Machine Learning

RAJESH SHARMA
Walt Disney Animation Studios

Thank you to ACM SIGGRAPH!

Pol Jeremias-Vila : SIGGRAPH 2021 Chair

Tomasz Bednarz: Frontiers Program Chair

Alex Bryant: Student Volunteers Chair

Tim Hendrickson: Digital Marketing Manager

Student Volunteers:
Rogelio, Trinity, Aurora, Emily, Hunter & Kendra

SIGGRAPH 2021

5

Machine Learning

Rajesh Sharma

Ming-Yu Liu

Research Scientist

NVIDIA

Ming-Yu Liu is a distinguished research scientist at NVIDIA Research. He received the R&D 100 Award by R&D Magazine for his robotic bin picking system in 2014. His semantic image synthesis paper and scene understanding paper are in the best paper finalist in the 2019 CVPR and 2015 RSS conferences, respectively. In SIGGRAPH 2019, he won the Best in Show Award and Audience Choice Award in the Real-Time Live show for his image synthesis work. His research focus is on generative image modeling. His goal is to enable machines' human-like imagination capability.
He is the author of popular Generative models: GauGan and GanCraft.

Research Scientist

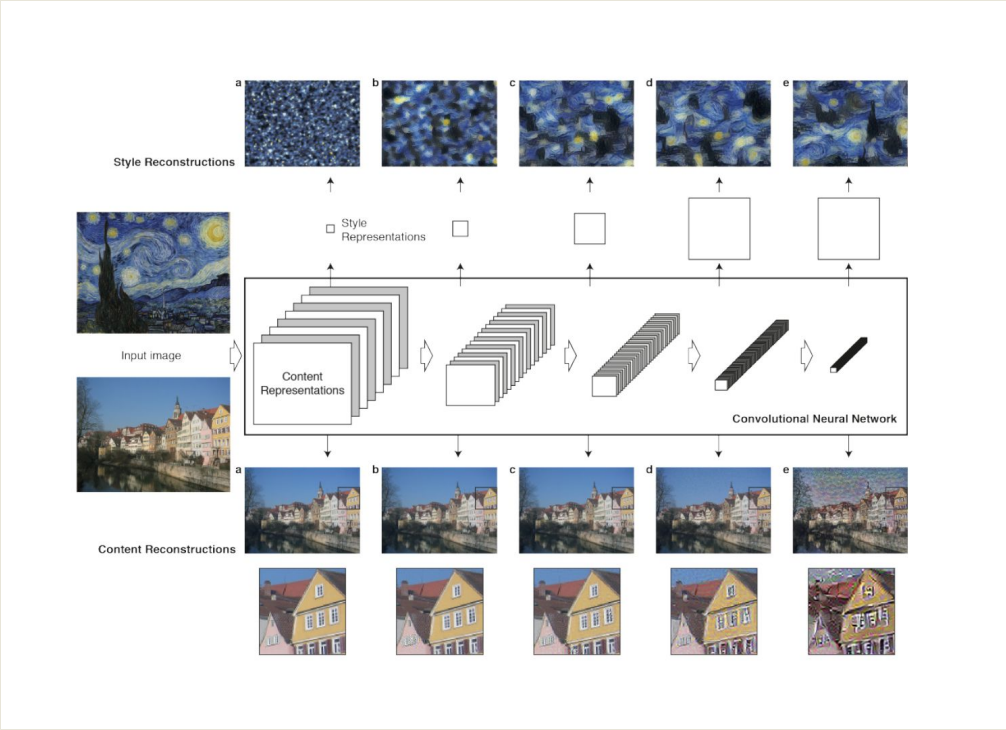
Today

- Quick Recap: Distributions, Autoencoder
- Transfer Learning / PreTrained Models
- Variational Autoencoder
- Generative Adversarial Networks (GAN)

Hands-on

- ★ Log in to your google drive
- ★ Make a shortcut to: `https://bit.ly/3oKCVCh`
- ★ Make a copy of:
  - `Autoencoder.ipynb`
  - `dataPipeline.ipynb`
  - `denoiserCNN.ipynb`
  - `styleTransfer.ipynb`
  - `facialRecognition01.ipynb`
  - `mnistGAN.ipynb`

