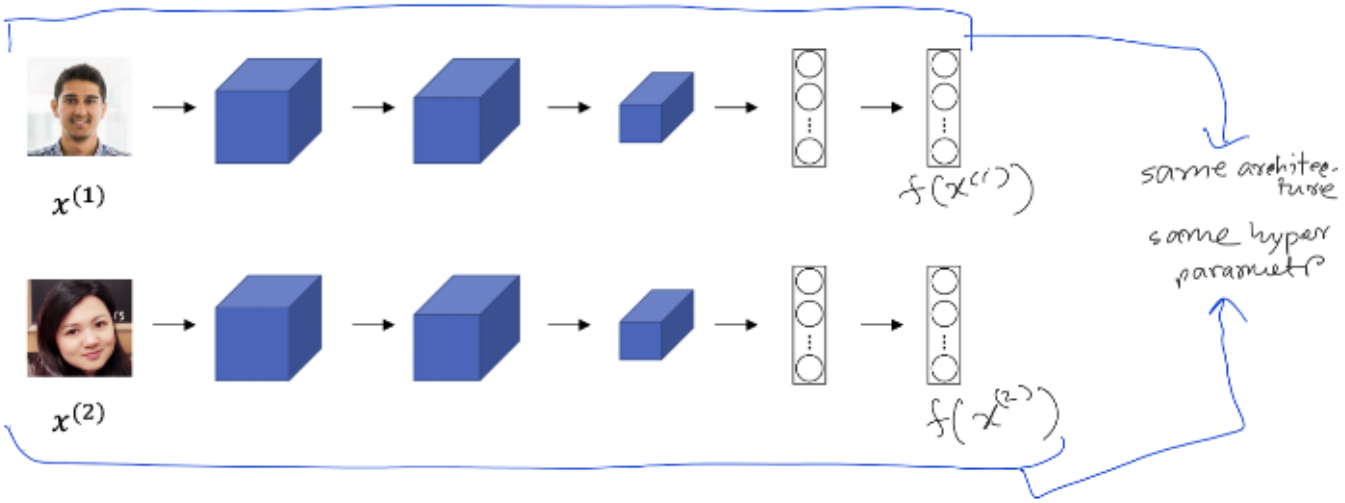
$> \tau$ "different"

we can use this technique in image recognition as well:-

```
for img2 in k:
    if d(img1, img2) ≤ τ:
```

person found in database

Siamese Network:-



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

Triplet loss:-

Anchor A | Positive P

Anchor A | Negative N

Learning Objective:-

we want:-

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

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

α ← margin

if $d(A,P) = d(A,N) = 0$ then this equation is solved. so

neural network might learn the objective by making everything $[f(A) = f(P) = f(N) = 0]$ by zero. To avoid this we need to modify the objective function by telling the neural network that $d(A,P) - d(A,N)$ should be less than zero. for that we add a parameter α.

Triplet loss function :-

It uses three images to find the loss function.

Given 3 images A, P, N :-

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

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

Training set :- 10k images of 1k person.

Choosing the triplets A, P, N

During training, if A, N, P are chosen randomly, $d(A,P) + \alpha \leq d(A,N)$ is easily satisfy.

We should choose a triplet that are hard to train :-

$$d(A,P) \approx d(A,N)$$

For example we should chose the triplet by group:

Gender

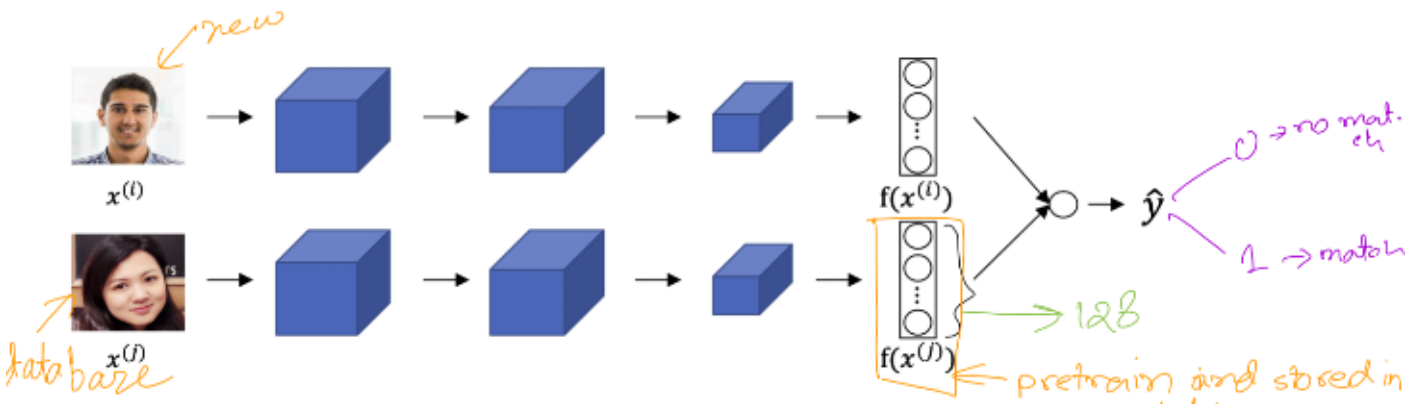
Gender

Age

color

ethnicity

Learning the similarity function (Alternative to triplet):-



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

← consider each point as feature

Element wise substracting

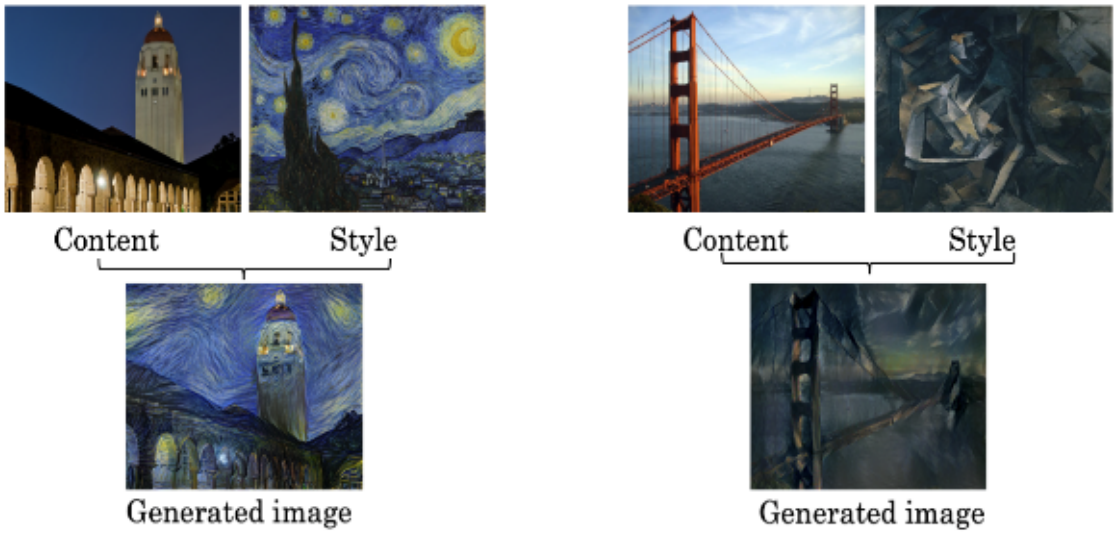
Alternative to

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

This is also called chi-squared χ^2

if a company have enabled facial recognition & a new employee join the company. Then the company doesn't need to train sample for the new employee. Instead they can use the pretrained version of the old employee & pass those to logistic function along with the new employee.

