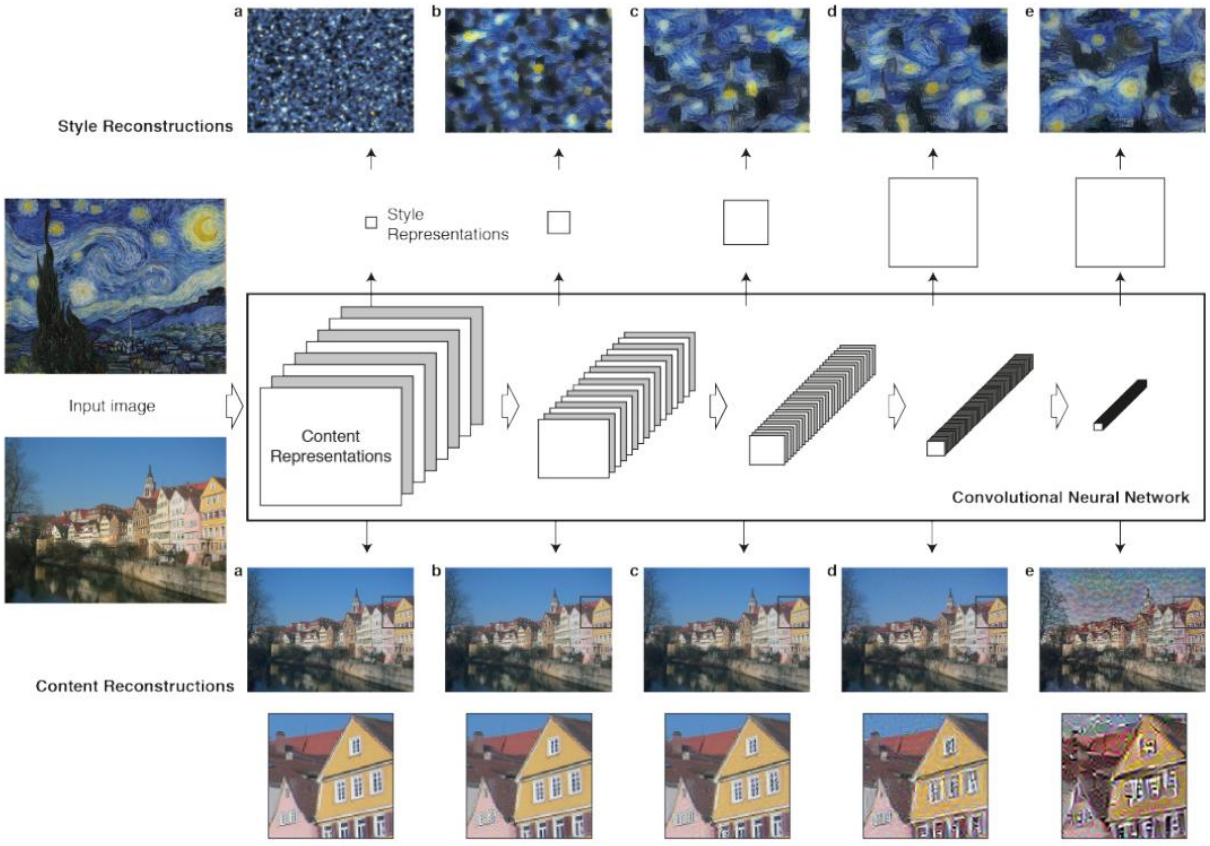
MRP - Week 6

This week we will be experimenting with the input layers taken to evaluate the loss function.

