1. 人脸识别

1.1 Verification

- Input：图片，名字/ID；
- Output：输入的图片是否存在对应的人；
- 一对一的问题。

1.2 Recognition

- 拥有K个人的数据库；
- 输入一个人脸图片；
- 如果该人脸属于该数据库，则输出对应的ID。

人脸识别问题的难度高于人脸验证。人脸验证问题中，如果有99%的精确度，那么这个系统已经有非常高的精度。但是如果应用于有K个人的人脸识别系统，那么这个系统的犯错误的机会就会变成K倍。

2. One-shot learning

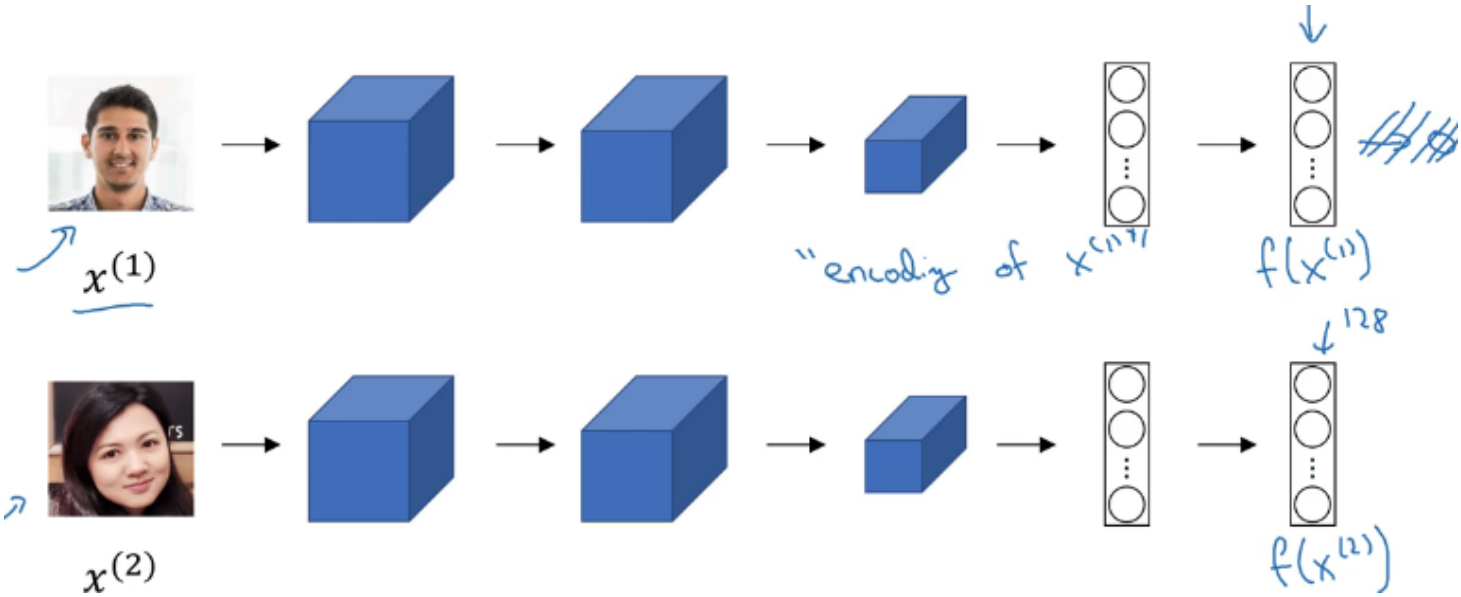
只有单个样本不足以训练一个稳健的卷积神经网络，而且如果新加入成员时还需要重新训练网络。

为了能够实现one-shot learning，需要让神经网络学习similarity函数：

- Input：两张图片
- Output：$d(img_1, img_2)$，即两张图片的差异度
  - 如果$d(img_1, img_2) \leq \tau$，则输出"same"
  - 如果$d(img_1, img_2) > \tau$，则输出"different"

对于人脸识别系统，只需将输入的图片和数据库的中的图片两两比较，就可以解决one-shot learning的问题。如果有新加入的成员，只需将图片添加到数据库即可。

