EE655: Computer Vision & Deep Learning

Lecture 16

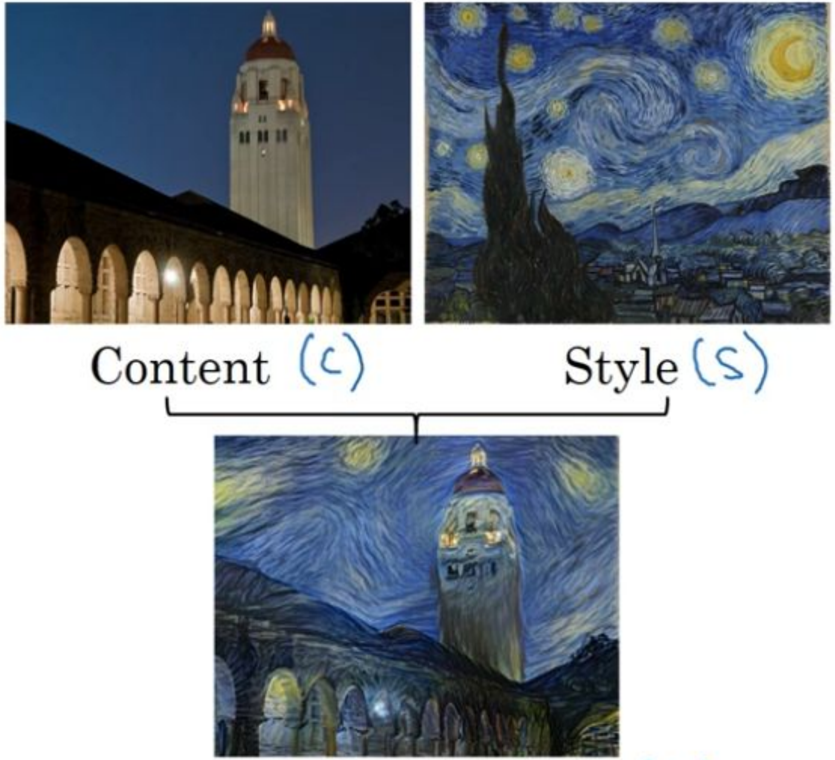
Koteswar Rao Jerripothula, PhD
Department of Electrical Engineering
IIT Kanpur

Overview

Neural Style Transfer

Generative Adversarial Networks (GANs)

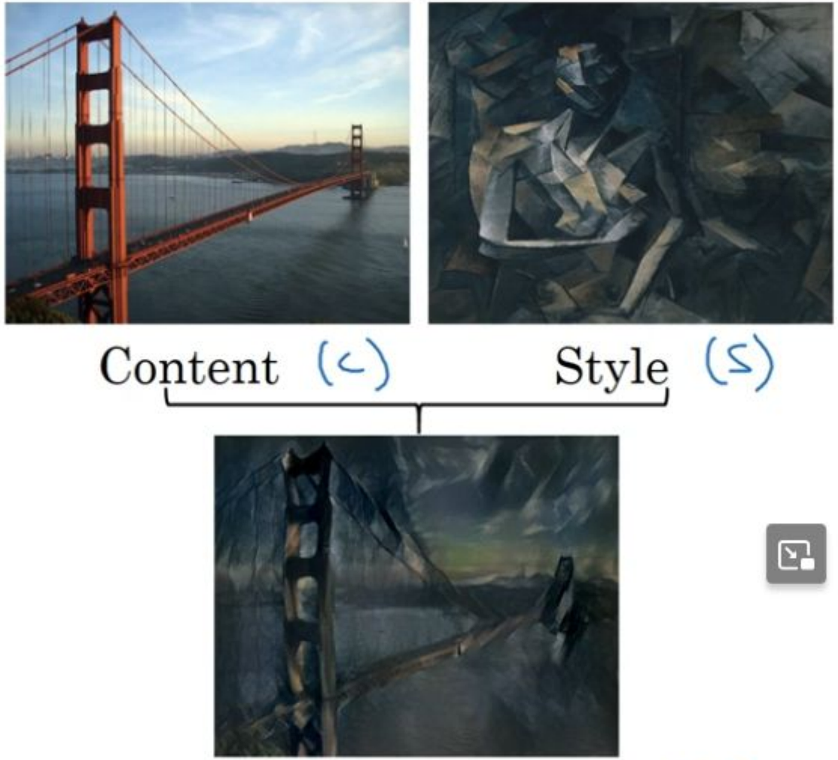
Neural style transfer



Content (C)

Style (S)

Generated image (G)

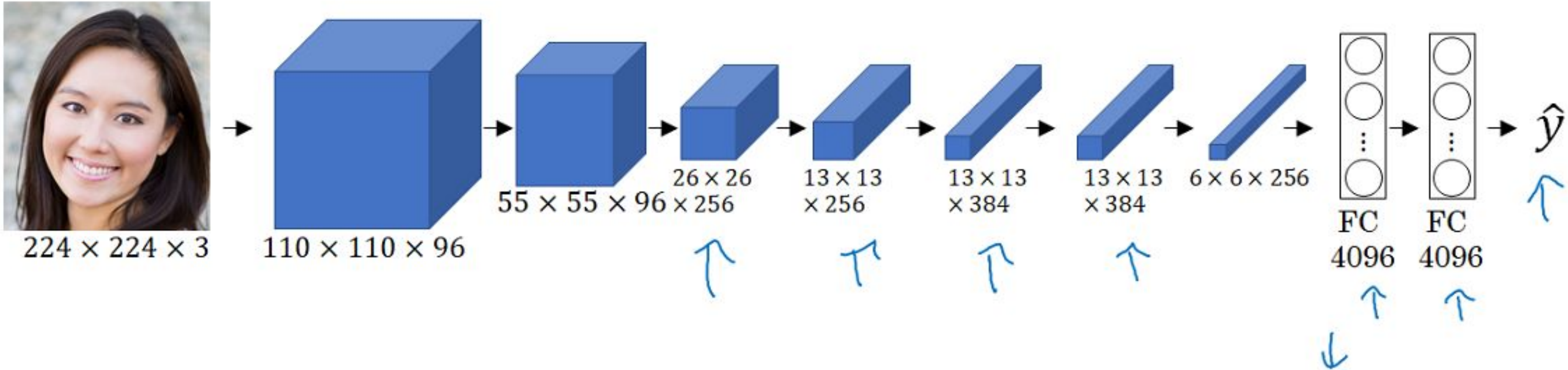


Content (C)

Style (S)

Generated image (G)

Visualizing what a deep network is learning

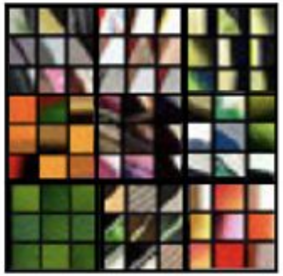


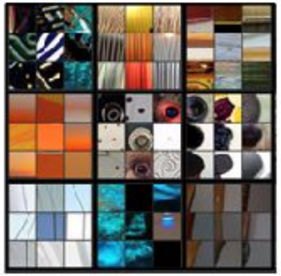
Pick a unit in layer 1. Find the nine image patches that maximize the unit's activation.

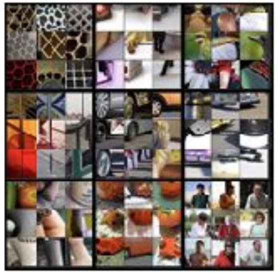
Repeat for other units.

