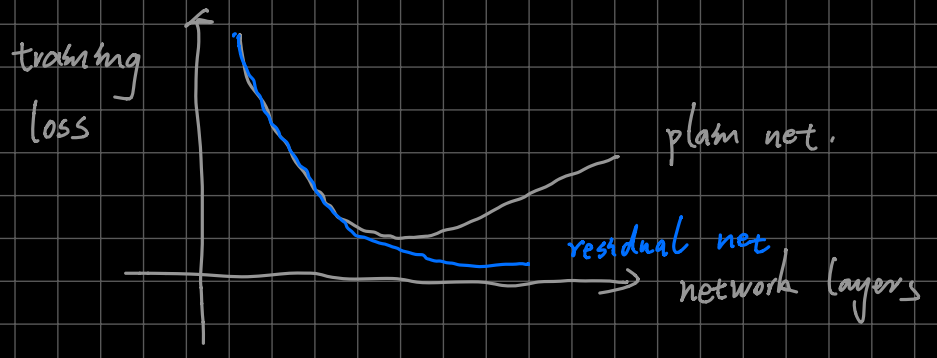
Plain networks: non-residual networks. ⇒ Theoretically, deeper (longer) the network, the training loss will be lower. But in fact, this is not true for plain network.



Famous Network Architecture

- Classic Networks
  - Lesnet
  - Alex net
  - VGG net
    - VGG16
    - VGG19
- Resnet ⇒ for deeper networks
- Inception-net ⇒ ① stacking inception blocks
  ② Using 1×1 convolutions to reduce the volume size before applying 3×3 / 5×5 convolution
  ③ Each inception block may be a combination of 1×1, 3×3, 5×5 convolutions and pooling

扩展一些细节知识点 CV: YOLO. Object Detection 两篇论

Resnet ⇒ The way to build a deep network

How: By stacking several Resnet blocks, each block has skip connections inside it.

Explanation: Resnet blocks with the shortcut make it very easy for one of the blocks to learn identity function. That's why you can stack on additional Resnet blocks with little risk of harming performance.

△ There is some evidence that the ease of learning an identity function accounts for Resnet's remarkable performance more than they help vanishing gradients do.

△ The skip connections compute a complex non-linear function of the input to pass to a deeper layer in the network.

Some facts about YOLO (You Only Look Once)

Why it is popular ?

High accuracy + Quick (run in real time)

Key thoughts ⇒ NMS (Non-Max Suppression)

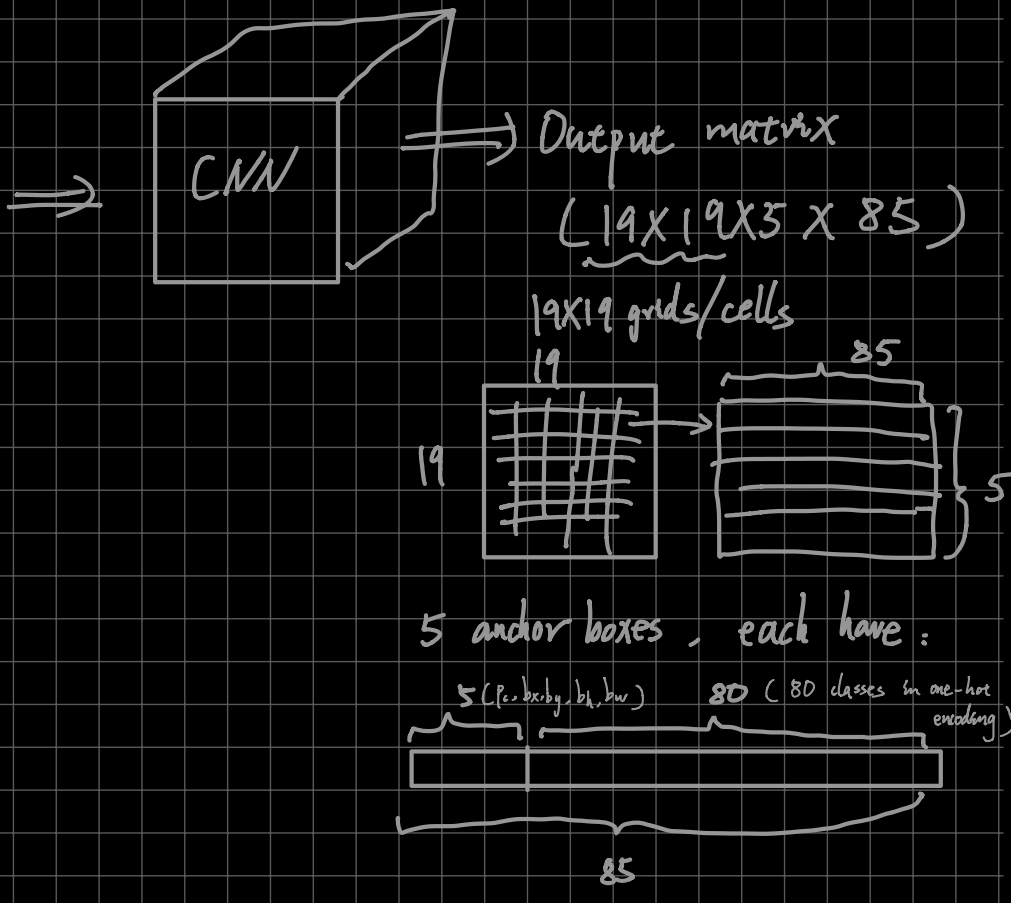
Step 1: Get rid of boxes with a low score

Step 2: Select the box that has the highest score.