Neural style transfer:-



What are deep convnet are learning :-

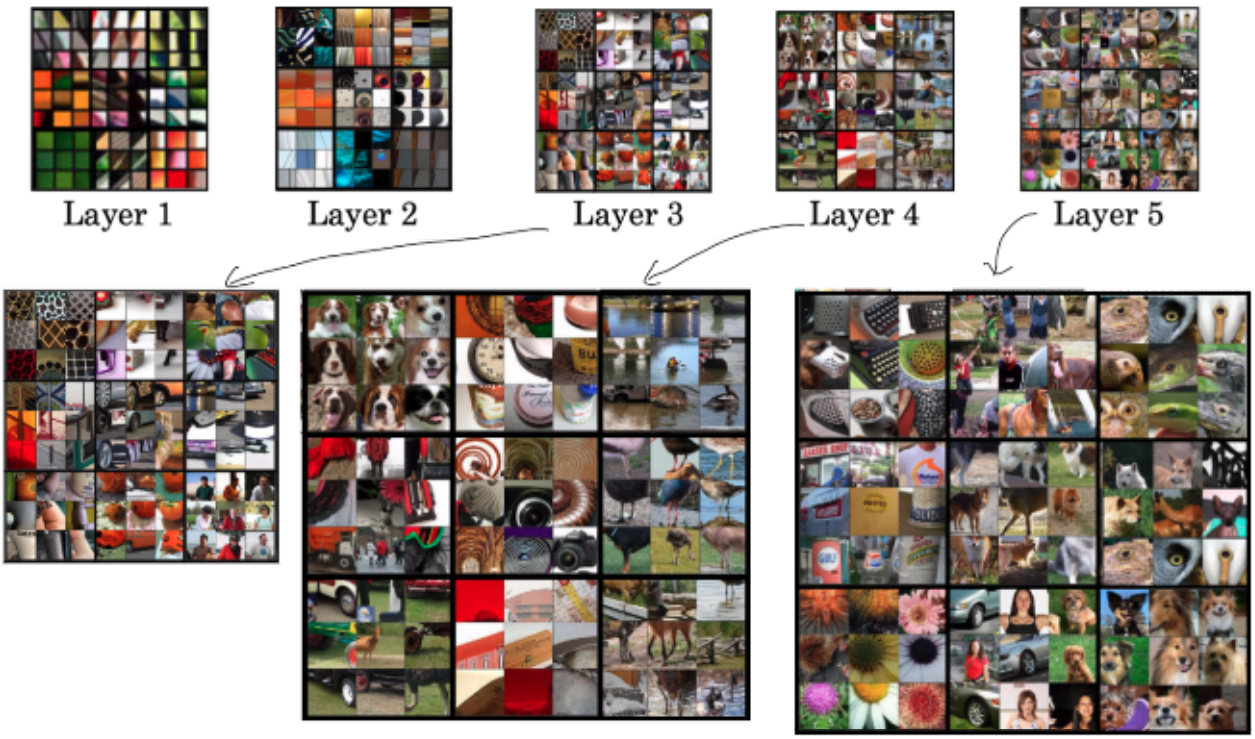
AlexNet

We want to visualize what the hidden unit in different layers are computing.

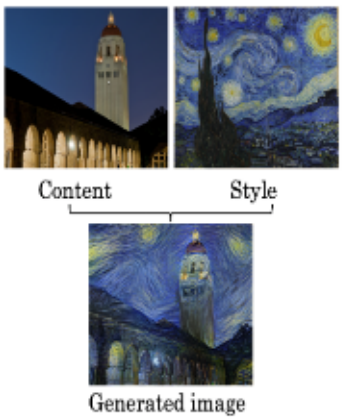
Pick one hidden unit from layer 1 & find 9 image patches that maximizes the unit's activation.

each box represents each individual neurons. & each of them tries to learn a different shape. As it is the first layers, the

neuron learns small patterns. In more deeper layers these learns more sophisticated pattern/shape.



Cost function for neural style:-



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

two hyperparameters are (α β) redundant. But the original author uses those.

Find the Generated image (G):-

1. Initiate G randomly

   G :- 100 × 100 × 3

2. Use gradient descent to minimize J(G)

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

Content cost function:-

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

→ Let, we use hidden layers l to compute content cost

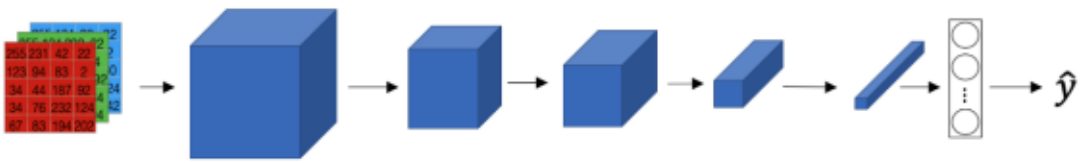
→ Use pretrained convNet (eg. VGGNet).

→ $a^{[l]C}$ & $a^{[l]G}$ be the activation layer l on the image

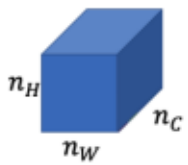
→ If $a^{[l]C}$ & $a^{[l]G}$ are similar then both image have similar content.