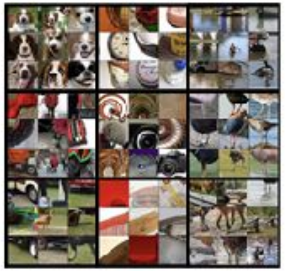
[Zeiler and Fergus., 2013, Visualizing and understanding convolutional networks]

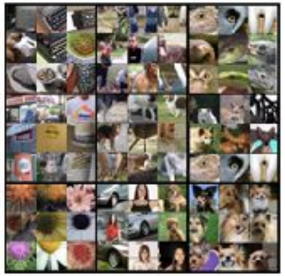
Visualizing deep layers: Layer 1

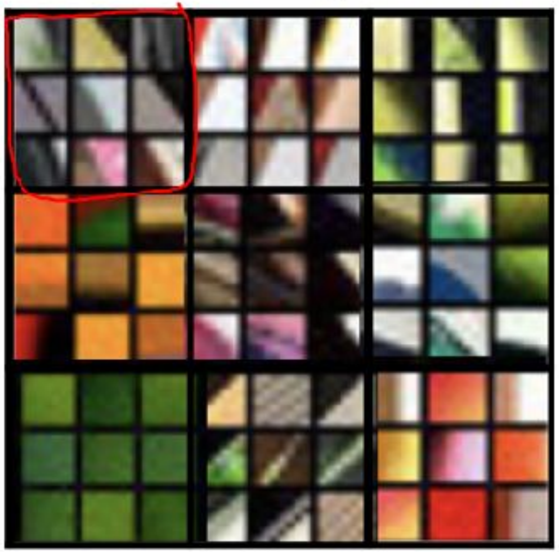

Layer 1


Layer 2


Layer 3


Layer 4


Layer 5



Visualizing deep layers: Layer 2

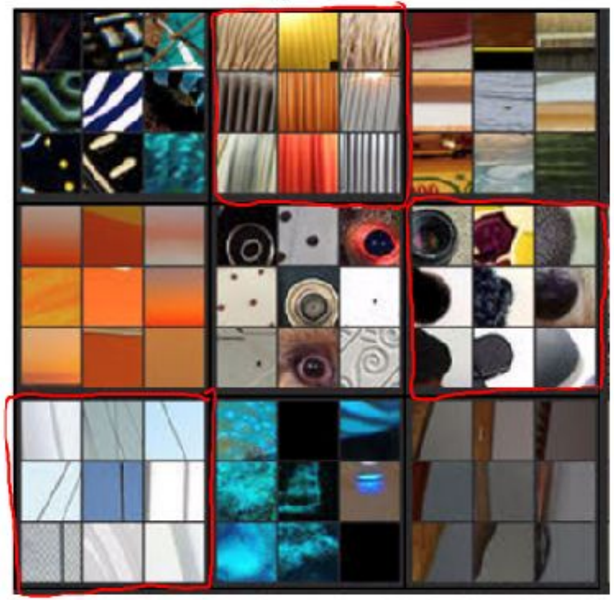
Layer 1

Layer 2

Layer 3

Layer 4

Layer 5



Visualizing deep layers: Layer 3

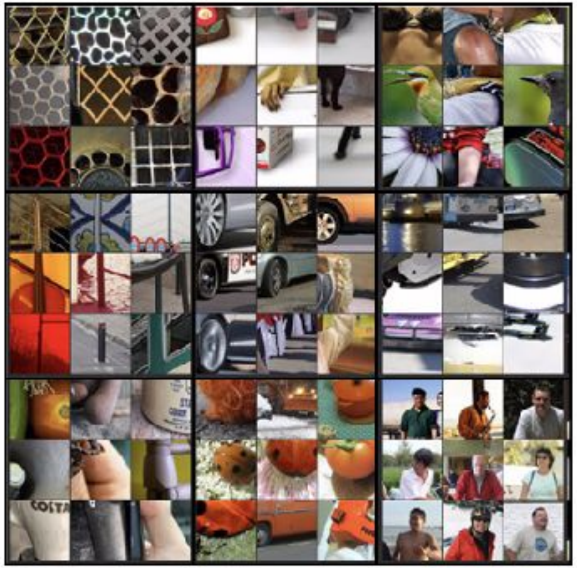
Layer 1

Layer 2

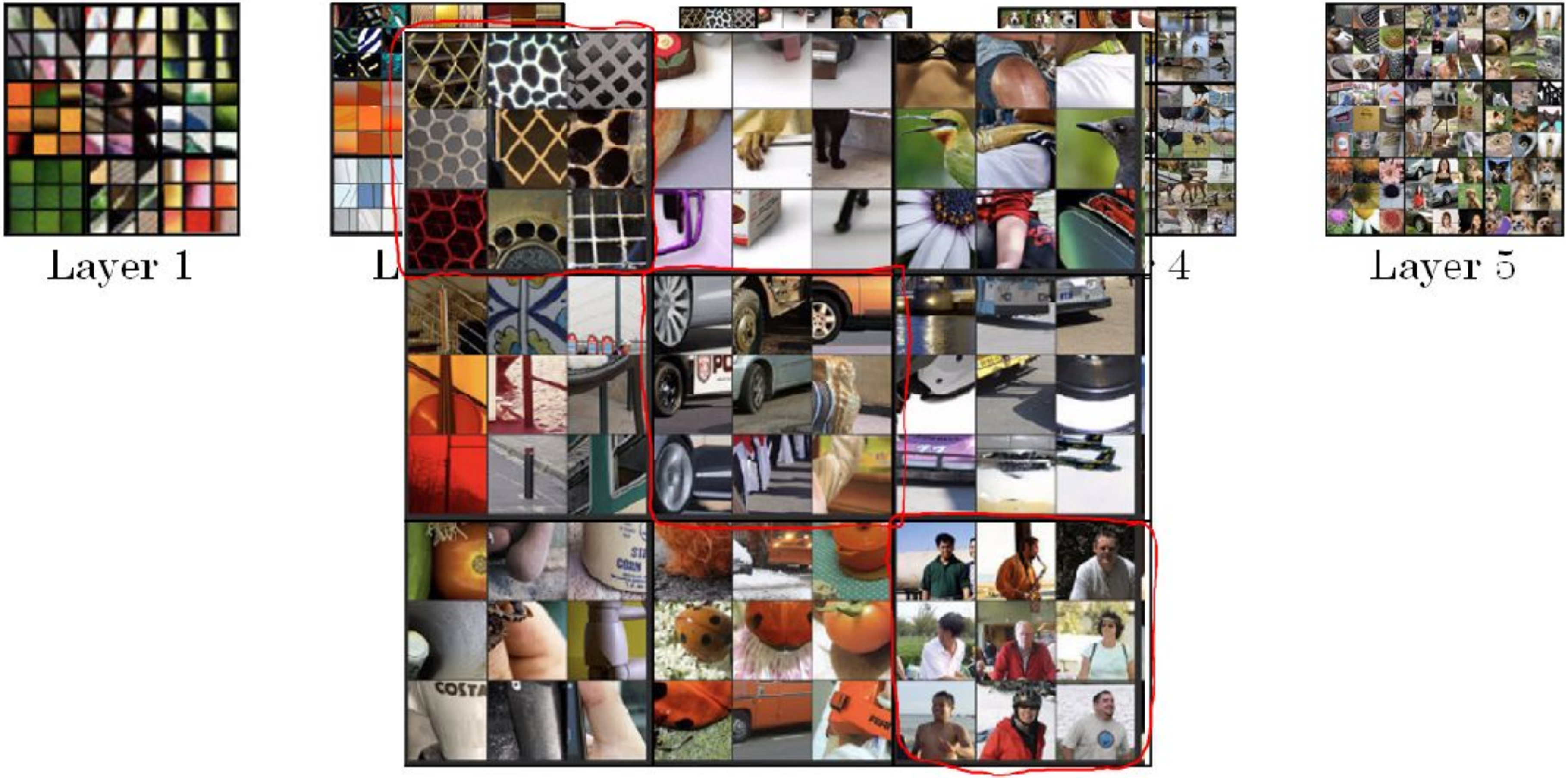
Layer 3

Layer 4