- Content Representation
  - Higher layers capture high-level content features. Not exact pixel value
  - Higher layers are more concerned with the shapes and arrangement of objects in the image
  - We care about these last layers because they contain the content we want to make artistic

- Style Reconstruction
  - To extract the style of an image (i.e. the stroke and colors), we calculate the correlation between the outputs of various layers
  - The similarities(correlations) between various layers captures information about the texture and colors



- The figure above shows a 5 layer Convolutional Network
- The top row (style reconstruction) shows the style extracted each layer
  - The style of the input image from style representations built on different subsets of CNN layers (‘conv1 1’ (a), ‘conv1 1’ and ‘conv2 1’ (b), ‘conv1 1’, ‘conv2 1’ and ‘conv3 1’ (c), ‘conv1 1’, ‘conv2 1’, ‘conv3 1’ and ‘conv4 1’ (d), ‘conv1 1’, ‘conv2 1’, ‘conv3 1’, ‘conv4 1’ and ‘conv5 1’ (e))

- The bottom row (content reconstruction) shows how shapes/colors are preserved but pixels are lost with deeper layers
    - We reconstruct the content image from layers ‘conv1 1’ (a), ‘conv2 1’ (b), ‘conv3 1’ (c), ‘conv4 1’ (d) and ‘conv5 1’ (e)
- The Layers of a Convolutional Network are given in the figure below.
- The green outline indicates the layers of VGG-16 network we will be using

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

```
 'block1_conv1/Relu:0' shape=(3, 300, 300, 64) c
 'block1_conv2/Relu:0' shape=(3, 300, 300, 64) c
'block1_pool/MaxPool:0' shape=(3, 150, 150, 64)
 'block2_conv1/Relu:0' shape=(3, 150, 150, 128)
 'block2_conv2/Relu:0' shape=(3, 150, 150, 128)
'block2_pool/MaxPool:0' shape=(3, 75, 75, 128) c
 'block3_conv1/Relu:0' shape=(3, 75, 75, 256) dt
 'block3_conv2/Relu:0' shape=(3, 75, 75, 256) dt
 'block3_conv3/Relu:0' shape=(3, 75, 75, 256) dt
'block3_pool/MaxPool:0' shape=(3, 37, 37, 256) c
 'block4_conv1/Relu:0' shape=(3, 37, 37, 512) dt
 'block4_conv2/Relu:0' shape=(3, 37, 37, 512) dt
 'block4_conv3/Relu:0' shape=(3, 37, 37, 512) dt
'block4_pool/MaxPool:0' shape=(3, 18, 18, 512) c
 'block5_conv1/Relu:0' shape=(3, 18, 18, 512) dt
 'block5_conv2/Relu:0' shape=(3, 18, 18, 512) dt
 'block5_conv3/Relu:0' shape=(3, 18, 18, 512) dt
'block5_pool/MaxPool:0' shape=(3, 9, 9, 512) dty
```

Loss function

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

- Leon Gatys used the following layers as input for the loss functions in his paper:
    - Content loss: block4_conv2
    - Style loss: block1_conv1, block2_conv1, block3_conv1, block4_conv1, block5_conv1
    - α /β = 1x10^-3 or 1x10^-4
- The layers used for the style loss function are spread across the all 5 blocks of the VGG-16 network. This gives the best results because we are able to form correlations between at least one layer of each block
- We can vary the layer used for the content loss function to see which layer gives the best results

Experiments:

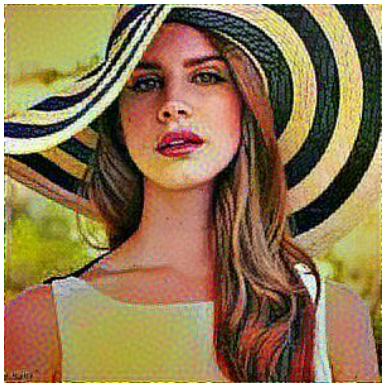
α = 0.5
β = 5.0
Style layers = block1_conv1, block2_conv1, block3_conv1, block4_conv1, block5_conv1

Exp#1:

Content layer = block1_cov2

Exp#2:

Content layer = block2_cov2



Exp#3:

Content layer = block3_cov3

Exp#4:

Content layer = block4_cov3

Exp#5:

Content layer = block5_cov3

The results of the generated image before block3_conv3 are weak and after block4_conv3 are too strong. Maybe the ideal layer is somewhere in block 4 layer 1 or 2.

Exp#6:

Content layer = block4_cov1

Exp#7:

Content layer = block4_cov

Experiment 6 seems to offer the best results visually speaking.