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

Style cost functions:-



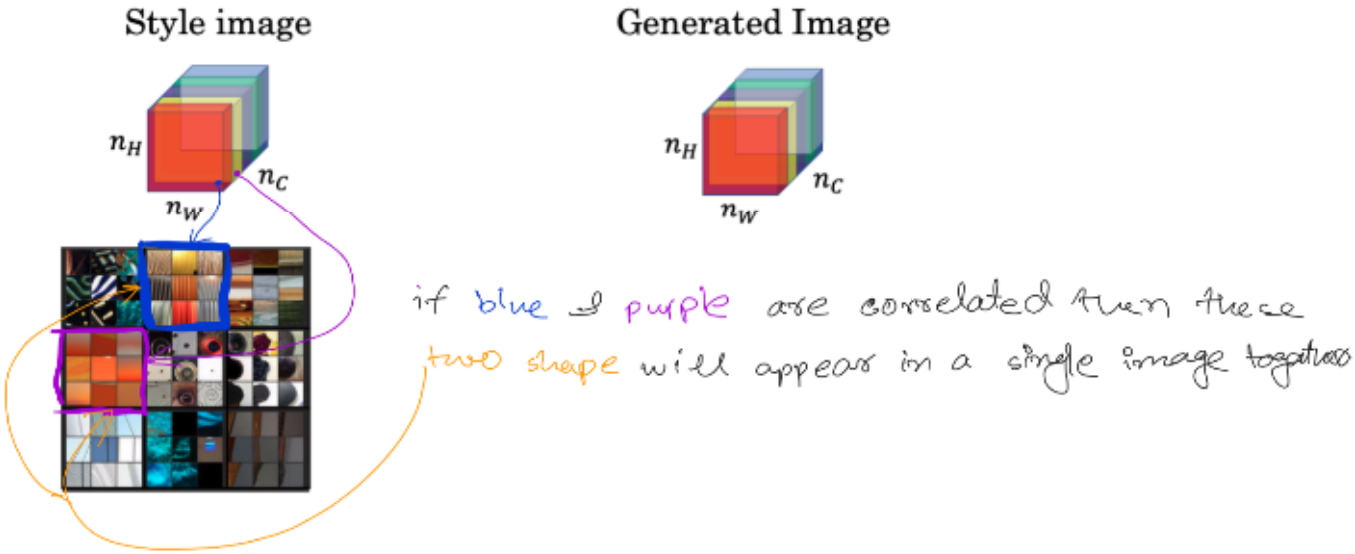
Say you are using layer l's activation to measure "style."
Define style as correlation between activations across channels.



How correlated are the activations across different channels?

Let's say we have 5 channel we want to find the correlation among the channel's activation function. If two channel's

neural's activations are highly correlated then there is high possibility that two pattern (recognized by two neurons) will be present in a single image.



style matrix :- (measure co-relation among channel's activation neurons)

Let, $a^{[l]}_{i,j,k}$ = activation at (i,j,k).
↑ ↑ ↑
H W n_c

$G^{[l]}$ is $n_c^{[l]} \times n_c^{[l]}$ → style matrix.
↳ square matrix

$G^{[l](s)}_{kk'}$ → correlation between channel k & k' of layer l.

$$\hookrightarrow \sum_{i}^{n_H^{[l]}} \sum_{j}^{n_W^{[l]}} a^{[l](s)}_{ijk}\, a^{[l](s)}_{ijk'}$$

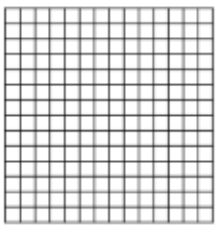
$$G^{[l](G)}_{kk'} = \sum_{i}^{n_H^{[l]}} \sum_{j}^{n_W^{[l]}} a^{[l](G)}_{ijk}\, a^{[l](G)}_{ijk'}$$

$$J^{[l]} \qquad \| G^{[l](S)} - G^{[l](G)} \|^2$$

$$J_{style}(S, G) = \frac{}{(2n_H n_W n_C)^2} \| G - G \|_F$$

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

Convnet in 1D or 2D:-



2D input image
14×14

*

2D filter
5×5

$14 \times 14 \times 3 \ * \ 5 \times 5 \times 3 \rightarrow 10 \times 10 \times 16$

$10 \times 10 \times 16 \ * \ 5 \times 5 \times 16 \rightarrow 6 \times 6 \times 32$

*

1	20	15	3	18	12	4	17

one dimensional data
Voltage of a ECG

1	3	10	3	1

filters

$14 \times 1 \ * \ 5 \times 1 \rightarrow 10 \times 16$

$10 \times 16 \ * \ 5 \times 16 \rightarrow 6 \times 32$

3D data:-

3D volume

*

3D filter

$14 \times 14 \times 14 \times 1^{n_c} \ *$

$5 \times 5 \times 5 \times 1$

$\rightarrow 10 \times 10 \times 10 \times 16 \rightarrow$ 16 filters

$10 \times 10 \times 10 \times 16 \ * \ 5 \times 5 \times 5 \times 16 \rightarrow$

$6 \times 6 \times 6 \times 32$