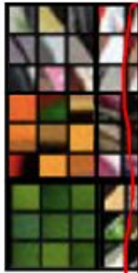
Layer 5



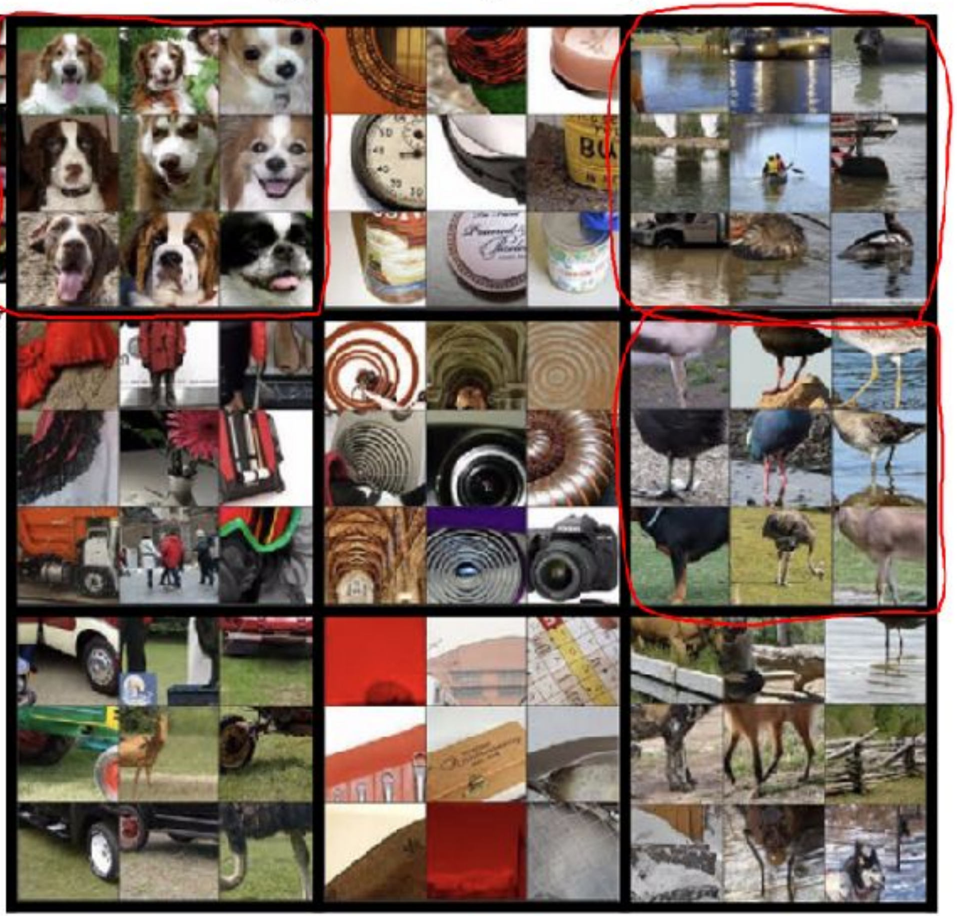
Visualizing deep layers: Layer 3



Visualizing deep layers: Layer 4



Lay

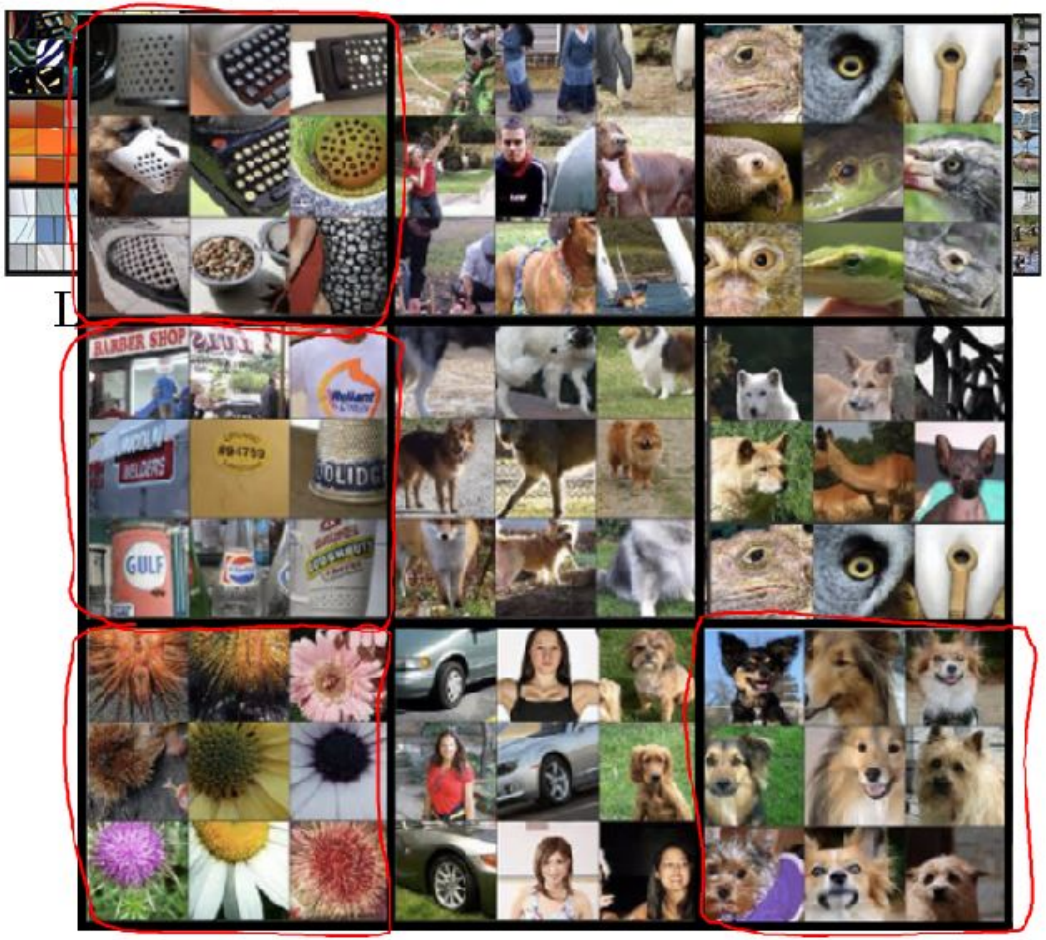


Layer 4

Layer 5

Visualizing deep layers: Layer 5

Layer 1



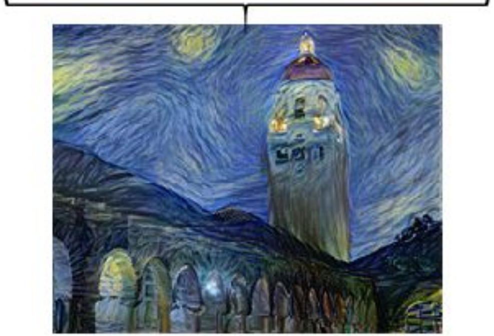
L

Layer 5

Neural style transfer cost function

Content C

Style S



Generated image G ←

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

[Gatys et al., 2015. A neural algorithm of artistic style. Images on slide generated by Justin Johnson]