3. Siamese network



对于一个卷积神经网络，我们去掉softmax层，把最后一层的输出向量作为编码。similarity函数表示为两个图片的编码之差的范数：

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

那么也就是说：

- 神经网络的参数定义了图片的编码$f(x)$；
- 学习网络的参数
  - 如果$x^{(i)}$和$x^{(j)}$是同一个人，那么$d(x^{(i)}, x^2)$很小
  - 如果$x^{(i)}$和$x^{(j)}$不是同一个人，那么$d(x^{(i)}, x^2)$很大

4. Triplet loss

4.1 Learning objective

为了使用triplet损失函数，需要比较成对的图像：

- Anchor (A)：目标图片

- Positive (P)：和anchor属于同一个人的图片
- Negative (N)：和anchor不属于不同一个人的图片

对于anchor和positive，我们希望二者的编码差异小一点；对于anchor和negative，我们希望编码的差异大一些。这个过程可以描述为：

$$d(A,P) = ||f(A) - f(P)||^2 \leq ||f(A) - f(N)||^2 = d(A,N)$$

也就是：

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

这个公式存在一个问题，如果$f(A)|f(P)|f(N)$都为0向量或者彼此相等时，总能满足这个方程。因此我们对上式进行修改，是两者的差距小于一个较小的负数：

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

一般将α写成$+\alpha$，称为"margin"：

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

4.2 Triplet loss function

- Input: anchor|positive|negative
- Loss: $L(A,P,N) = max(||f(A) - f(P)||^2 - ||f(A) - f(N)||^2 + \alpha, 0)$

整个网络的loss为：$J = \sum_{i=1}^{m} L(A^{(i)}, P^{(i)}, N^{(i)})$0

假设有一个10000张图片的training set，里面包含1000个人。我们需要从这10000张图片中抽取图片生成(A,P,N)三元组来训练算法，并在triplet损失函数上进行梯度下降。

