卷积神经网络

一. 卷积神经网络基础 | Parameter sharing 参数共享. Sparsity of connection 稀疏连接

1. 边缘检测

Sobel filter. $\begin{bmatrix}1&0&-1\\2&0&-2\\1&0&-1\end{bmatrix}$ Schoss filter $\begin{bmatrix}3&0&-3\\10&0&-10\\3&0&-3\end{bmatrix}$

2. Padding.

Valid. Conv $nn * ff = (n-f+1)(n-f+1)$

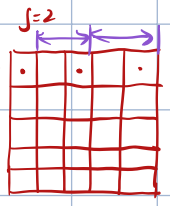
Same Conv. $(n+2p)(n+2p) * ff = n*n$ $p = \frac{f-1}{2}$

$\Downarrow (n+2p)^2 * f^2 = (n+2p-f+1)^2$.

Stride Conv.

$\left\lfloor \frac{(n+2p)-f}{s} + 1 \right\rfloor^2$. 下取整.

↓并头第1个



3. ① Conv.

$f^{[l]}$ = filter size

$p^{[l]}$ = padding

$s^{[l]}$ = stride

$n_c^{[l]}$ = num of filters

> Input: $n_H^{[l-1]} * n_W^{[l-1]} * n_c^{[l-1]}$

+ filter: $f^{[l]} * f^{[l]} * n_c^{[l-1]}$

> Output: $n_H^{[l]} * n_W^{[l]} * n_c^{[l]}$

$$n_{H/W}^{[l]} = \left\lfloor \frac{n_{H/W}^{[l-1]} + 2p^{[l]} - f^{[l]}}{s^{[l]}} + 1 \right\rfloor$$

② Pooling.

Max Pooling 无需学习的参数

Average Pooling.

S. f. $\begin{cases}\text{max}, \text{激活在前后无差别}\\ \text{avg}, \text{先激活后池化}\end{cases}$

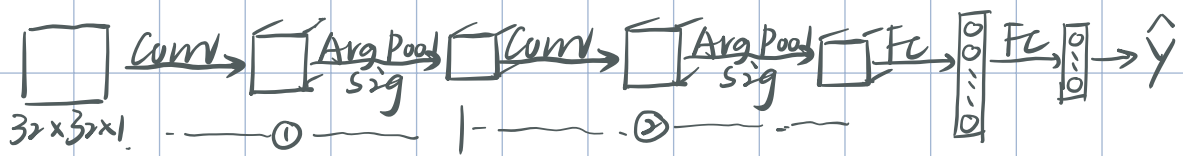
↓建议. Conv → Activation → Pooling

③ Fully connected. FC.

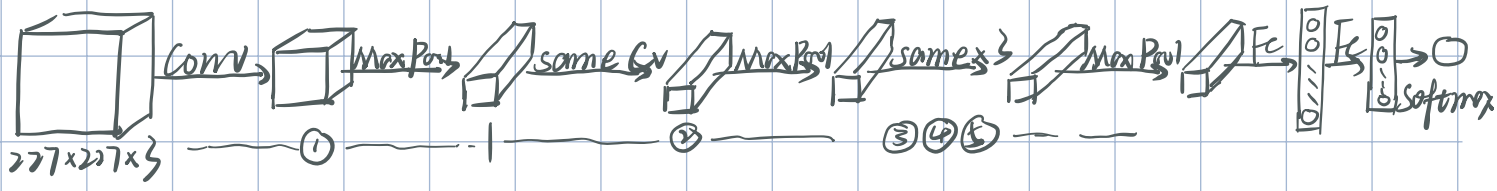
二. Deep CNN

1. 经典网络

> LeNet-5. 手写字符识别.