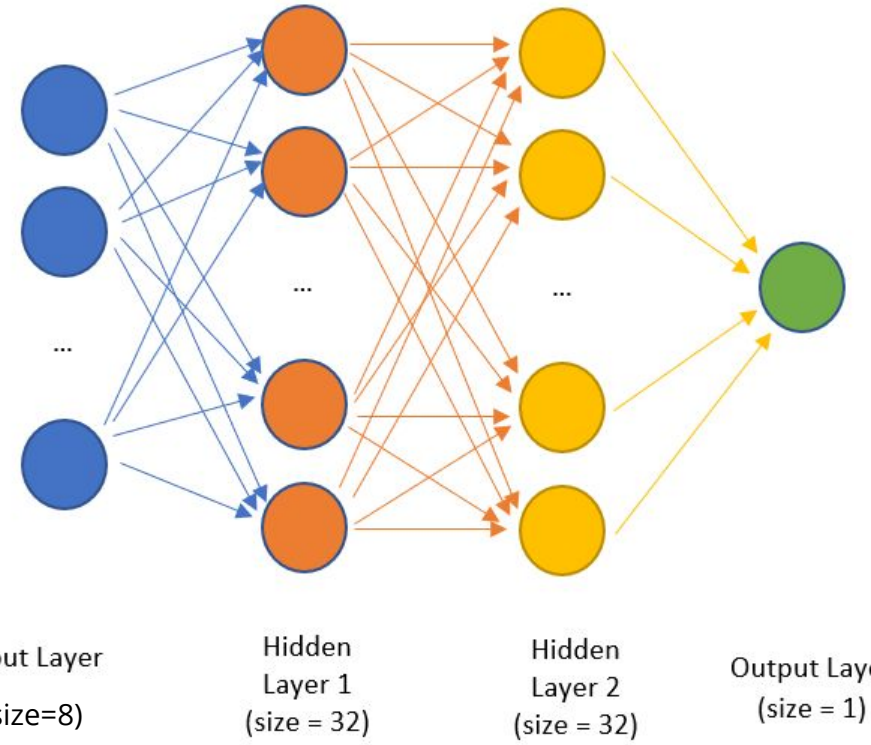
Artistic Style Transfer



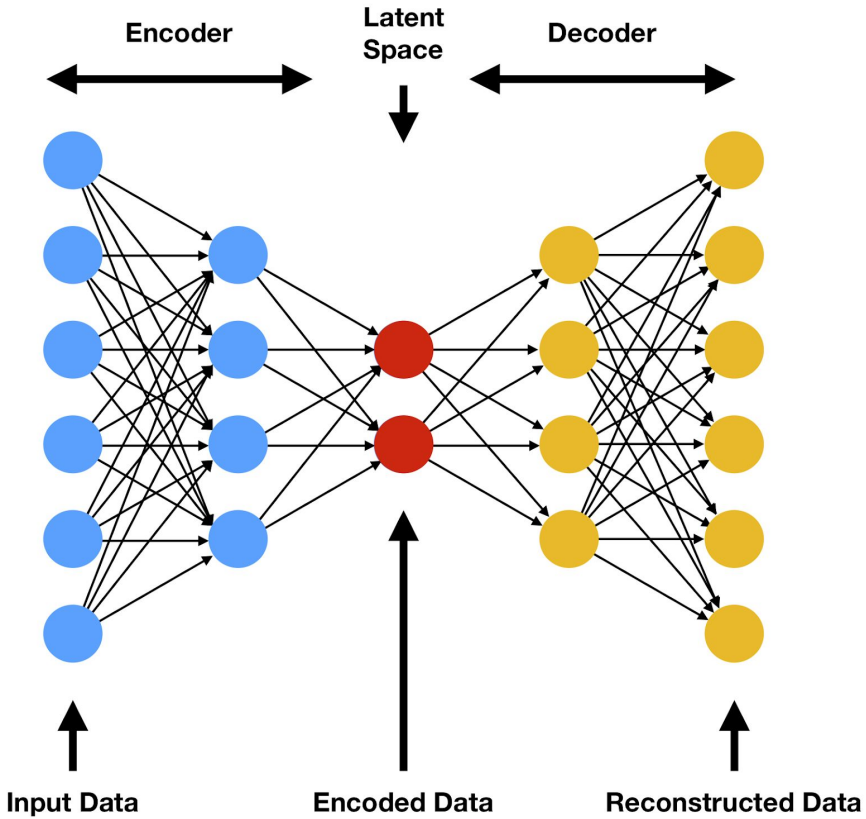
$$L_{total}(\dot{c}, \dot{s}, \dot{x}) = \alpha L_{content}(\dot{c}, \dot{x}) + \beta L_{style}(\dot{s}, \dot{x})$$

Autoencoder

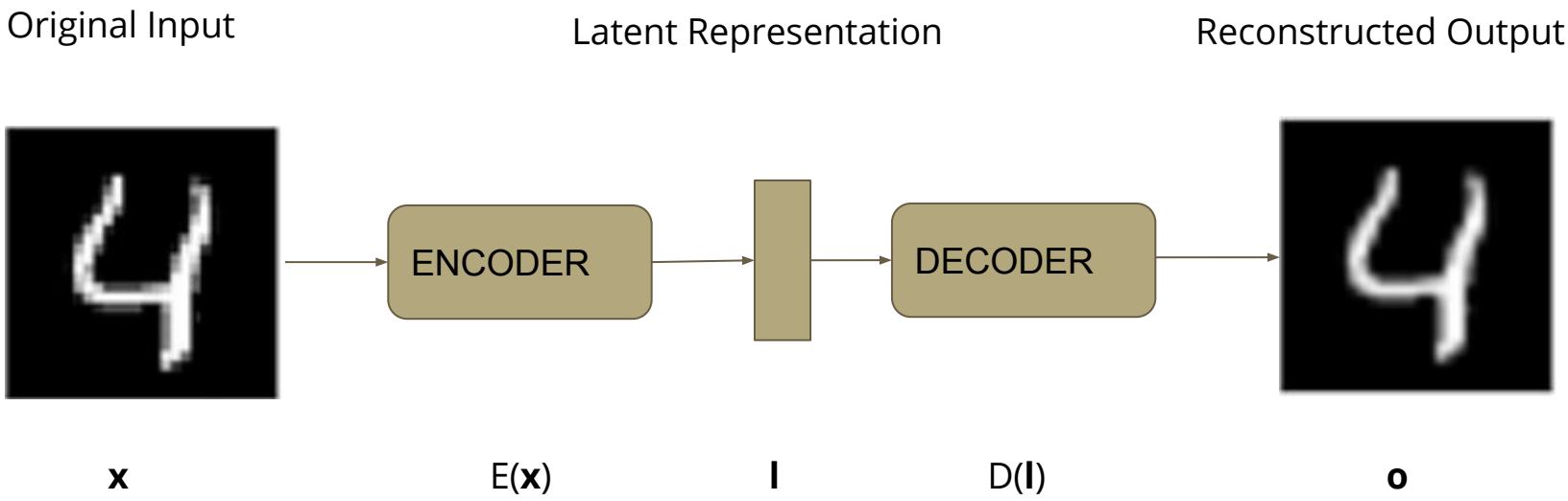
For regression, we had a fully-connected network, output layer size=1



Autoencoder



Autoencoder



Autoencoder - Model

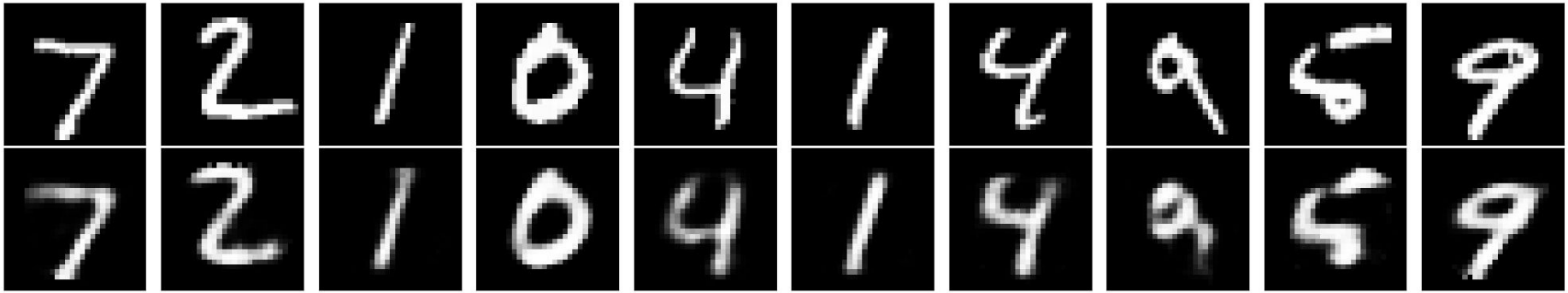
```
# build an autoencoder
model = tf.keras.models.Sequential([
    tf.keras.layers.InputLayer(IMG_SHAPE),
    tf.keras.layers.Flatten(),
    # encoder
    tf.keras.layers.Dense(128, activation='relu'),
    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dense(32, activation='relu'),
    # decoder
    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dense(128, activation='relu'),
    tf.keras.layers.Dense(784, activation='sigmoid'),
    tf.keras.layers.Reshape(IMG_SHAPE)
    ])

# compile
model.compile(optimizer='adamax', loss='mse')

# fit
model.fit(x_train, x_train, epochs=17,batch_size=256, shuffle=True, validation_data=(x_test, x_test))

# predict
decoded_imgs = model.predict(x_test)
```