为了训练我们的网络，必须拥有achor和positive，所以只要求了每个人都必须有多张图片。如果仅有一张图片，那么将无法训练网络。

4.3 Choosing the triplets (A,P,N)

在训练的过程中，随机选择图片组成三元组，是很容易满足$d(A,P) + \alpha \leq d(A,N)$这一条件。

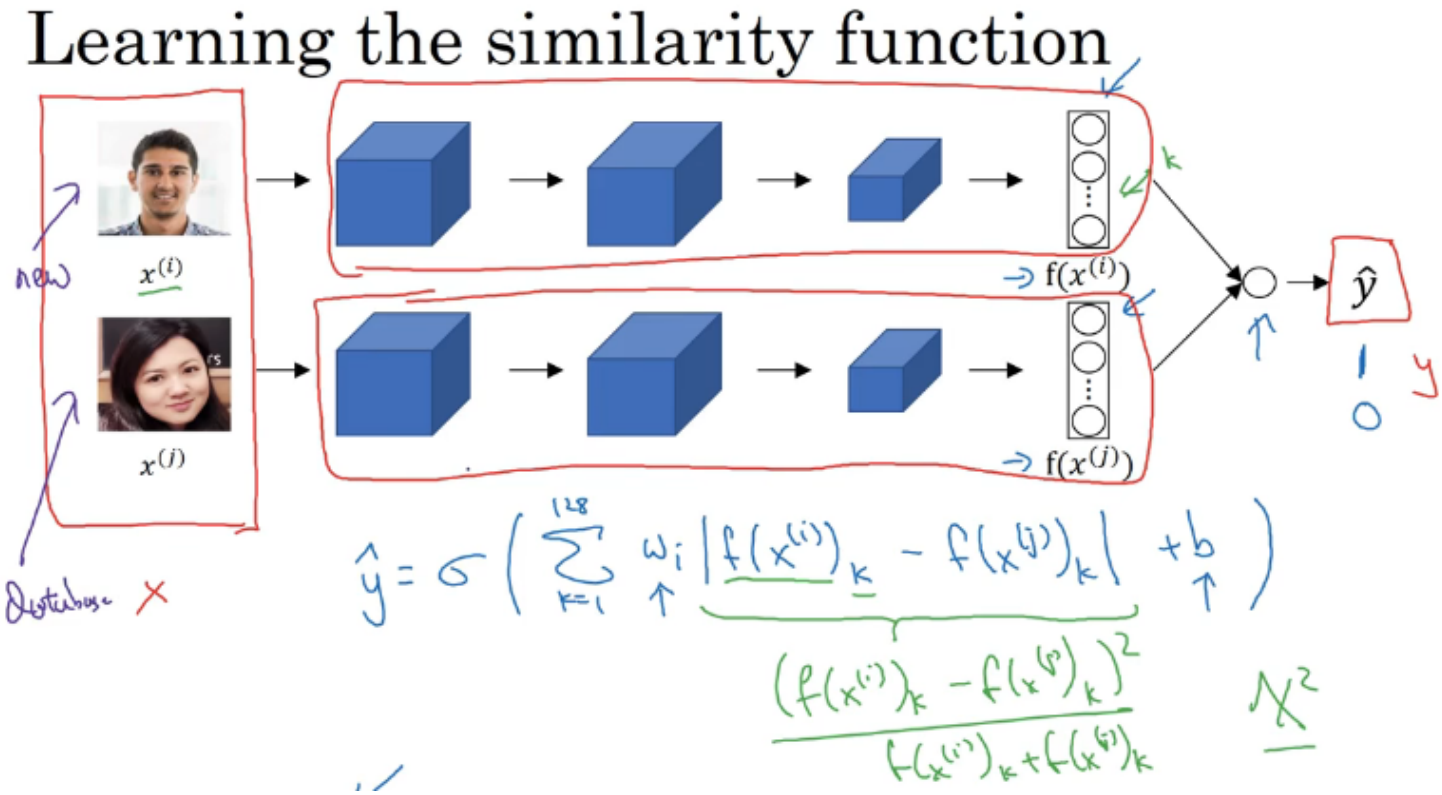
所以，为了更好地训练网络，需要选择具有难度的三元组，即满足$d(A,P) \approx d(A,N)$

- 算法将会努力使$d(A,N)$变大，或者使得$d(A,P) + \alpha$变小，使两者之间至少有一个α的间隔；
- 可以增加学习算法的计算效率。

最终通过训练，使得网络对于同一个人的图片，编码的距离很小；对于不同人的图片，编码的距离就很大。

5. 脸部识别和二分类

除了了用triplet损失函数来学习人脸识别卷积网络参数的方法外，还有其他方式。可以将人脸识别问题利用Siamese网络转化为二分类的问题。



对两张图片应用Siamese网络，各自得到128维的编码，将这两个编码输入到logistic回归单元中进行预测。如果是相同的人，那么$\hat{y} = 1$，否则$\hat{y} = 0$。