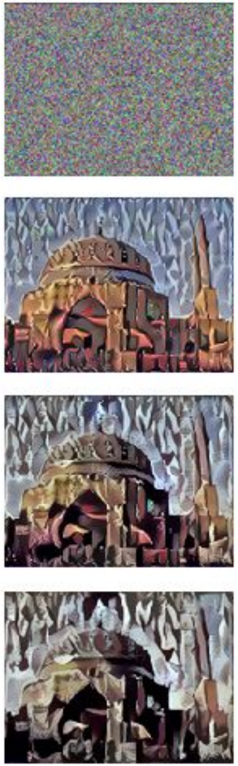
Find the generated image G

1. Initiatc G randomly

   $\underline{\text{G}}: \underline{100} \times \underline{100} \times \underset{\underset{\text{RGB}}{\uparrow}}{3}$

2. Use gradient descent to minimize $\underline{J(G)}$

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



[Gatys ct al., 2015. A ncural algorithm of artistic stylc]

Content cost function

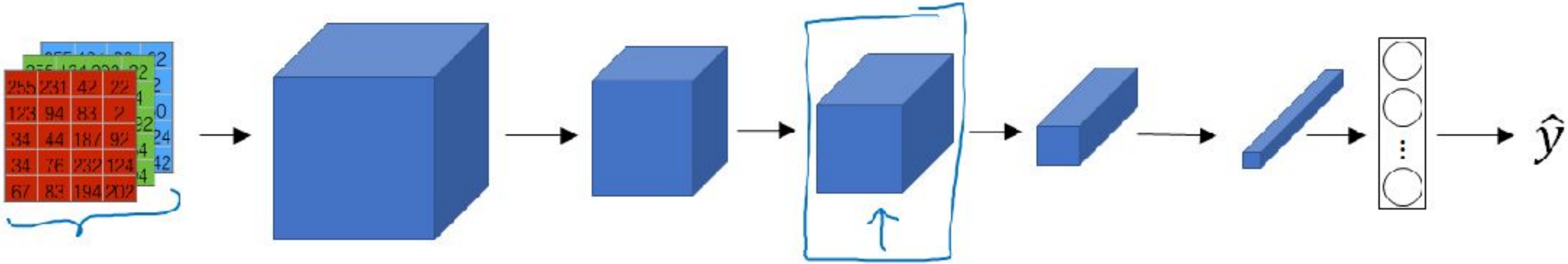
$$\underline{J(G)} = \underset{\uparrow}{\alpha}\, \underline{J_{content}(\overset{\downarrow}{C}, \overset{\downarrow}{G})} + \beta\, J_{style}(S, G)$$

- Say you use hidden layer $\underline{l}$ to compute content cost.
- Use pre-trained ConvNet. (E.g., VGG network)
- Let $\underline{a^{[l](C)}}$ and $\underline{a^{[l](G)}}$ be the activation of layer l on the images
- If $a^{[l](C)}$ and $a^{[l](G)}$ are similar, both images have similar content

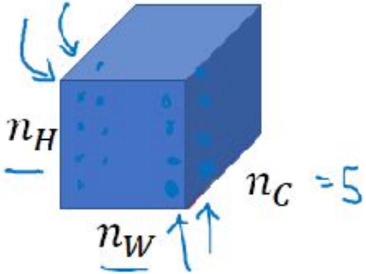
$$J_{content}(C, \underset{\uparrow}{G}) = \frac{1}{2} \underbrace{\| \overset{\downarrow}{\underline{a}}^{[l](C)} - \overset{\downarrow}{\underline{a}}^{[l](G)} \|^2}$$

[Gatys et al., 2015. A neural algorithm of artistic style]

Meaning of the "style" of an image



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

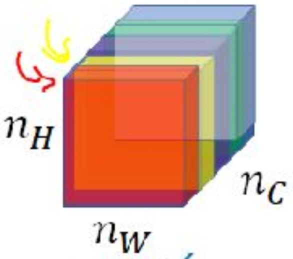


How correlated are the activations across different channels?

[Gatys ct al., 2015. A ncural algorithm of artistic stylc]

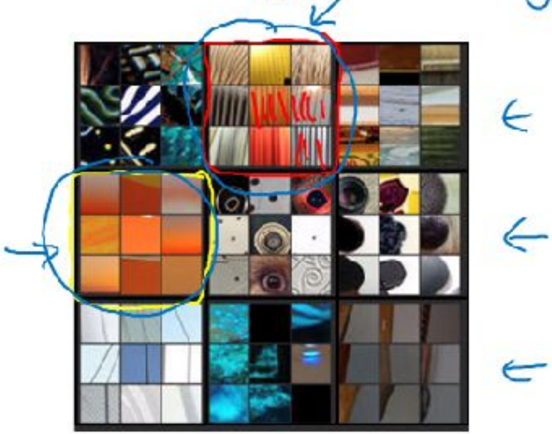
Intuition about style of an image

Style image



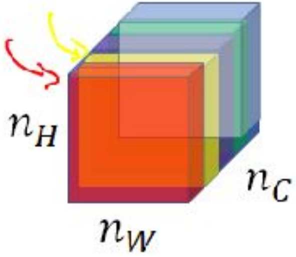
Correlated?

Uncorrelated



[Gatys et al., 2015. A neural algorithm of artistic style]

Generated Image



Style matrix

Let $\underline{a^{[l]}_{i,j,k}}$ = activation at $(\overset{\overset{H}{\downarrow}}{i}, \overset{\overset{W}{\downarrow}}{j}, \overset{\overset{C}{\swarrow}}{k})$. $\underline{G^{[l]}}$ is $\underline{n_c^{[l]}} \times \underline{n_c^{[l]}}$

$$\overset{n_c}{G^{[l]}_{\underset{\uparrow\ \uparrow}{kk'}}} \qquad \underline{k = 1, \dots, n_c^{[l]}}$$

$$\rightarrow \underline{G^{[l](S)}_{kk'}} = \sum_{i=1}^{n_H^{[l]}} \sum_{j=1}^{n_W^{[l]}} \underline{a}^{[l](S)}_{ij\underset{\uparrow}{k}} \, \underline{a}^{[l](S)}_{ij\underset{\uparrow}{k'}}$$

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

"$\underline{\text{Gram}}$ matrix"

$$J^{[l]}_{style}(S,G) = \frac{1}{\underset{C}{(\dots)}} \left\| G^{[l](S)} - G^{[l](G)} \right\|_F^2$$

$$\overset{\uparrow}{\beta} \qquad = \frac{1}{(2 n_H^{[l]} n_W^{[l]} n_C^{[l]})^2} \sum_k \sum_{k'} \left(G^{[l](S)}_{kk'} - G^{[l](G)}_{kk'} \right)^2$$

[Gatys et al., 2015. A neural algorithm of artistic style]

Style cost function

$$\left\| G^{[l](S)} - G^{[l](G)} \right\|_F^2$$

$$J^{[l]}_{style}(S,G) = \frac{1}{\left(2n_H^{[l]}n_W^{[l]}n_C^{[l]}\right)^2} \sum_k \sum_{k'} \left(G_{kk'}^{[l](S)} - G_{kk'}^{[l](G)}\right)^2$$

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

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

[Gatys et al., 2015. A neural algorithm of artistic style]