Autoencoder - results



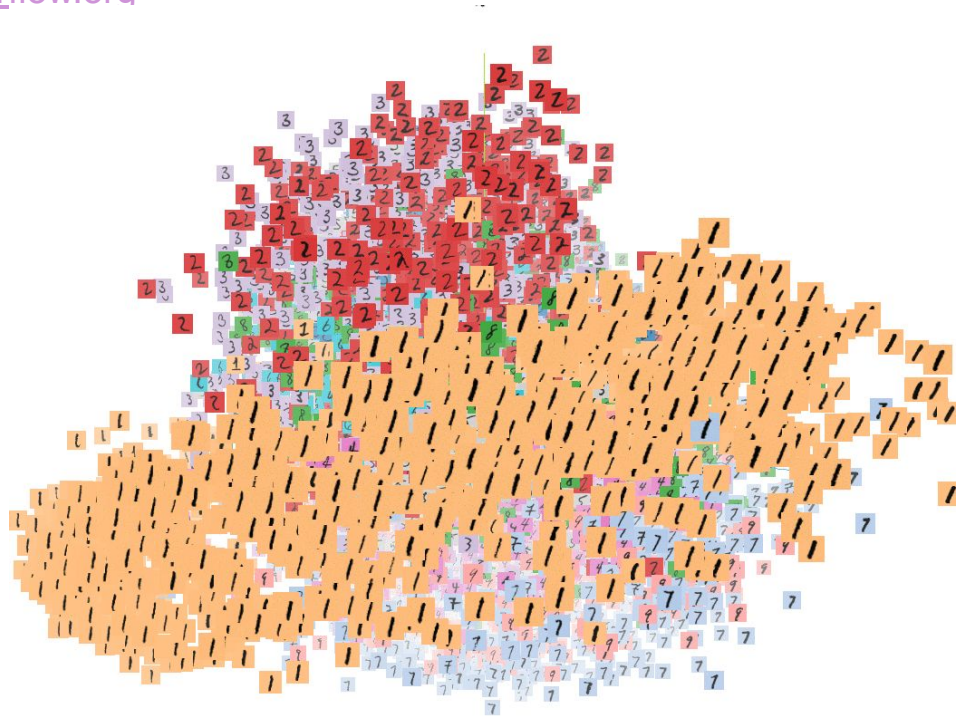
Compression Factor: 28x28/32 ~ 25X

Hands-on

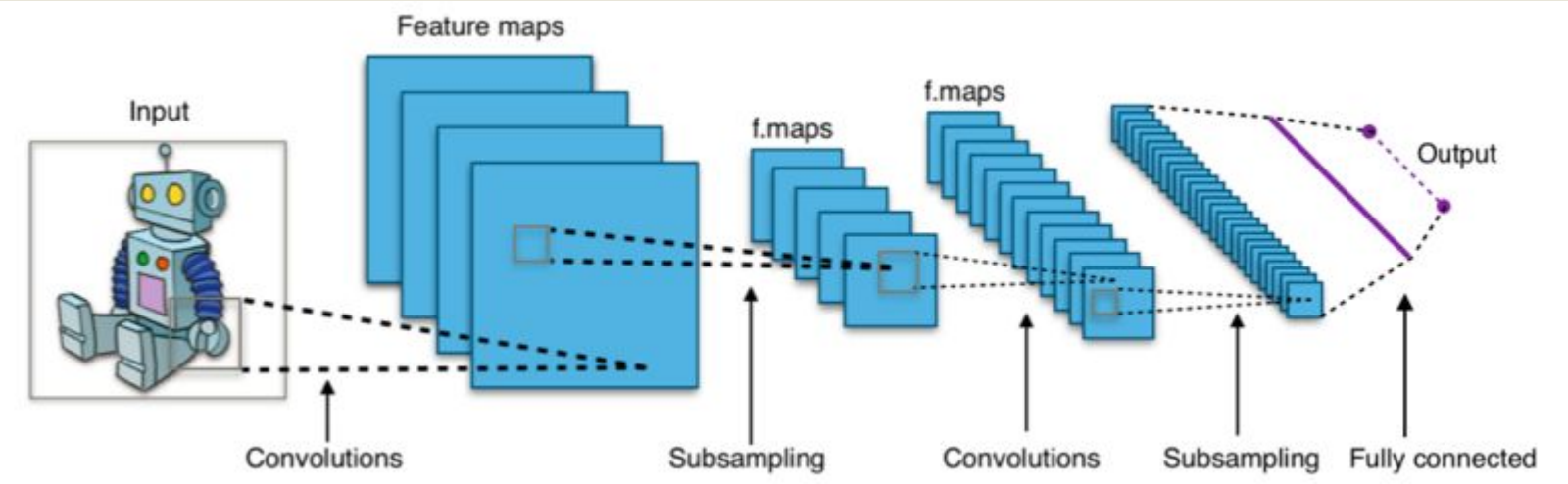
- ★ *Log in to your google drive*
- ★ Find the shared folder
- ★ Make a copy of:
  - AutoEncoder.ipynb