对于最后的sigmoid函数，进行如下的运算：

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

其中$f(x^{(i)})$表示第i个图片的编码，k表示编码向量中的第k个元素。

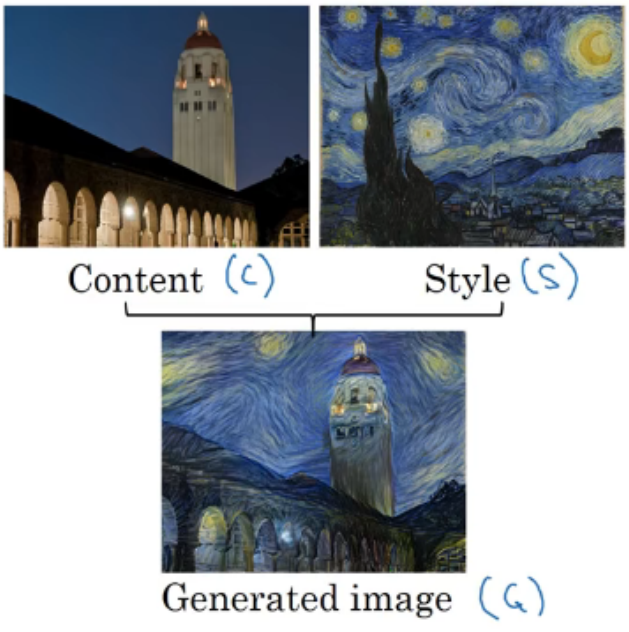
可以使用χ方公式代替一阶范式：

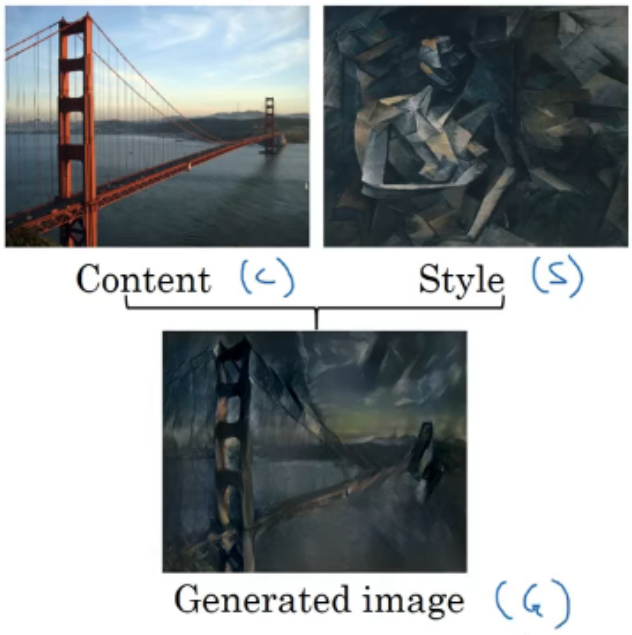
$$\frac{(f(x^{(i)})_k - f(x^{(j)})_k)^2}{f(x^{(i)})_k + f(x^{(j)})_k}$$