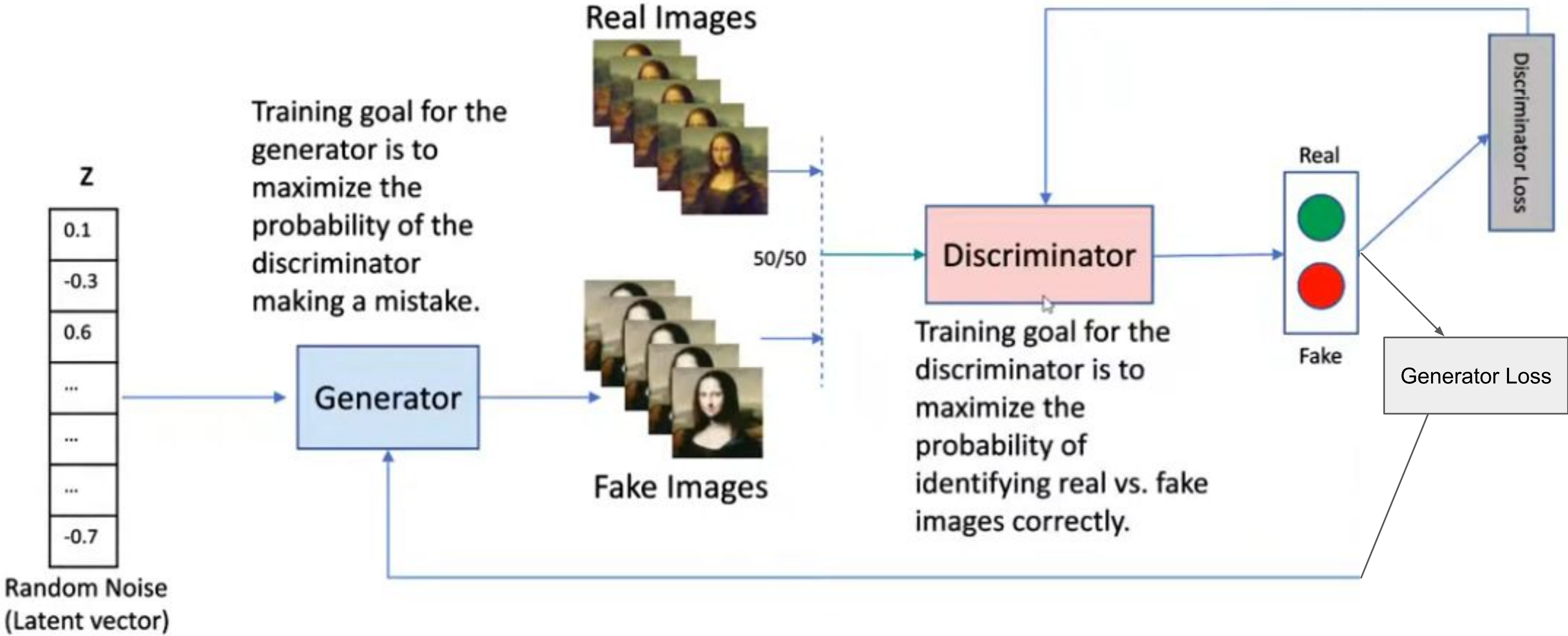
Generative

Generates data
(Creates fake data)

Adversarial

Generator and **discriminator**, each competing to win.

Generator trying to fake
and
Discriminator, trying not to be fooled.



z

0.1

-0.3

0.6

...

...

...

-0.7

Random Noise
(Latent vector)

Training goal for the generator is to maximize the probability of the discriminator making a mistake.

Generator

Real Images

Fake Images

50/50

Discriminator

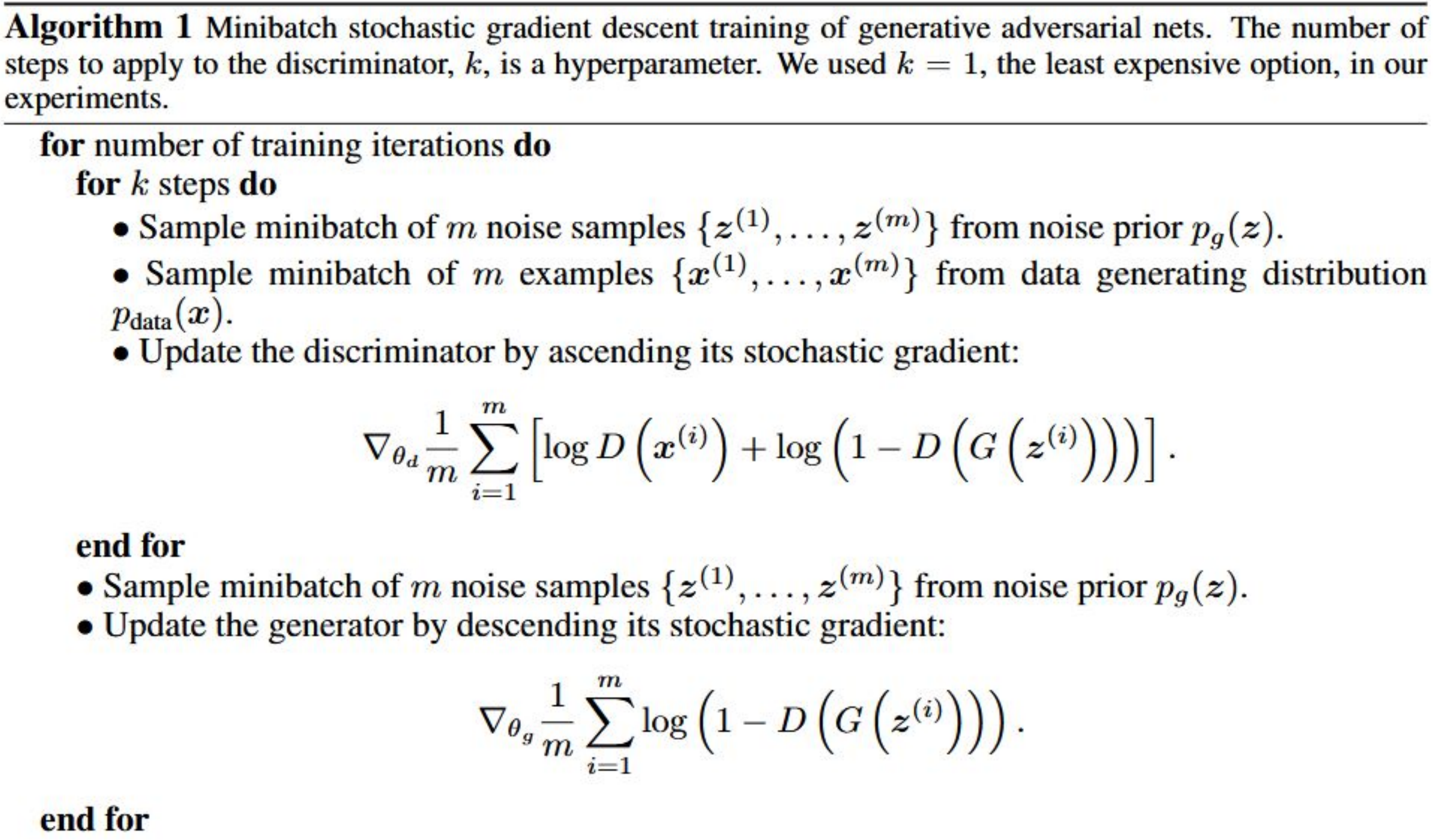
Training goal for the discriminator is to maximize the probability of identifying real vs. fake images correctly.