Latent Spaces and Embeddings

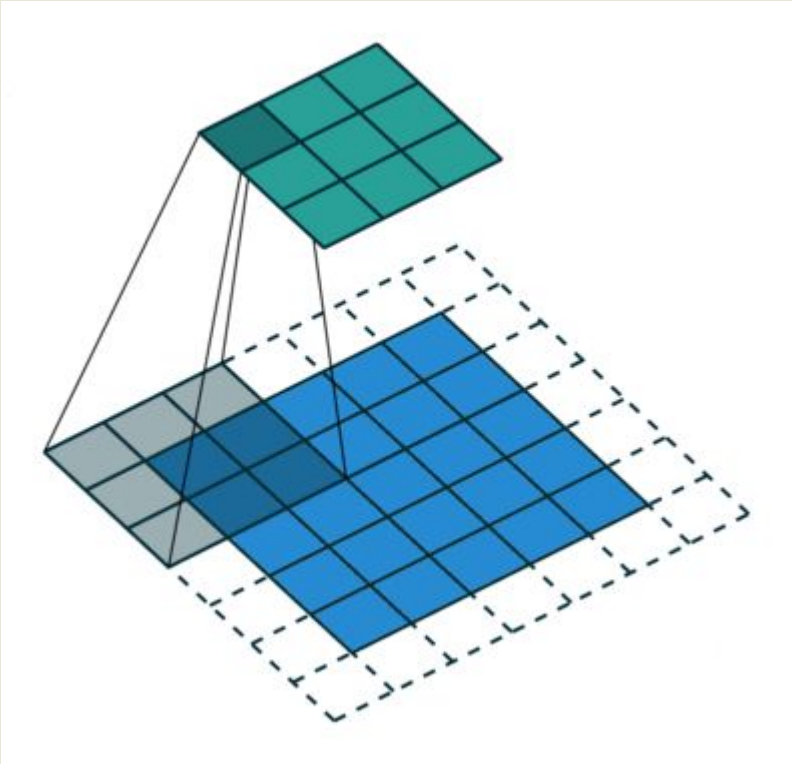
https://projector.tensorflow.org



Convolutional Neural Network (CNN)



Convolution (Extract High-Level Features)



Noisy image.....<similar image>.....Clean image

Denoising with Kernel Prediction and Asymmetric Loss Functions SIGGRAPH 2018, Vogels et al

Noisy image.....<similar image>.....Clean image

- If we have a set of noisy images and, a set of corresponding clean images,

- We can train our network to recover
  - Clean images from noisy images

- How
  - By setting Clean image as the ground truth,
  - the Noisy image as input and,
  - the loss function as the difference btwn the two

Noisy image...<similar image>...Clean image

- Take a look at denoiserCNN.ipynb
  - Make a CNN

~~Noisy~~ image.....<similar image>.....Clean image

degraded

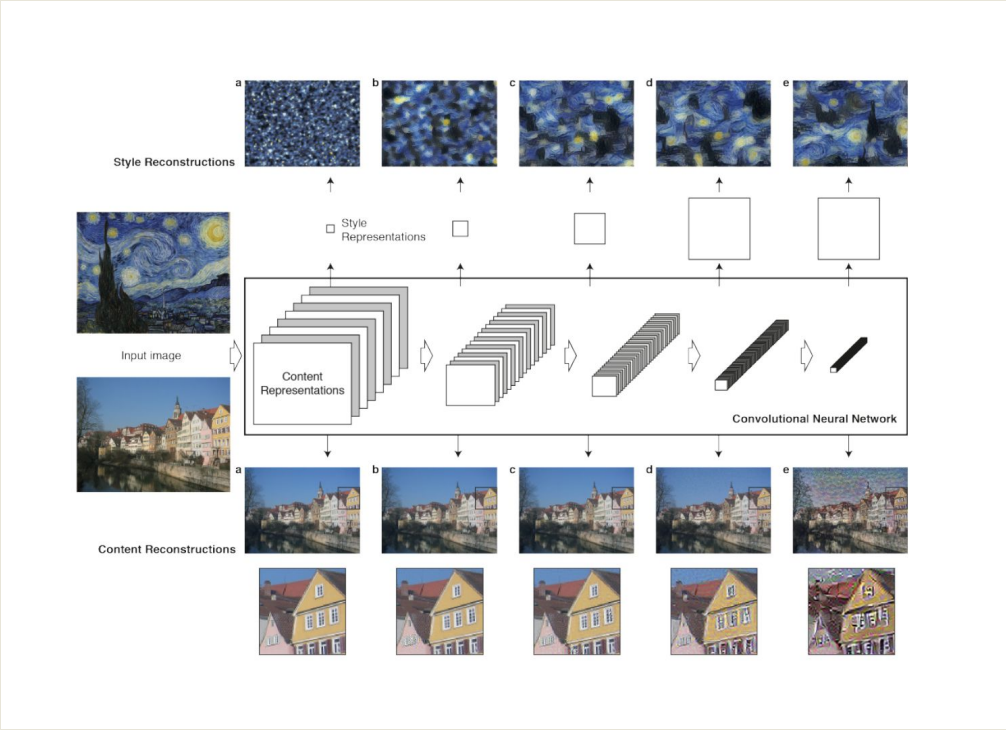
What else can we do?

Tint removal:	Image with tint.....<similar image>.....Clean image
In-painting:	Image with holes....<similar image>...Clean image
Dirt-removal:	Image with speckle....<similar image>...Clean image
Colorization:	Grayscale Image....<similar image>....Color image
Up-resing:	Lowres Image....<similar image>....Hires image
Inbetweening:	Image1, Image3....<similar image>.... Image2

Using off-the-shelf pretrained models

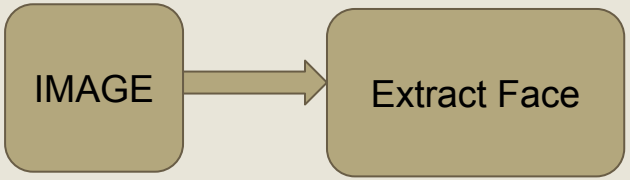
- Style Transfer
- MT-CNN

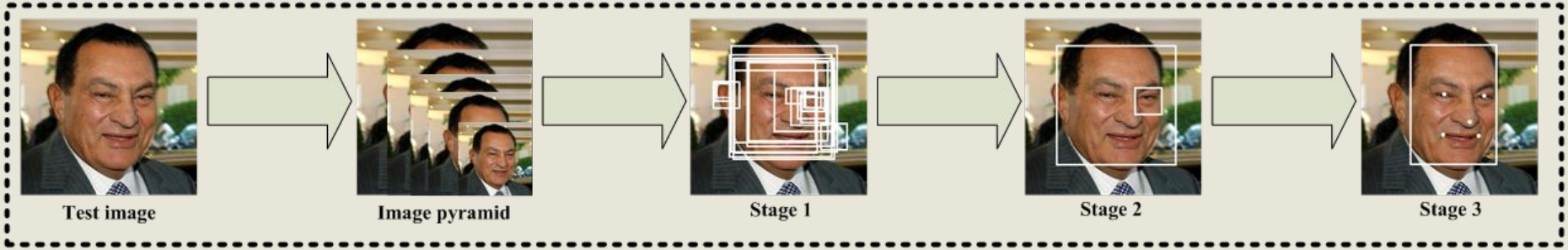
Artistic Style Transfer



$$L_{total}(\dot{c}, \dot{s}, \dot{x}) = \alpha L_{content}(\dot{c}, \dot{x}) + \beta L_{style}(\dot{s}, \dot{x})$$