可以将已在数据库当中的员工的人脸图片的编码预先计算好后保存，然后与新图片的编码一起输入到logistic回归单元中进行预测，这样可以提高效率。

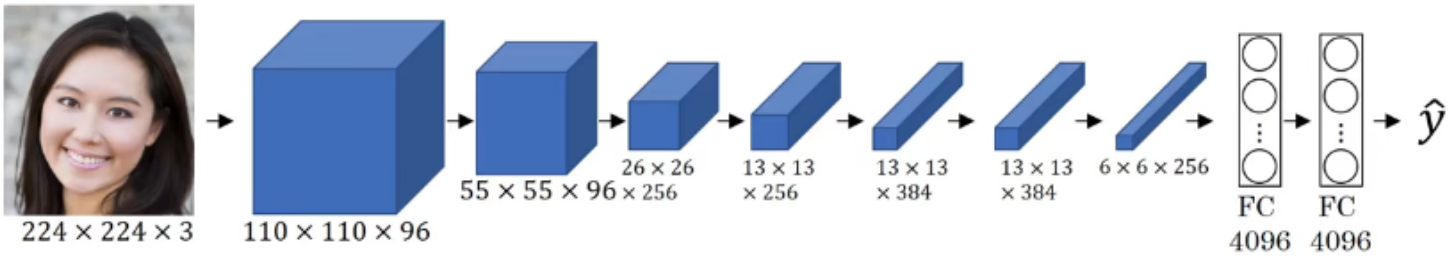
6. 神经风格迁移

为了实现神经风格迁移，需要从不同的卷积神经网络中提取特征。



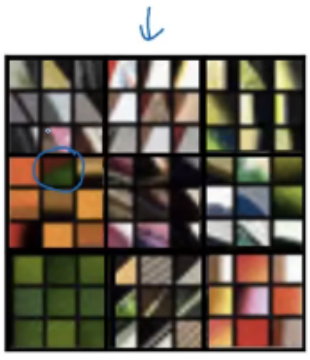


7. 深度卷积神经网络可视化



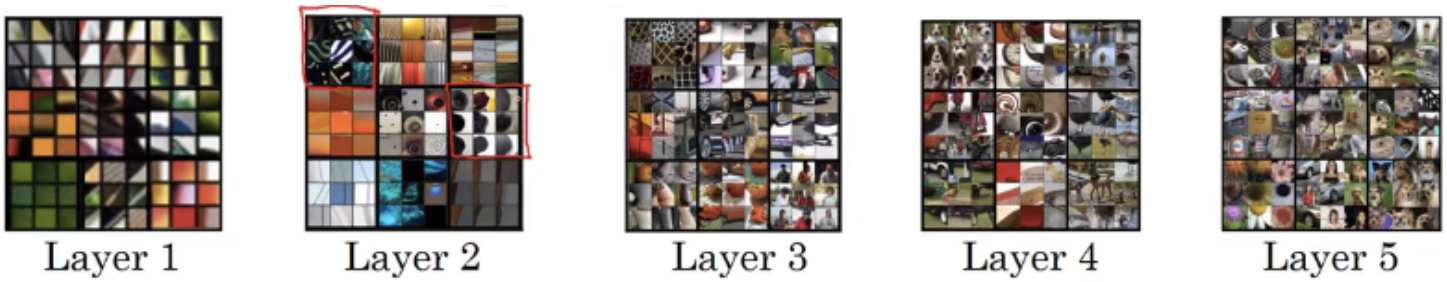
Pick a unit in layer 1. Find the nine image patches that maximize the unit's activation.

Repeat for other units.



对于每一层，我们执行如下的操作：

- 在当前层挑选一个unit；
- 遍历训练集，找到最大化地激活该单元的图片或者图片块；
- 对该层的其他单元执行操作。

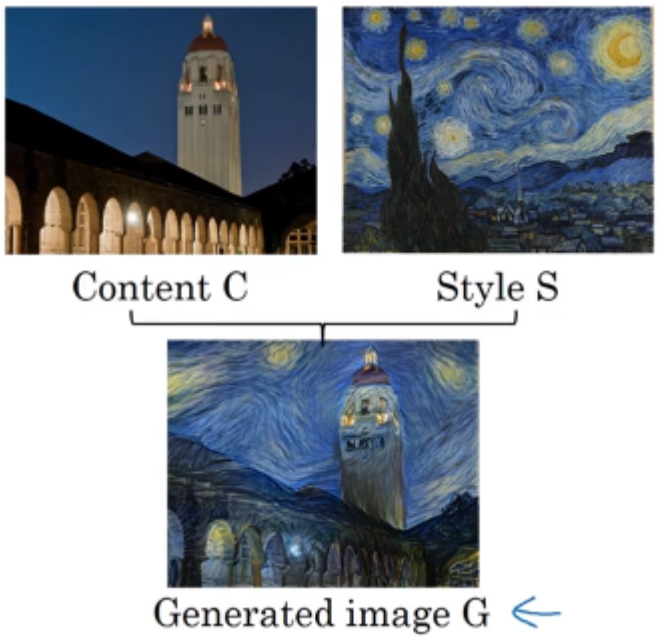


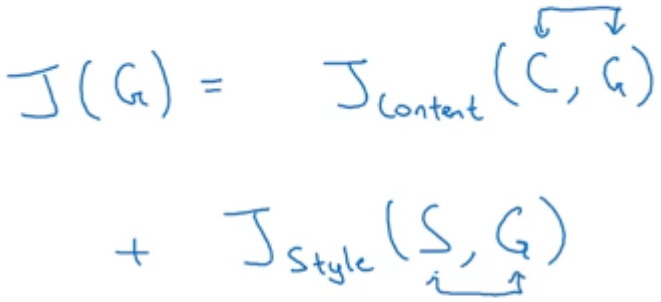
随着网络深度的增加，能够学习到更加复杂的特征，能认识到更加复杂的事物。

8. Cost function

为了实现神经风格迁移，我们需要为生成的图片定义一个代价函数。通过最小化代价函数，可以生成任何想要的图像。

8.1 定义cost function





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

- $J_{content}(C, G)$表示生成图片G和内容图片C内容的相似度；
- $J_{style}(S, G)$表示生成图片G和风格图片S的内容的相似度。