Real

Fake

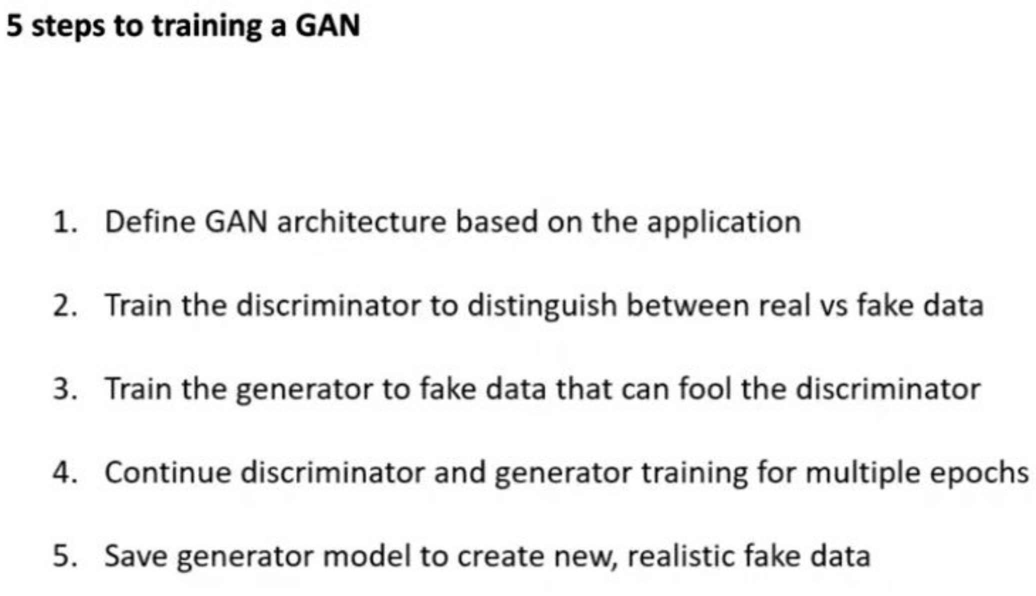
Discriminator Loss

Generator Loss



Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k, is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

```
for number of training iterations do
    for $k$ steps do
        • Sample minibatch of $m$ noise samples $\{\boldsymbol{z}^{(1)}, \dots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
        • Sample minibatch of $m$ examples $\{\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\boldsymbol{x})$.
        • Update the discriminator by ascending its stochastic gradient:
            $\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[ \log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right) \right].$
    end for
    • Sample minibatch of $m$ noise samples $\{\boldsymbol{z}^{(1)}, \dots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
    • Update the generator by descending its stochastic gradient:
        $\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right).$
end for
```

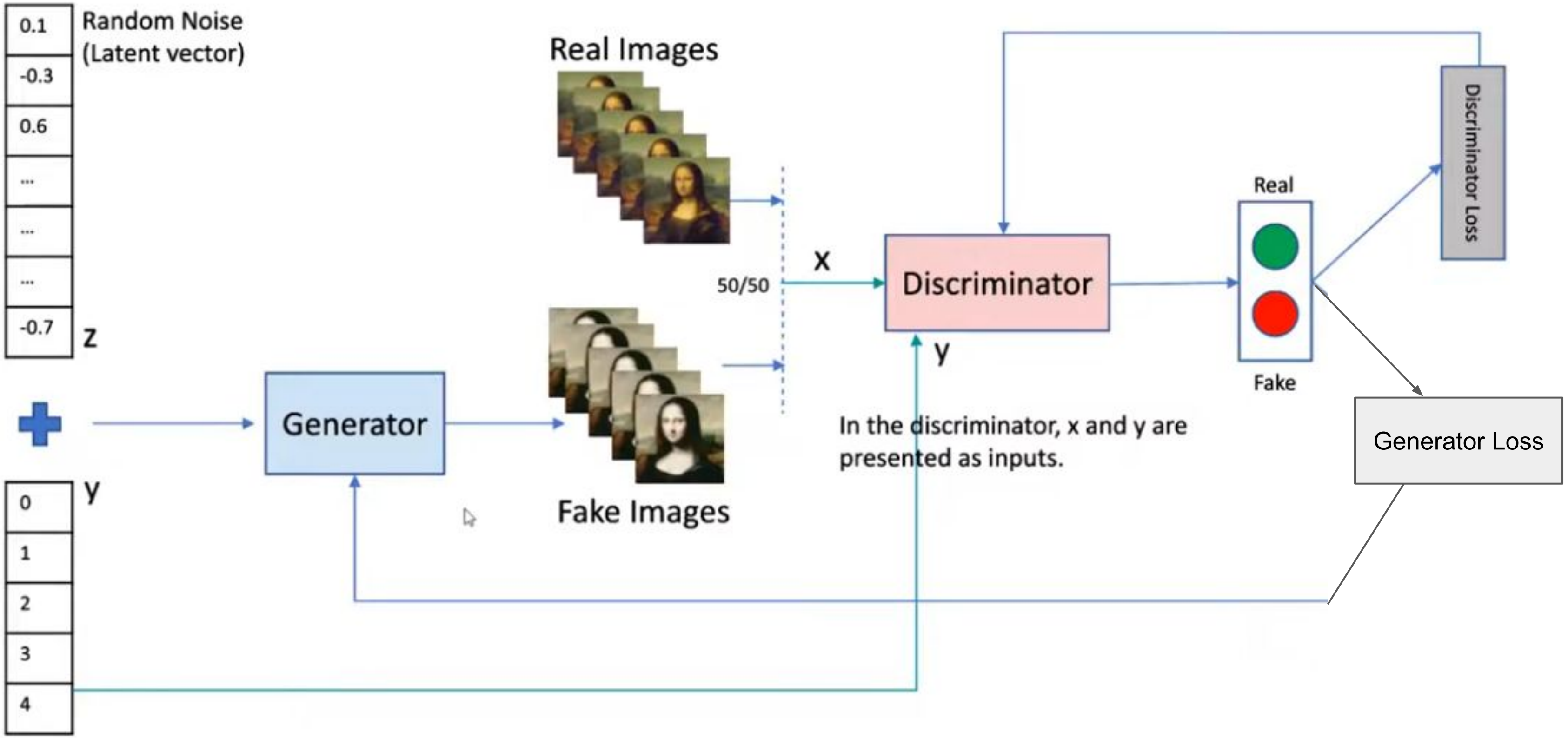


5 steps to training a GAN

1. Define GAN architecture based on the application
2. Train the discriminator to distinguish between real vs fake data
3. Train the generator to fake data that can fool the discriminator
4. Continue discriminator and generator training for multiple epochs
5. Save generator model to create new, realistic fake data

NOTE: When training the discriminator, hold the generator values constant; and when training the generator, hold the discriminator values constant. Each should train against a static adversary.

Conditional Generative Adversarial Network



In the discriminator, x and y are presented as inputs.

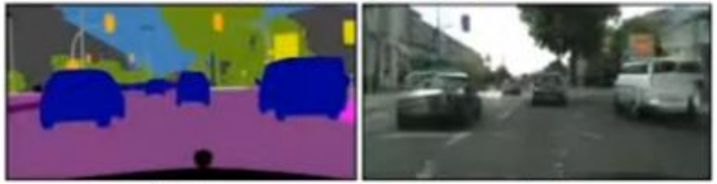
Conditional data (y): can be class labels or data from other modalities.