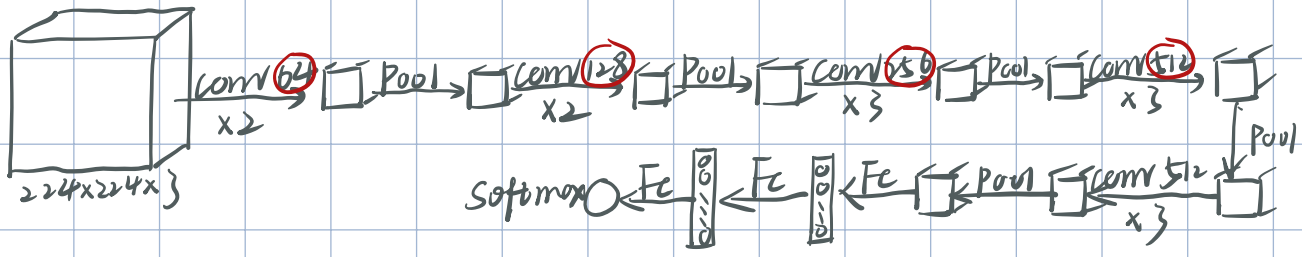


> AlexNet.



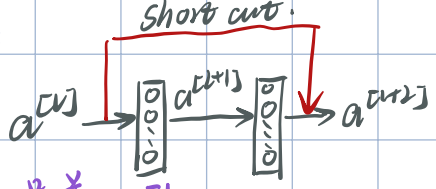
> VGG-16 Fc+Conv=16

Conv. 3×3 filter, S=1. P=same. MAX Pool = 2×2, S=2.



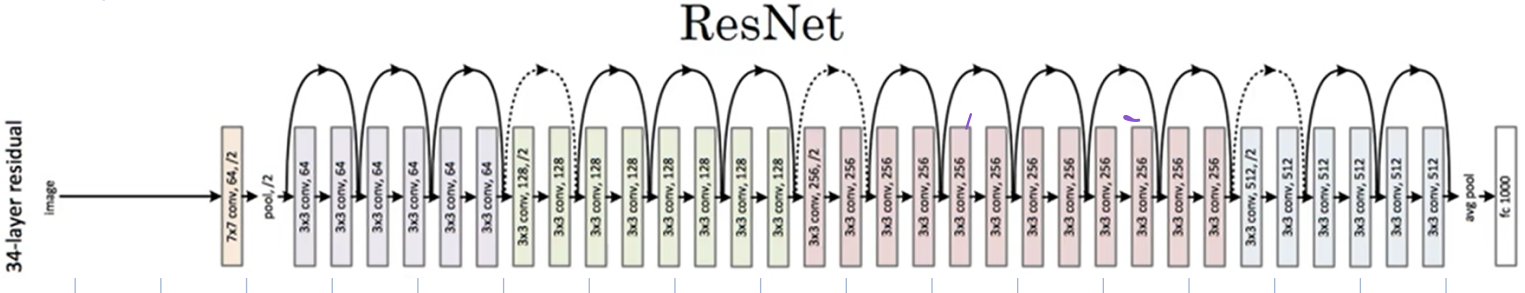
2. ResNets 残差网络

Residual block.



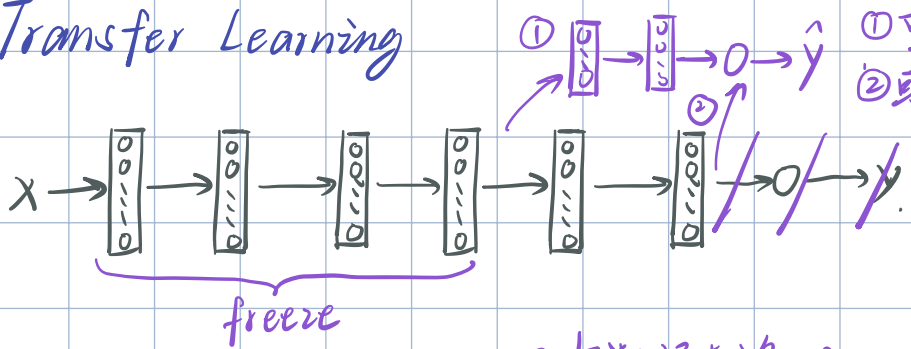
$a^{[l+2]} = g(z^{[l+2]} + a^{[l]})$

- 防止过深的层数导致的误差上升.
- 便于训练学习



3. Inception (GoogLeNet)
"Go Deeper"

4. Transfer Learning



① 重新设置 layer.
② 或用权重初始化.