Extracting Faces -- MT-CNN





Homework:

Colorization: `tf.image.adjust_saturation`

Up-resing:

```
tf.image.resize(image, size=[256,256], method=tf.image.ResizeMethod.NEAREST_NEIGHBOR)
```

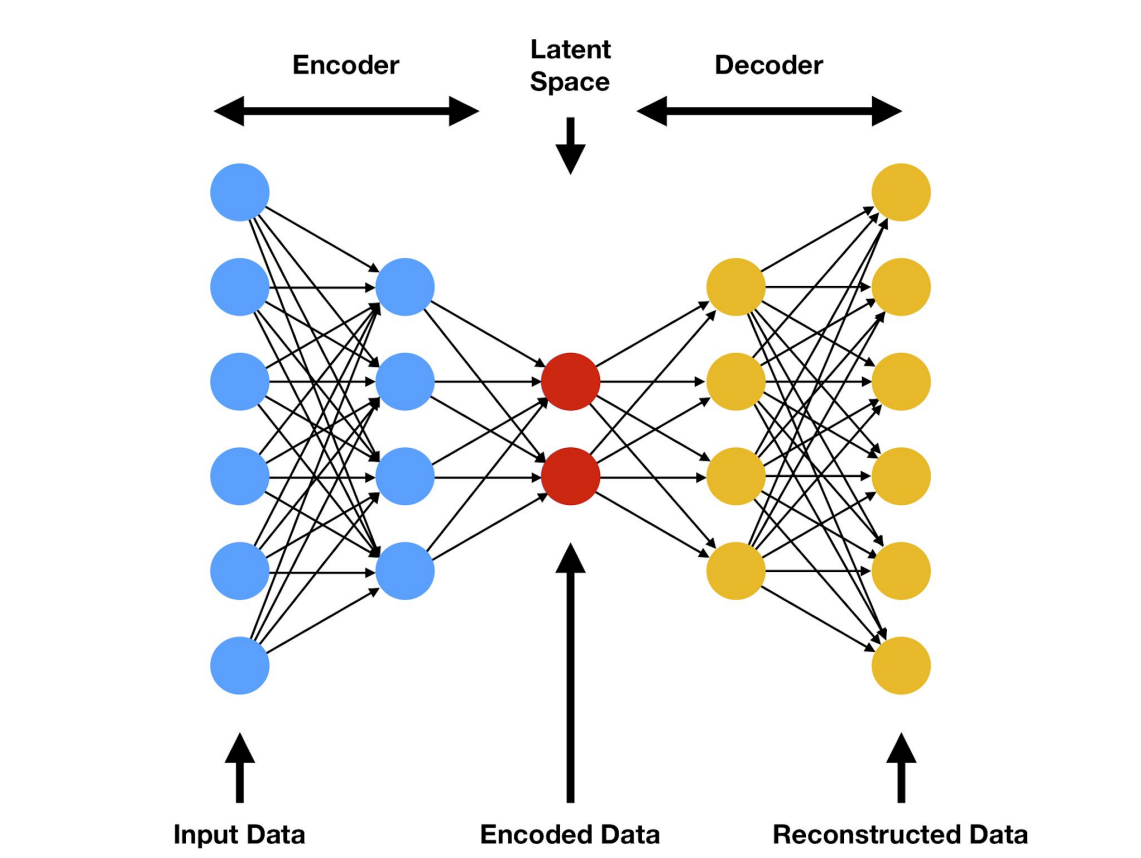
In-Painting:

```
mask = np.ones((PATCH_WIDTH, PATCH_HEIGHT), dtype=np.float32)
scale = 0.25
low, upper = int(PATCH_WIDTH * scale), int(PATCH_HEIGHT * (1.0 - scale))
mask[:, low:upper, low:upper] = 0.
tf.multiply(patch, mask)
```

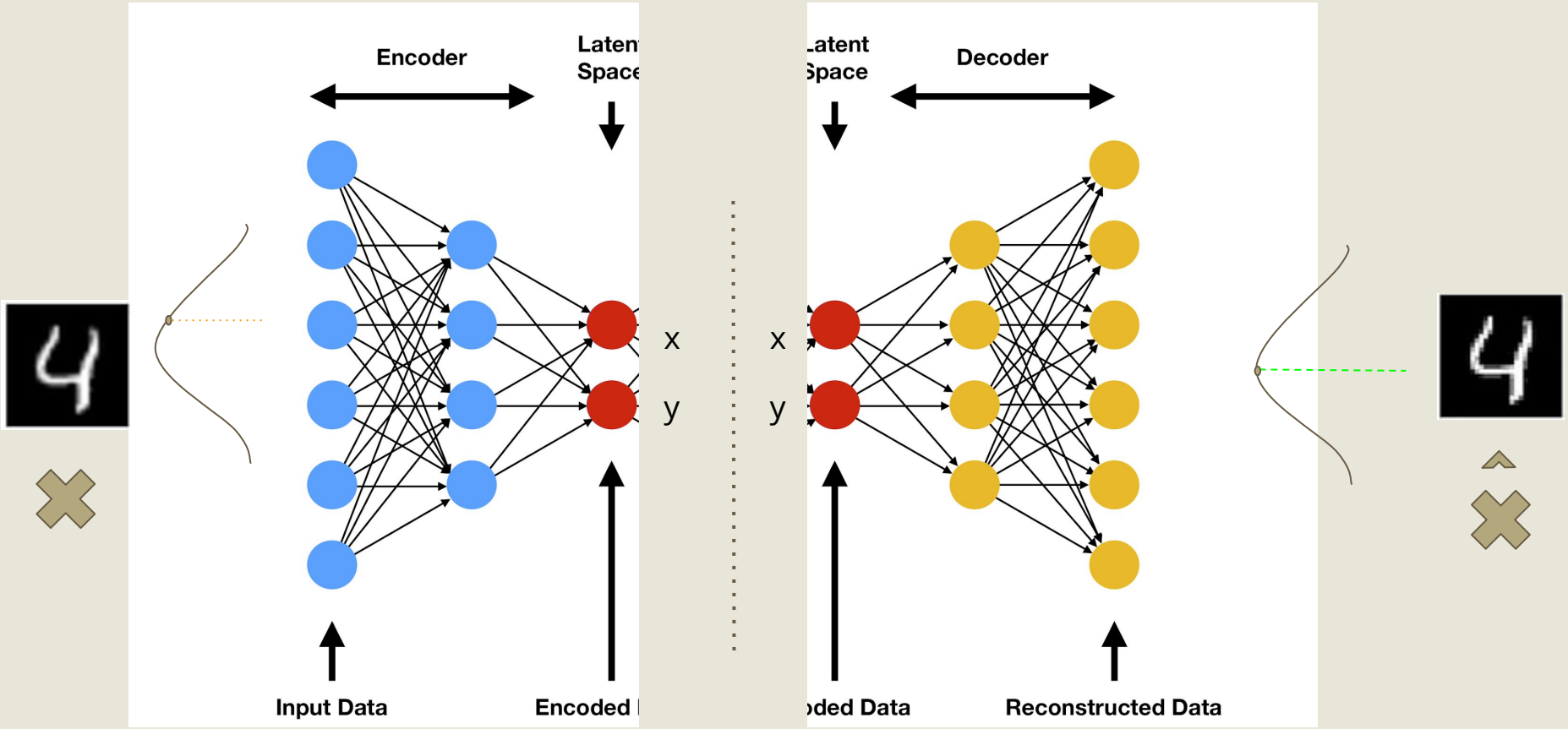
Frame interpolation:

```
stacked = tf.concat([frame1, frame3], axis=-1)
```