Applications of Conditional GANs

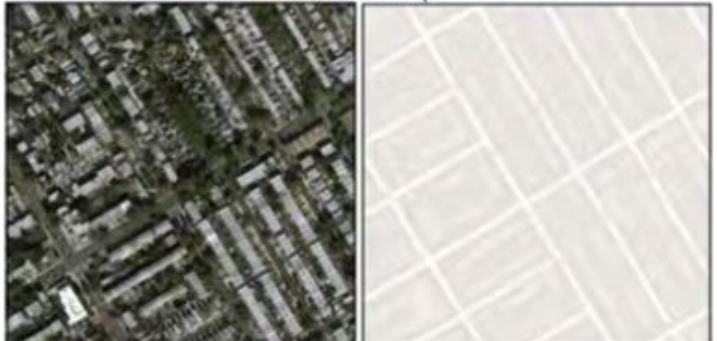
Image-to-Image Translation: Pix2Pix GAN

- Takes an image as input and maps it to a generated output image with different properties.
- Example: Train an image-to-image GAN to take sketches of handbags and turn them into photorealistic images of handbags.
- The system requires pairwise correspondences between images during training.

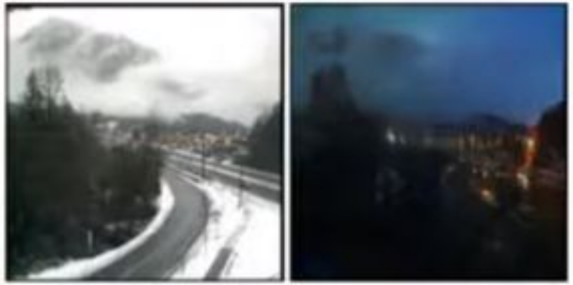
Labels to Street Scene



Aerial to Map



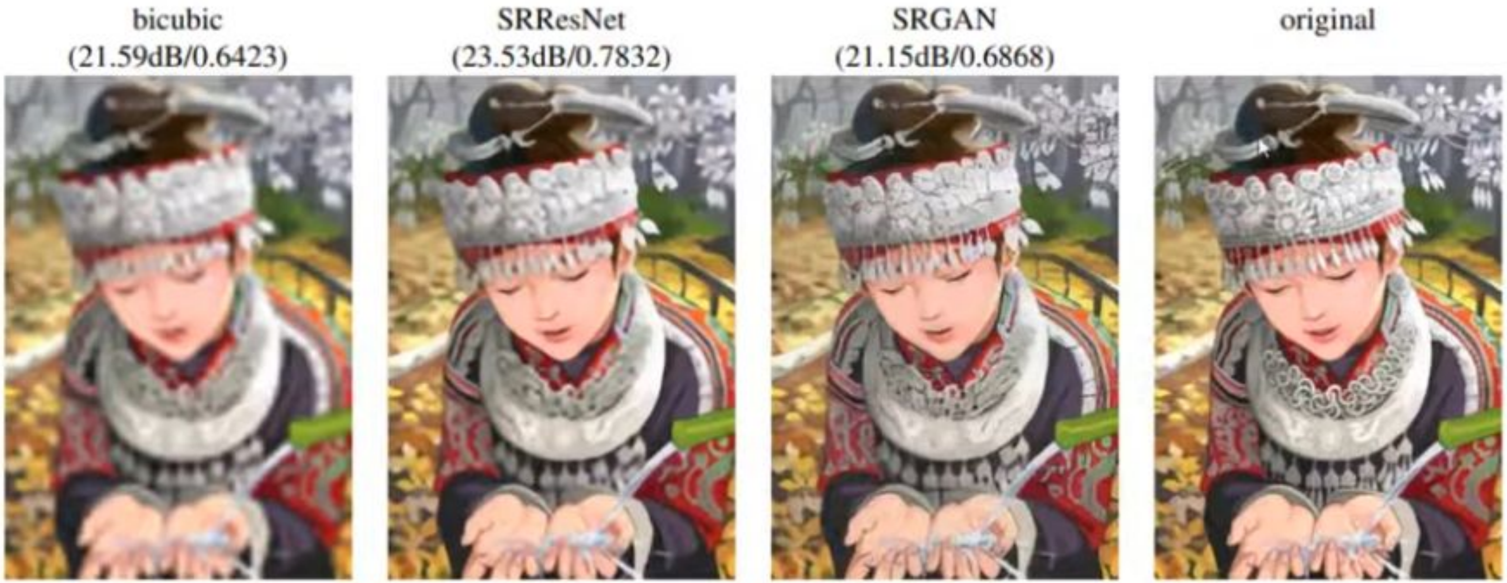
Day to Night



Applications of Conditional GANs

Super-resolution

Increase the resolution of images, adding detail where necessary to fill in blurry areas.

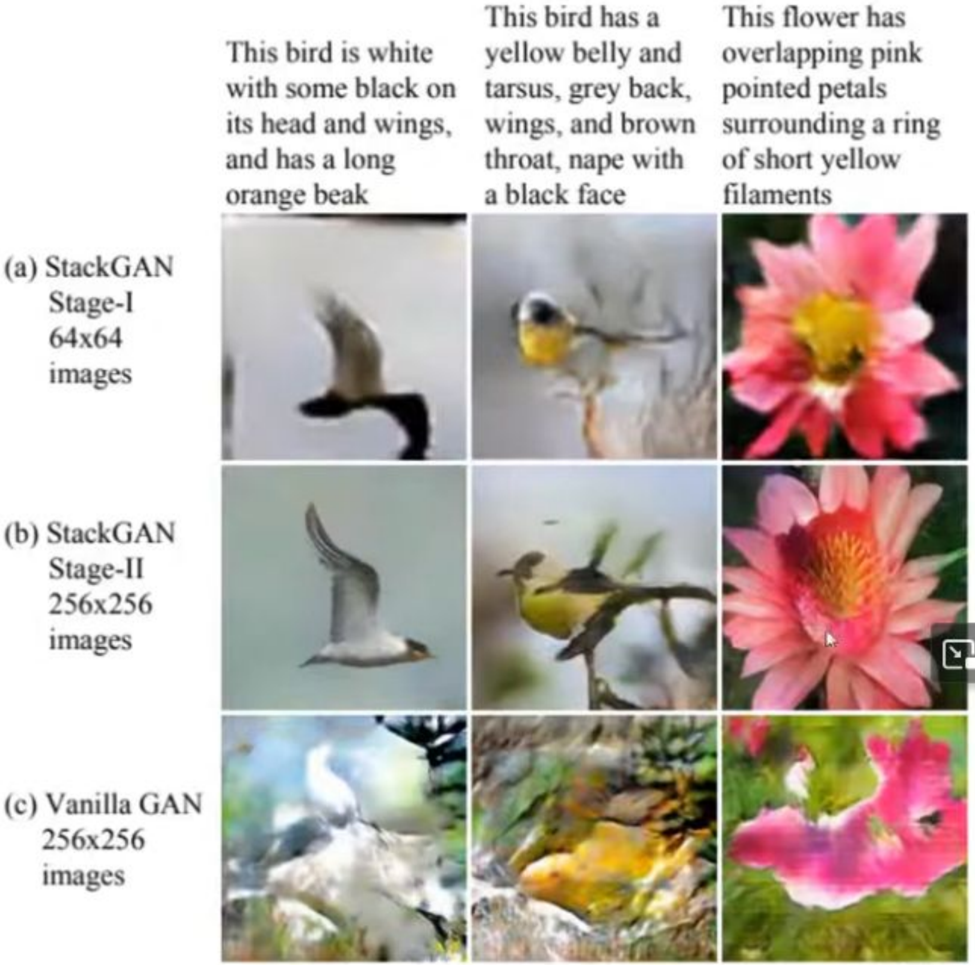


The GAN-generated image looks very similar to the original image, but if you look closely at the headband...