已有数据越多, freeze 越少

5. Data Augment. 数据增强

1) mirror. 镜像
random cropping 随机裁剪
rotation
shearing
local warping 扭

2) color shift [PCA Color Augment]

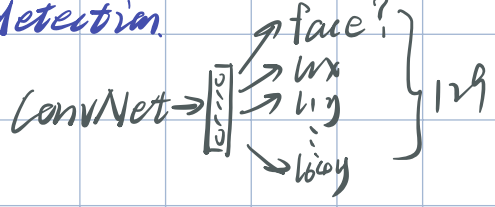
三. Object Detection

classification → localization → detection

1. Localization 定位.

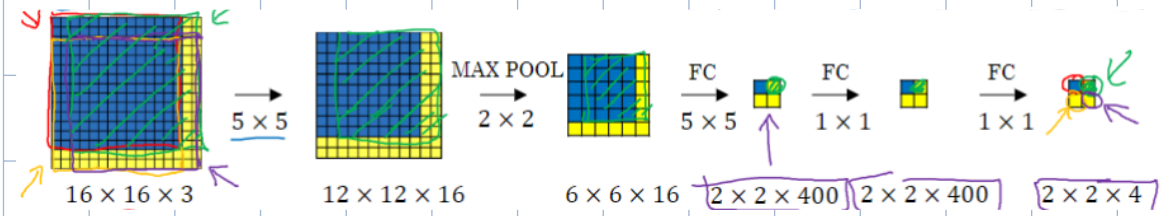
$$y=\begin{bmatrix} P_c \\ b_x\ b_y \\ b_h\ b_w \\ c_1\ c_2\ c_3 \end{bmatrix} \begin{matrix} \text{是否有} \\ \\ \\ \text{种类} \end{matrix}$$

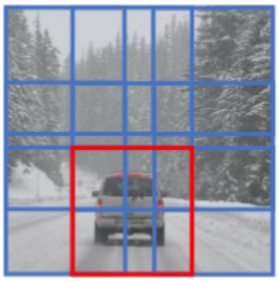
2. Landmark detection 特征点检测



3 detection

滑动窗口. Sliding Window



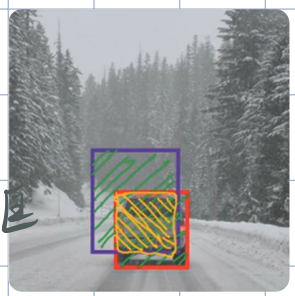


用卷积代替全连接层

4. 交并比 Intersection over Union

△ 评价检测质量

$IoU = \frac{size(I)}{size(U)}$ $IoU \geq 0.5$. ✓



5. 非最大抑制 Non-max suppression

△ 防止一个对象检测多次(对象周围的很多窗口都可能检测出边框)

1) 去除 $P_c \leq 0.6$

2) while 有剩余框：选择 max P_c.

去除对该框 $IoU \geq 0.5$ 的其他框.

△ 每类对象单独做非最大抑制

6. 锚框 Anchor Box

△ 一格子检测多对象

每个 cell grid 添加多个锚框，每个对象分配给 IoU 最大的锚框.

○ 用人工方法手动制定锚框的大小

7. YOLO

图片分成 9×9 & 16×16，每个格子 anchor box，最后 non-max suppression

8. RPN (R-CNN)

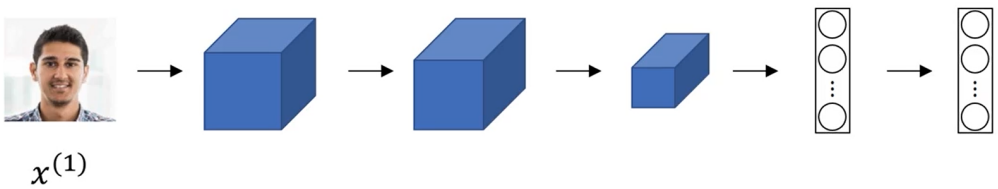
先语义分割，再对区域做目标识别.

Fast R-CNN

四. Face Detection

1. One Shot.

> Siamese network 输入两张人脸，判断其相似度.



$d(img_1, img_2)$ 相似度

$= \|f(img_1) - f(img_2)\|_2^2$ 全连接层输出向量

训练方法：

1> Tripet Loss

Anchor < Positive / Negative

三张图为一组.

A：原图，P：同人的不同照，N：不同人的照片.