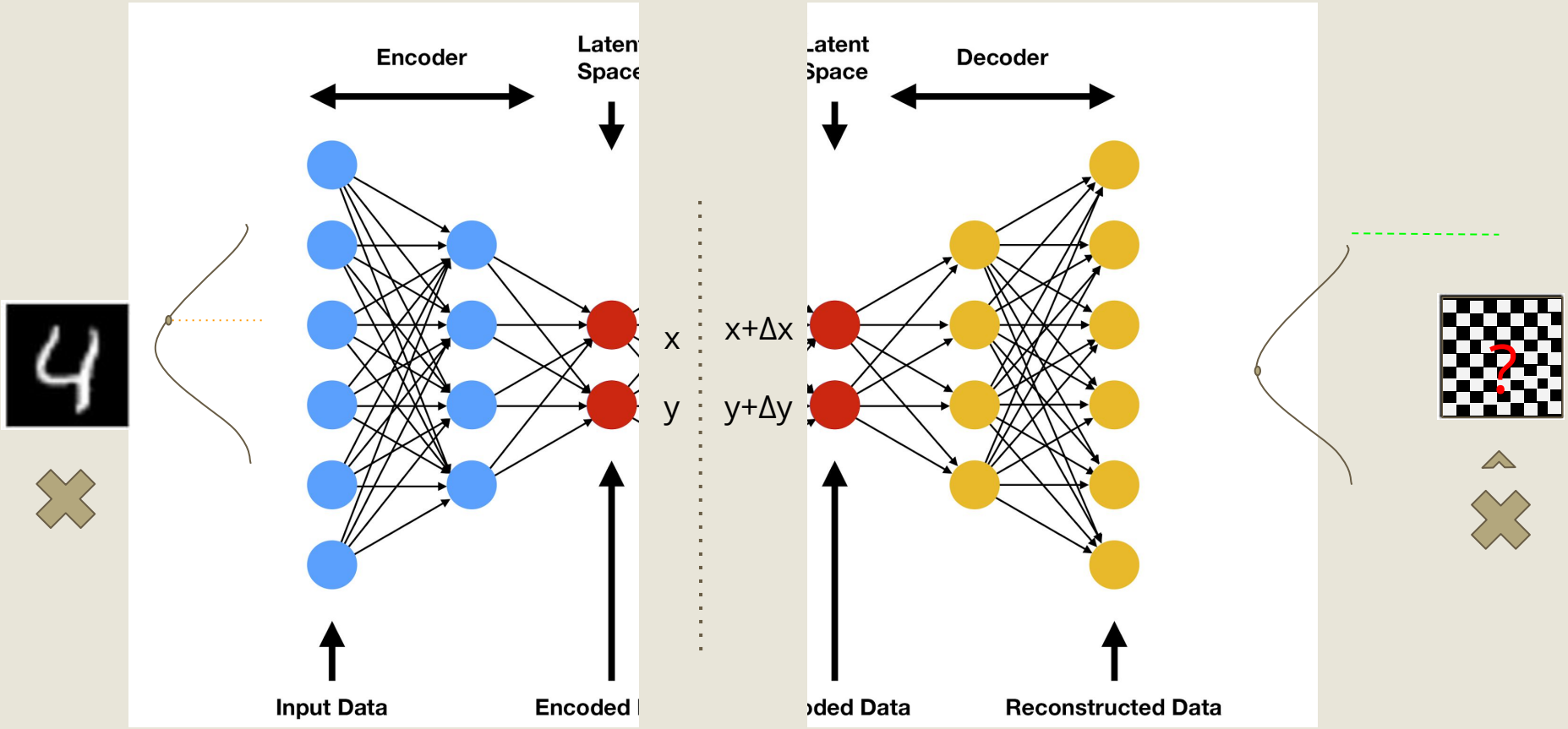
Autoencoder



Autoencoder - break it up after training



Autoencoder - A variation



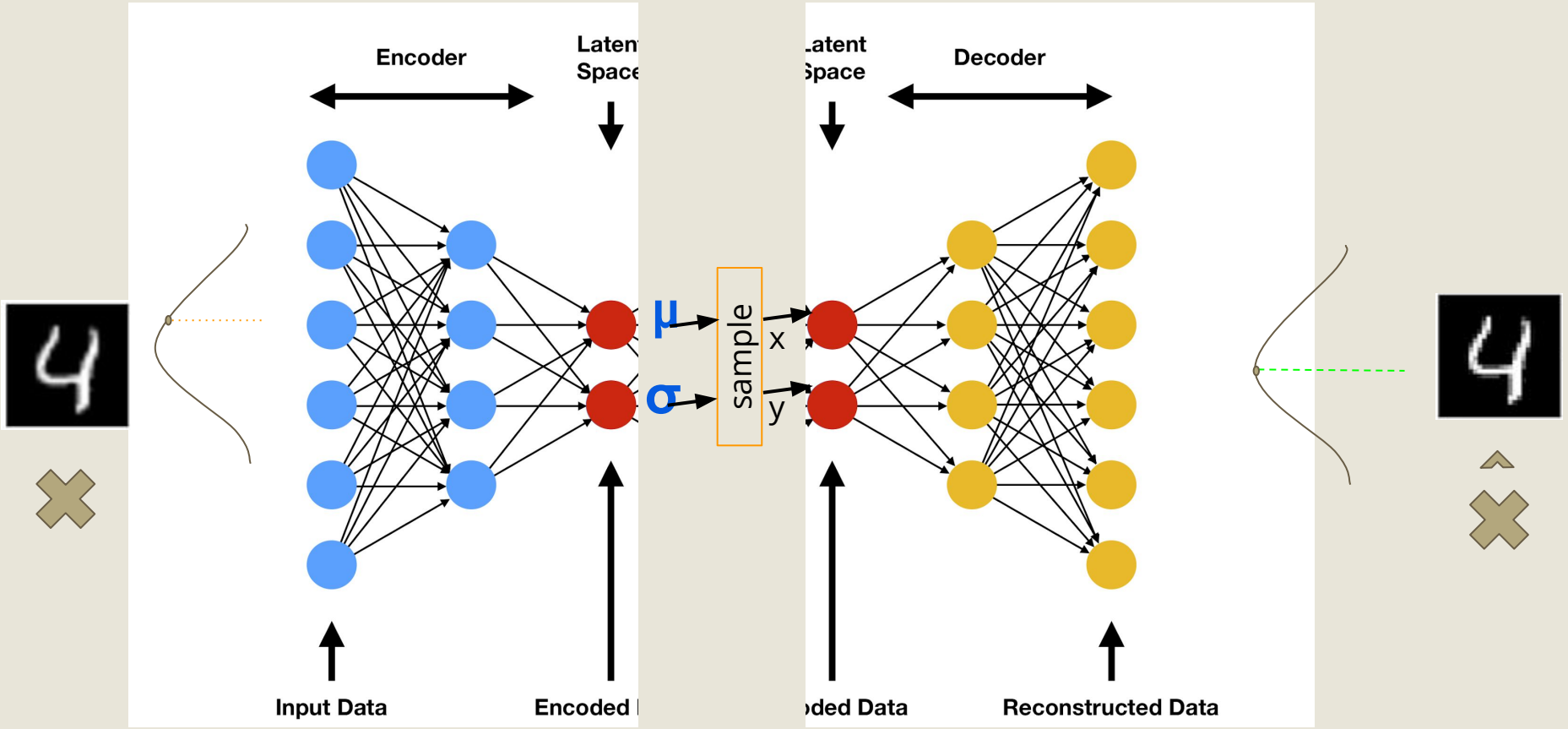