Get rid of the boxes that : IOU ≥ threshold

Intersection over Union

Network structure :



Face Recognition

Different Problems:

Face verification

Input: image/name/ID

Output: Whether the input is that of the claimed person

(Sometimes we call it an one-to-one problem)

E.g: 笔记本电脑登录人脸识别

(只识别是否是你)

Face recognition

image

ID, if the input is any of the K persons in the database (this is the difference)

Not recognized, otherwise.

公司签到人脸识别

(识别所有公司的员工)

One Shot Learning Problem:

For most face recognition APPs, you can only get one training example (one short image of a person)

Difficulty: ① You have too little data to train the model

② Once you need to add a new person, you need to change model's parameters.

How to solve? Calculate the similarity of an image.

Intuition: ① Similarity 能较大地提取出一张图片的特征
相比 CNN，它提取的更彻底
⇒ solve ①
② Similarity 针对每张图片，因此无需改变参数 ⇒ solve ②

Siamese Network: 连体网络

What?

Parameters of NN is a good encoding $f(x^{(i)})$ of input images $x^{(i)}$.

How to train?

Learn parameters so that:

- If $x^{(i)}, x^{(j)}$ are the same person, $\left\| f(x^{(i)}) - f(x^{(j)}) \right\|^2$ is small.
- If $x^{(i)}, x^{(j)}$ are different person, $\left\| f(x^{(i)}) - f(x^{(j)}) \right\|^2$ is large.
  even slightly different

Loss: triplet loss

anchor image - positive image - negative image

Ensure: $\left\| f(A) - f(P) \right\|^2 + \alpha \leq \left\| f(A) - f(N) \right\|^2$

margin, to prevent the network get trivial solutions $\Rightarrow f(A) \equiv f(N) \equiv f(P) \equiv 0$

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

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

△ In training state, you need to ensure that there are lots of training examples (e.g. 10k of 1k person).
While in applying state, you just need at least one short of a person.

△ Need to choose triplets that're hard (→ $d(A,P) \approx d(A,N)$) to train on.
random choosing triplets are easily satisfied.
⇒ push the siamese network to gain a high accuracy.

△ One way to do face recognition is to perform binary classification (Input: a pair of images
Network: two similar sub-network
Output: 0 ⇒ not 1: verified

Neural Style Transfer

Cost function:

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

- $J(G)$: use GD to minimize
- Generated image
- C: content image
- Style image

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

- $a^{[l](C)}$: output of network lth layer using content image as input
- $a^{[l](G)}$: activation of network lth layer using generated image as input.
- Using pre-trained ⇒ VGG net

$$= \frac{1}{4 \times n_H \times n_W \times n_C} \cdot \sum_{\text{all entries}} \left(a^{(C)} - a^{(G)} \right)^2$$

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

- $\lambda^{[l]}$: decay factor

normalization term $\frac{1}{\left(2 n_H^{[l]} n_W^{[l]} n_C^{[l]}\right)^2}$ $\cdot \sum_k^{n_C^{[l]}} \sum_{k'}^{n_C^{[l]}} \left(G_{kk'}^{[l](S)} - G_{kk'}^{[l](G)} \right)^2$

- $G_{kk'}$: different channel's activation correlation

$$G_{kk'}^{[l](S)} = \sum_{i=1}^{n_H^{[l]}} \sum_{j=1}^{n_W^{[l]}} a_{ijk}^{[l](S)} \cdot a_{ijk'}^{[l](S)}$$

$$G_{kk'}^{[l](G)} = \sum_{i=1}^{n_H^{[l]}} \sum_{j=1}^{n_W^{[l]}} a_{ijk}^{[l](G)} \cdot a_{ijk'}^{[l](G)}$$

Gram matrix:

Compares how similar two matrix are.

If they are highly similar, $G_{kk'}$ will be large.

Gram Matrix:

$G_{gram} = A_{unrolled} * A^T_{unrolled}$, shape: (nc, nc)

$A_{unrolled}$: $A = (m, n_H, n_W, n_C) \xrightarrow{\text{tf.transpose}} (m, n_C, n_H, n_W) \xrightarrow{\text{tf.reshape, unrolled}} (m, n_C, n_H \times n_W)$

$G_{gram}(i,j)$: how similar the activation of channel i with the activation of channel j

$G_{gram}(i,i)$: (对角线上元素) how active a channel i is. (large $\Leftrightarrow$ the channel i is highly activated by input image)

△ In neural style transfer, we optimize the cost function $J(G)$ to update generated image's {pixel values. (not the network parameters)}

△ We apply transfer learning in NST (use VGG19 as pre-trained network)

△ The layers chosen in computing $J_{content}$ and J_{style} are middle layers (not shallow layers, not too deep layers)