8.2 执行过程

- 随机初始化生成图片G，大小为$100 \times 100 \times 3$；
- 使用梯度下降最小化$J(G)$，即$G = G - \frac{\partial}{\partial G}J(G)$；
- 通过不断的训练，可以由初始噪声图片转化为最终的风格迁移图片。

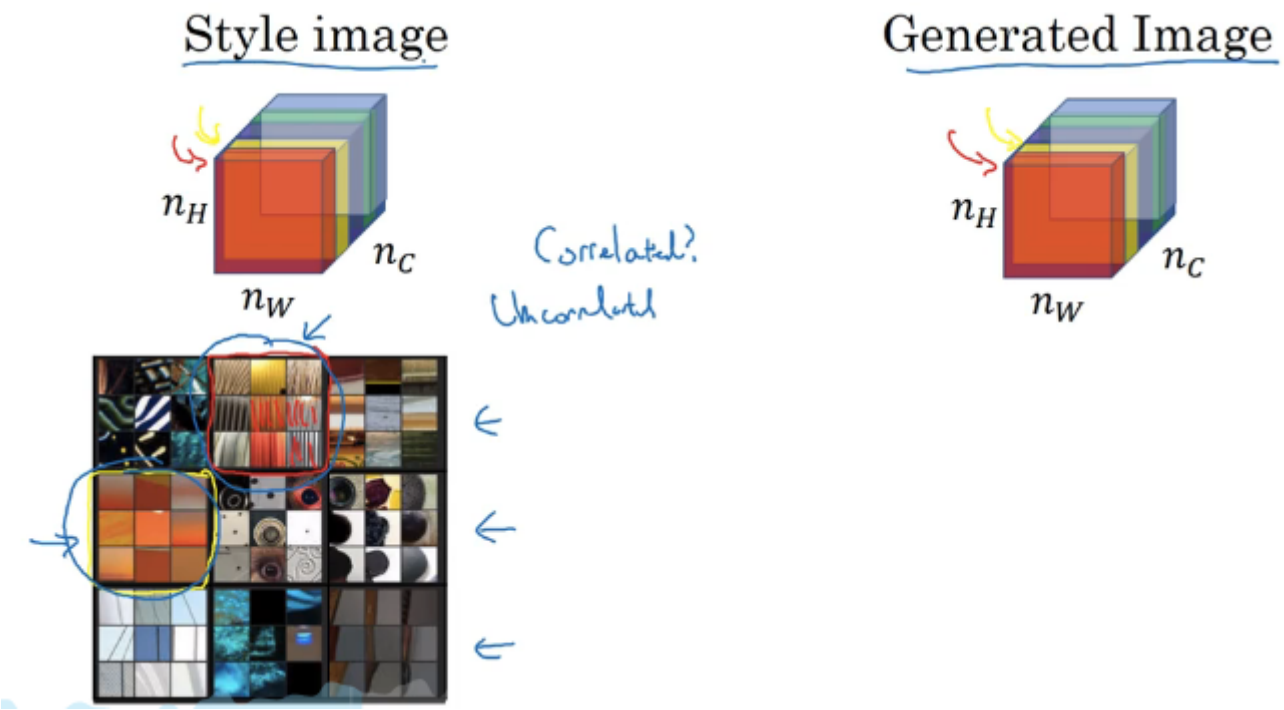
9. Content cost function

- 使用隐藏层l来计算内容代价。如果l太浅，那么生成图片G会非常接近内容图片C；如果l太深，则会使生成图片G会产生内容图片C中拥有的物体。所以一般会选择中间层。
- 使用一个预训练的卷积模型（比如VGG）。
- $a^{[l](C)}$和$a^{[l](G)}$分别代表内容图片C和生成图片G的第l层的激活值，如果它们的值相似，那么两张图片就有相似的内容。

内容代价函数的定义如下：

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

10. Style cost function



定义"style"表示某种中间层l的各个通道之间的相关性。每个通道代表学校到的不同特征，相关性则表示图片中含有某种特征的可能性大小。我们将相关系数用于风格图片S和生成图片G的对应通道上，便可以度量两个图片之间的对应特征的相似度。