You don’t because…

The latent space and the input distributions are *different*!

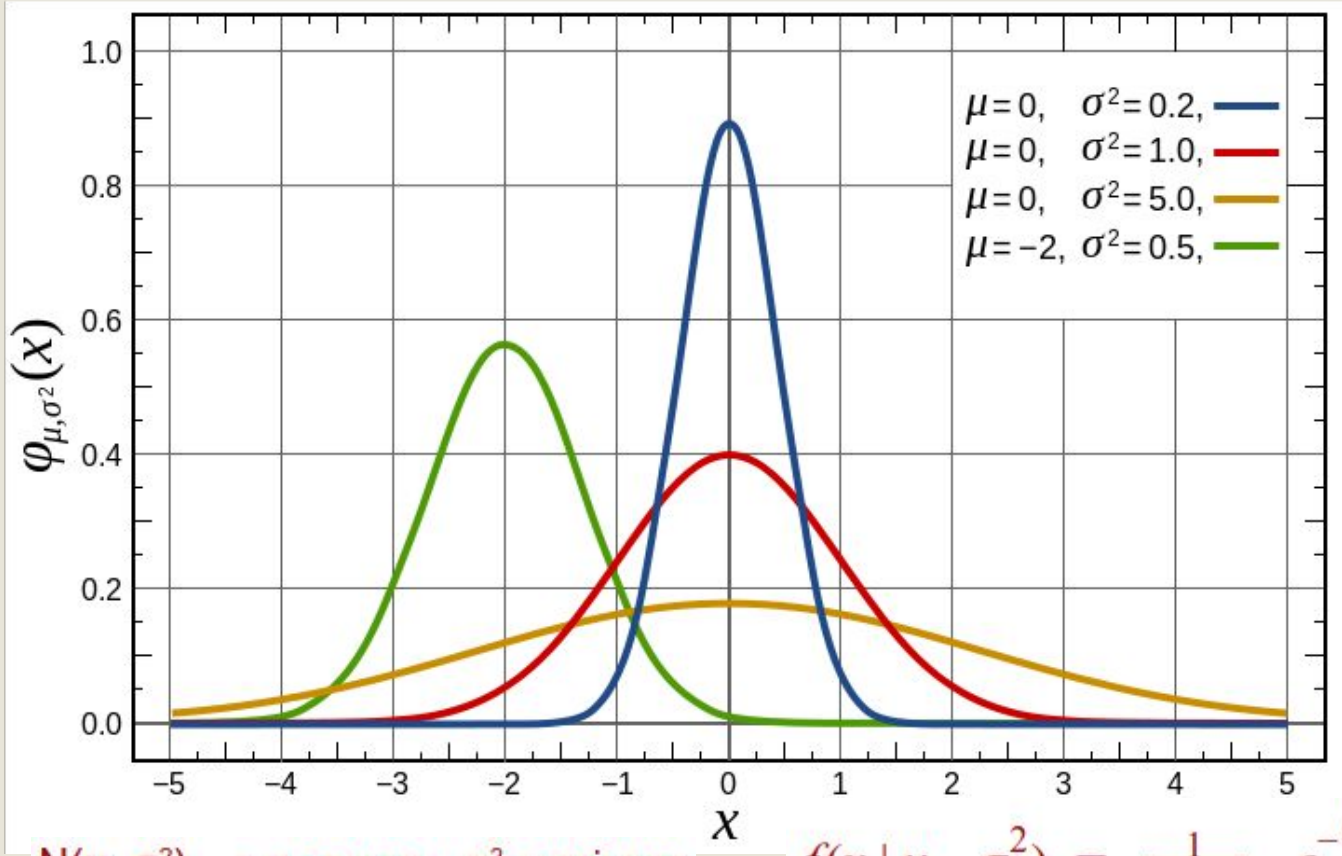
But there is a way:

Treat encoder output as μ and **σ** of a distribution

Variational Autoencoder



You get nice continuous distribution for each input



$N(\mu, \sigma^2)$; μ - mean , σ^2 - variance

$$f(x \mid \mu, \ \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \cdot e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

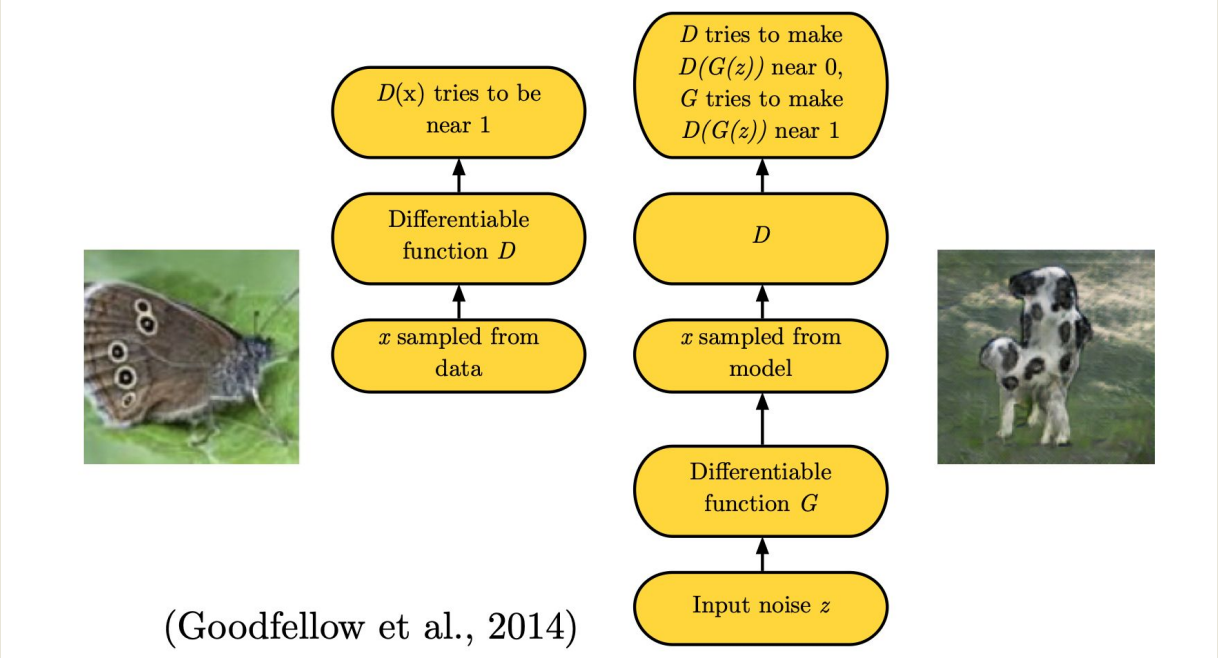
What are GANs?

GANs are similar to variational Autoencoder

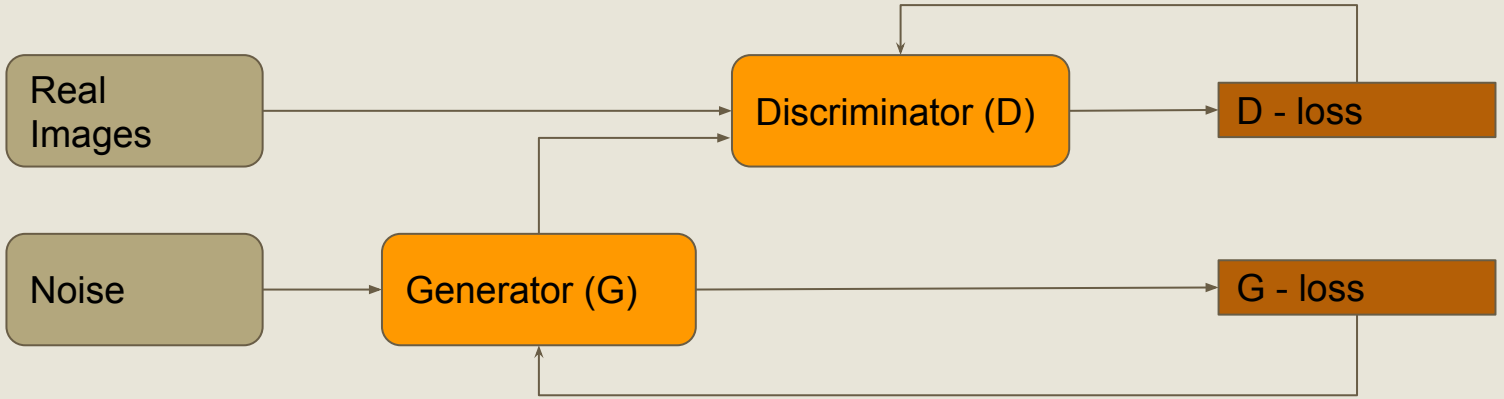
Instead of estimating the distribution (var, mean)

They try to sample from the distribution directly
by generating the sample from noise

GAN - Generative Adversarial Networks



GAN - Generative Adversarial Networks



GAN training proceeds in alternating periods:

1. The discriminator trains for one or more epochs.
2. The generator trains for one or more epochs.
3. Repeat steps 1 and 2 to continue to train the generator and discriminator networks.
4. Both the generator and the discriminator are neural networks.
5. The generator output is connected directly to the discriminator input.
6. Through backpropagation, the discriminator's classification is used by the generator to update its weights.

Easier to explain by building up an example

Goal: Generate random unseen cat images

Step1: **Need a classifier: cat or not**

Input: **labeled images of cat and not cats**

(we did this with iris flowers)

GAN…

Goal: Generate random unseen cat images

Step2: **Random image generator**

Input: **Noise**

GAN...

Goal: Generate random unseen cat images

Step3: Hook them up together!

Hands on...

Find and open:

`mnistGAN.ipynb`

Advanced Examples

GANimals

GauGAN

Next Class

- Facial Recognition Part II
- Exploring Latent Spaces
- Recurrent Neural Networks, Transformers
- Homework:
  - Send 2 pics
  - Using GAN, generate satellite images
- @xarmalarma, #siggraph2021