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

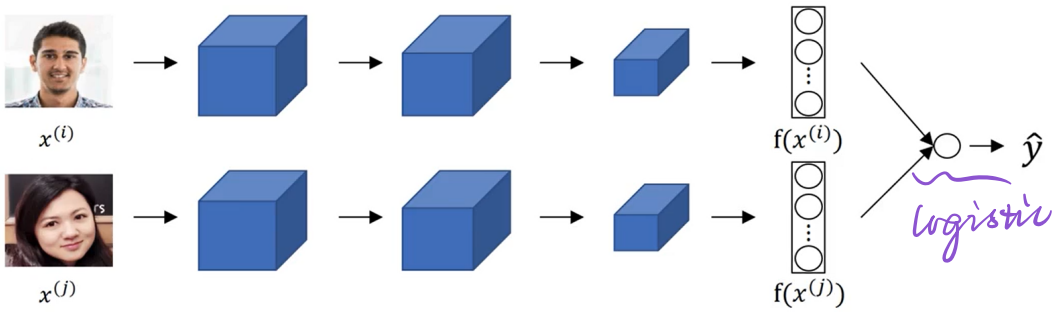
$J = \sum L(A^{(i)}, P^{(i)}, N^{(i)})$

↓期望其<0，若>0，则有损失

寻找 $d(A,P) \approx d(A,N)$ 的进行训练，损失较大，网络学习更多.

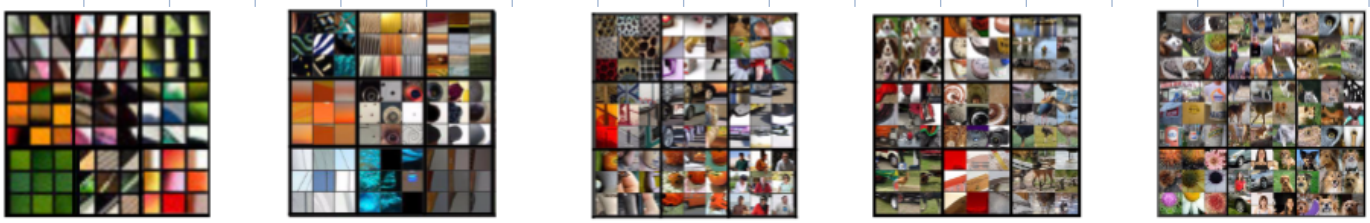
若 $d(A,P) <= d(A,N)$ 则无损失，学不到知识.

2). Binary classfication



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

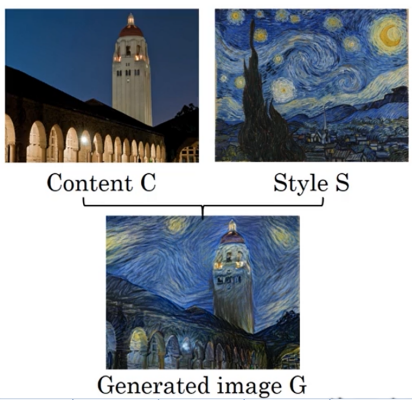
五. 神经风格转换. Neural Style Transfer



Layer 1 Layer 2 Layer 3 Layer 4 Layer 5

原始图像在每一层最大化激活的单元块(图片块)

1. Cost Function. 代价函数



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

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

C、S、G 都进行卷积.

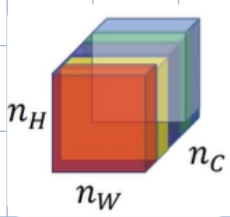
1.1 Content

找到卷积中间层 l 用来代表图像内容. Layer 1. 过于相似. Layer-1 代表逻辑.

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

a 代表 l 层的激活.

1.2 Style



← Layer l. 某点定位 (i, j, k) $H \times W \times C$

定义风格矩阵 $G^{[l]}$. $n_C^{[l]} * n_C^{[l]}$, 两两通道间 kk'

↓代表风格.

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

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

↓

l层损失 $J_{style}^{[l]} = \|G^{[l][S]} - G^{[l][G]}\|_F^2 = \frac{1}{(2n_H^{[l]} n_W^{[l]} n_C^{[l]})^2} \sum_k \sum_{k'} (G_{kk'}^{[l][S]} - G_{kk'}^{[l][G]})^2$

↓

总体损失 $J_{style} = \sum_l \lambda^{[l]} J_{style}^{[l]}$