- 我们设$a^{[i]}_{i,j,k}$是(i,j,k)位置的激活值，其中i、j、k分别表示高、宽、通道。

- $G^{[l]}$是一个$n_c^l \times n_c^l$大小的矩阵：

$$G_{kk'}^{[l](S)} = \sum_{i=1}^{n_h^{[l]}} \sum_{j=1}^{n_w^{[l]}} a_{i,j,k}^{[l](S)} a_{i,j,k'}^{[l](S)}$$

$$G_{kk'}^{[l](G)} = \sum_{i=1}^{n_h^{[l]}} \sum_{j=1}^{n_w^{[l]}} a_{i,j,k}^{[l](G)} a_{i,j,k'}^{[l](G)}$$

  这个矩阵在线性代数中被称为Gram矩阵，这里被称为风格矩阵。
- 在这基础上我们定义代价函数：

$$J_{style}^{[l]}(S,G) = \frac{1}{2n_h^{[l]} n_w^{[l]} n_c^{[l]}} ||G^{[l](S)} - G^{[l](G)}||_F^2 = \frac{1}{2n_h^{[l]} n_w^{[l]} n_c^{[l]}} \sum_k \sum_{k'} (G_{kk'}^{[l](S)} - G_{kk'}^{[l](G)})^2$$

  内容代价函数和风格代价函数前的归一化系数可以加也可以不加。
- 对各层使用风格代价函数，可以使结果变得更好：

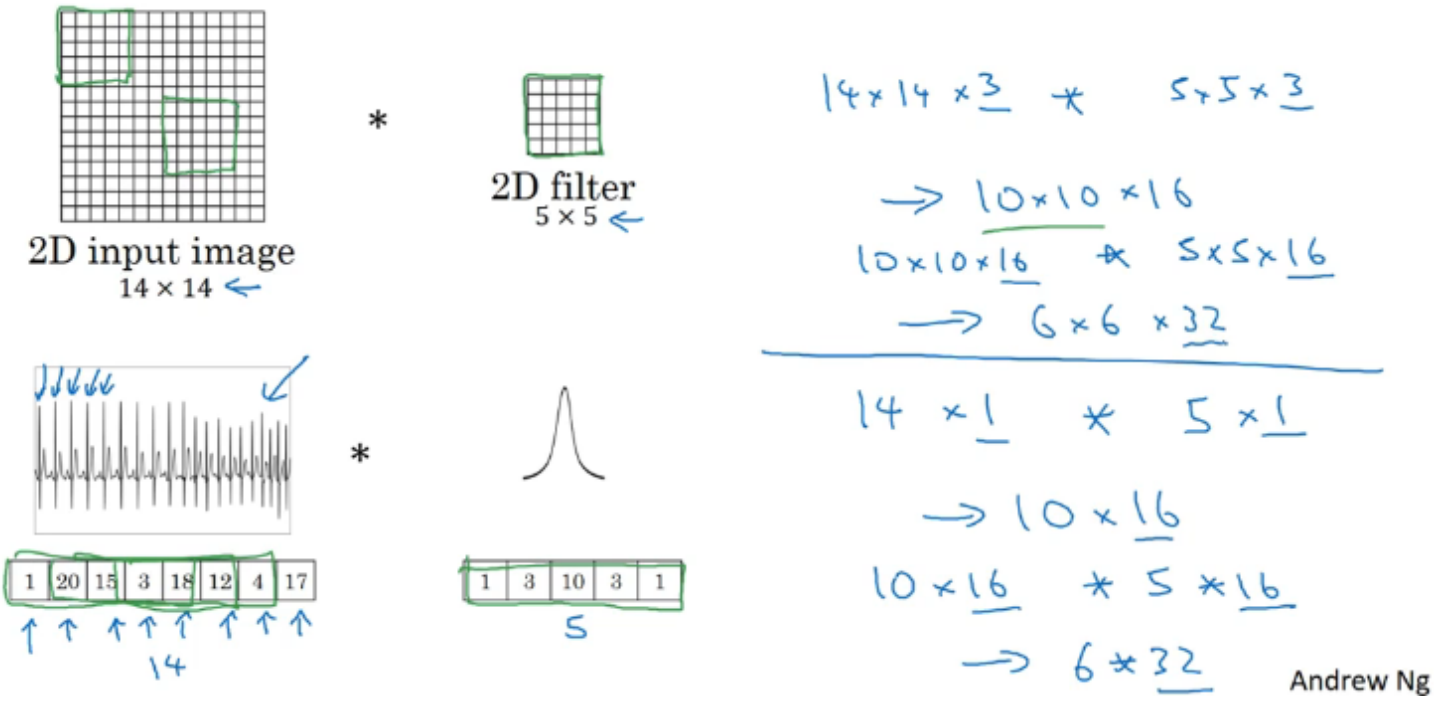
$$J_{style}(S,G) = \sum_l \lambda^{[l]} J_{style}^{[l]}(S,G)$$