Applications of Conditional GANs

Text-to-Image Synthesis

Take text as input and produce images as described by the text.



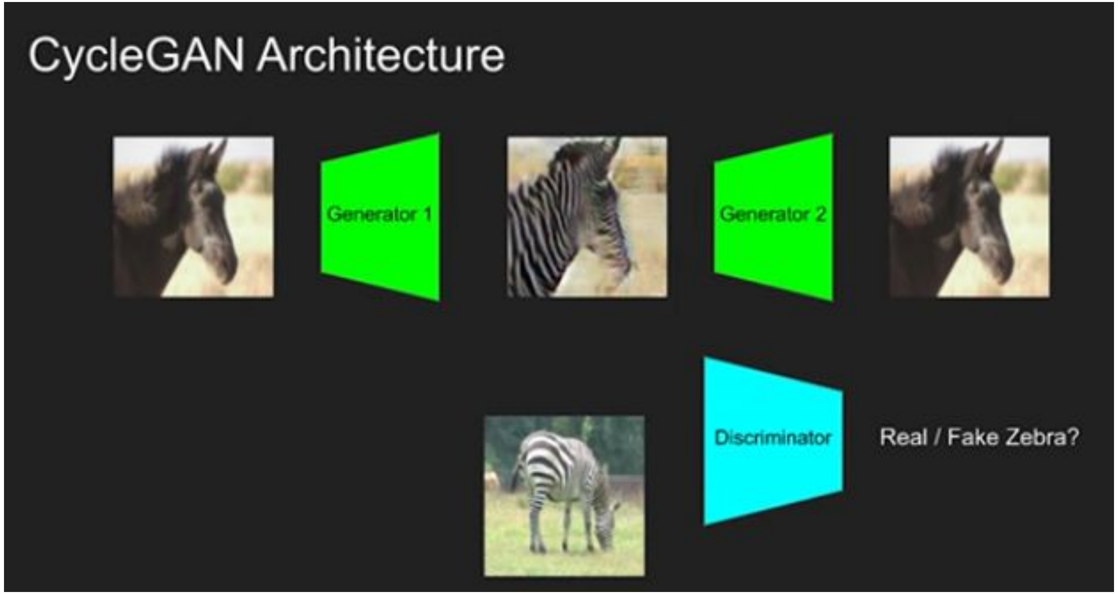
This bird is white with some black on its head and wings, and has a long orange beak

This bird has a yellow belly and tarsus, grey back, wings, and brown throat, nape with a black face

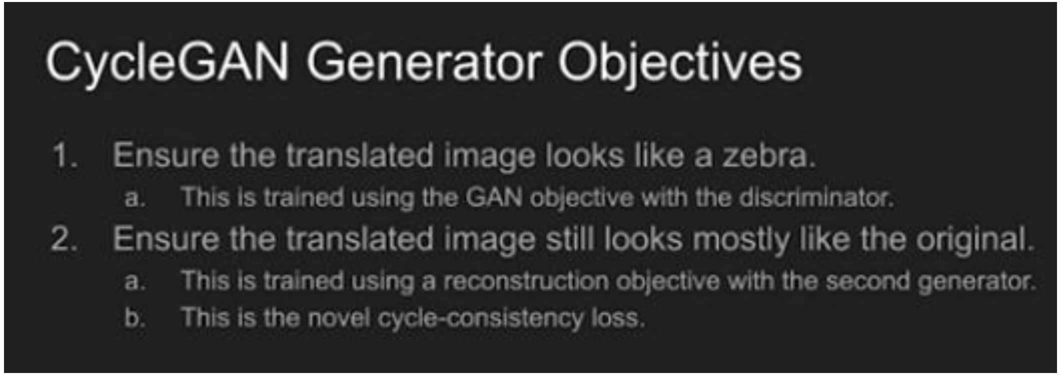
This flower has overlapping pink pointed petals surrounding a ring of short yellow filaments

(a) StackGAN Stage-I 64x64 images
(b) StackGAN Stage-II 256x256 images
(c) Vanilla GAN 256x256 images

https://arxiv.org/abs/1612.03242



CycleGAN Architecture



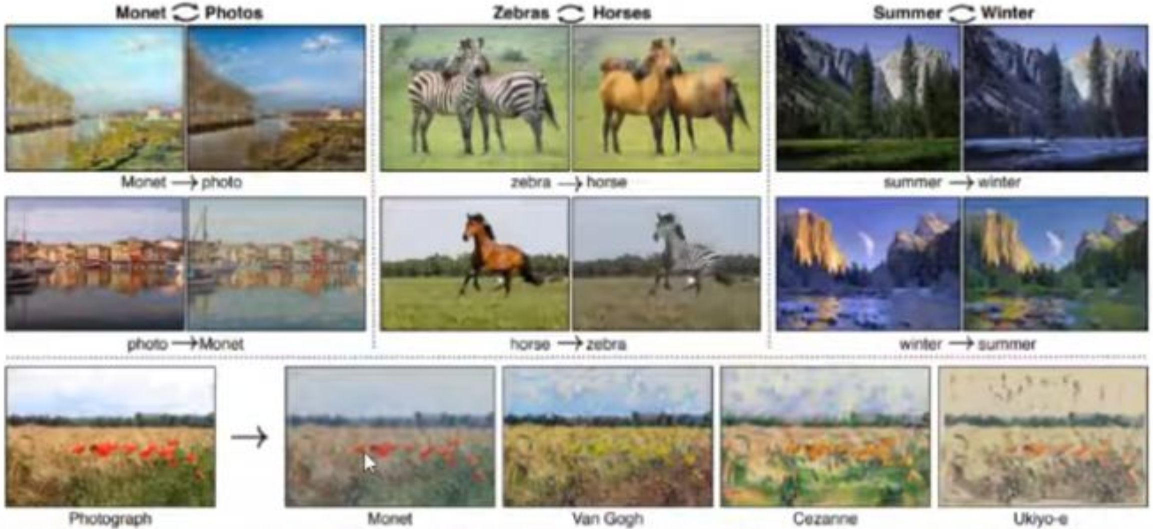
CycleGAN Generator Objectives

1. Ensure the translated image looks like a zebra.
   a. This is trained using the GAN objective with the discriminator.
2. Ensure the translated image still looks mostly like the original.
   a. This is trained using a reconstruction objective with the second generator.
   b. This is the novel cycle-consistency loss.

Applications of Conditional GANs

CycleGAN

- Transform images from one set into images that could belong to another set.
- Example: Convert an image of a horse into an image of a zebra.
- The training data for the CycleGAN is simply two sets of images (e.g., a set of horse images and a set of zebra images).
- The system requires no labels or pairwise correspondences between images for training.



References

https://www.youtube.com/watch?v=R39tWYYKNcI&list=PLkDaE6sCZn6Gl29AoE31iwdVwSG-KnDzF&index=37

https://www.youtube.com/watch?v=ChoV5h7tw5A&list=PLkDaE6sCZn6Gl29AoE31iwdVwSG-KnDzF&index=38

https://www.youtube.com/watch?v=xY-DMAJpIP4&list=PLkDaE6sCZn6Gl29AoE31iwdVwSG-KnDzF&index=39

https://www.youtube.com/watch?v=b1I5X3UfEYI&list=PLkDaE6sCZn6Gl29AoE31iwdVwSG-KnDzF&index=40

https://www.youtube.com/watch?v=W5NPlZzebO0

https://www.youtube.com/watch?v=-8hfnlxEPn4