- 最终的代价函数可以表示为：

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

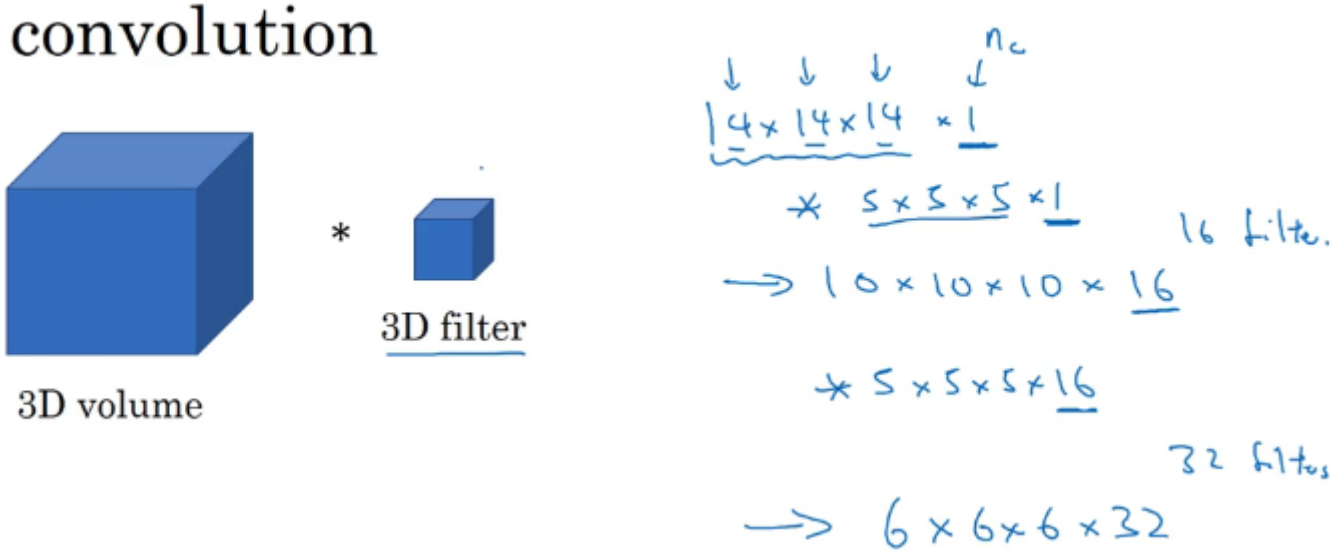
11. 一维到三维的卷积

上面所提到的卷积多数是在二维上进行的，事实上卷积能够推广到一维和三维。



- 2D卷积：$14 \times 14 \times 3 * 5 \times 5 \times 3 \rightarrow 10 \times 10 \times n_c$
- 1D卷积：$14 \times 1 * 5 \times 1 \rightarrow 10 \times n_c$

3D convolution



- 3D卷积：$14 \times 14 \times 14 \times 1 * 5 \times 5 \times 5 \times 1 \rightarrow 10 \times 10 \times 10 \times n_c$
- CT扫描、电影切片等。