SIGGRAPH THINK BEYOND

2020 S2020.SIGGRAPH.ORG

WALT DISNEY ANIMATION STUDIOS

# JUNE 2020 WEBINAR

## Hands-on Workshop: Machine Learning and Neural Networks

SIGGRAPHNOW

**RAJESH SHARMA**

**SOFTWARE ENGINEER**

Walt Disney Animation Studios

6

# Machine Learning

Rajesh Sharma

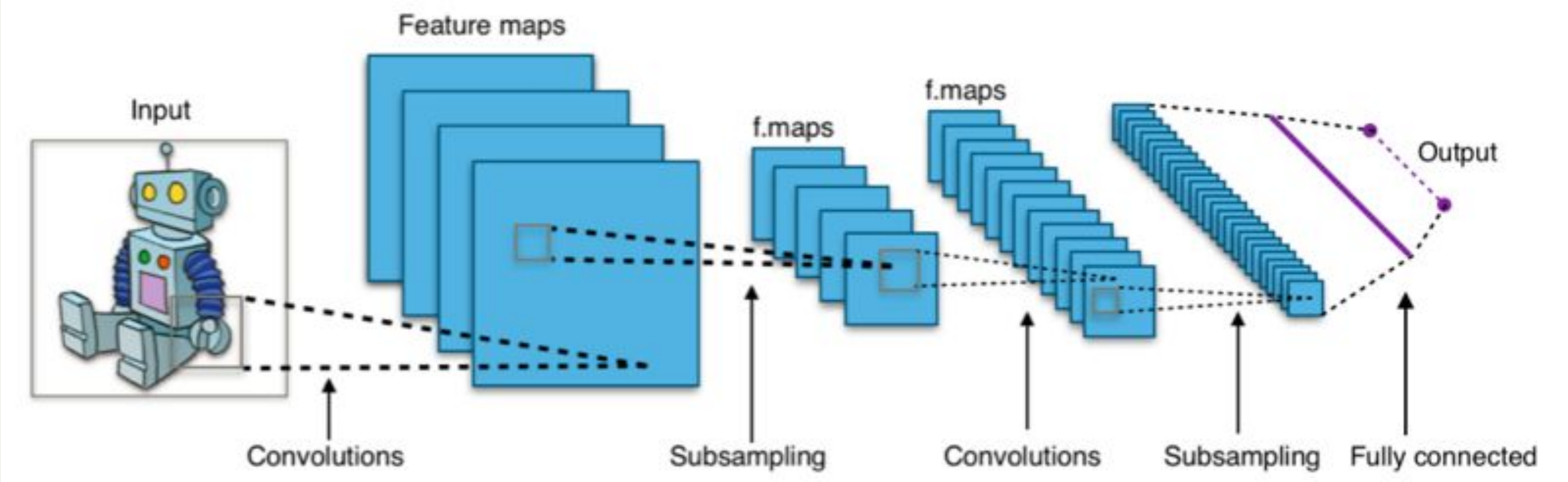
# Today

- Recap
- Transfer Learning
  - Building a Facial Recognition System

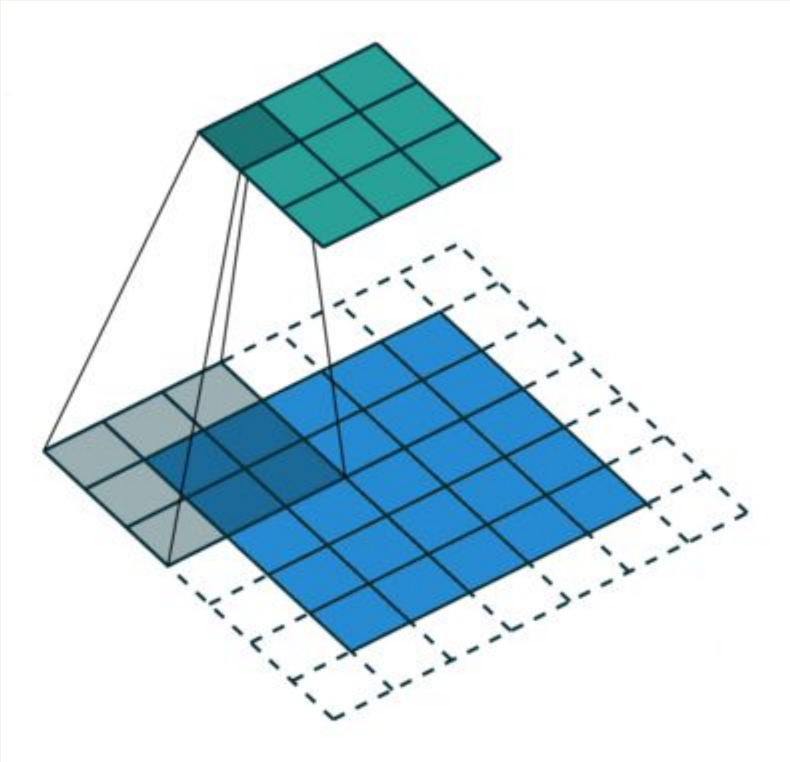
# Questions

| | |
|---|---|
| Is it a bug on line 7 of code where it says "patch_width=PATCH_HEIGHT"? ...not a problem if images are square, but otherwise could be... | Randi Rost |
| Are feature maps just multiple sets of patches? If yes, are the patch sizes different across feature maps for a given layer? | Ganesh Belgur Ramachandra |
| What strategies will you use to make sure that denoised images are temporally coherent when you have a sequence of images? | Esan Mandal |
| What is pooling and is it used here? | Alberto Grimaudo |
| Why the number of layer is three in this example? | Satoshi Nishimura |
| And why did you choose the values you chose? | Gabriel Zachmann |
| usually filter sizes are 3x3 or 5x5 or 7x7, but I never saw something like 4x4 ... why? | Vahe Vardanyan |
| Is the convolution filter given What is it? Laplace, edge detector, blurr, who decides it? | Anonymous Attendee |
| So the denoised image here is smaller than the original image? | Andrew Douglas |
| Why do the denoised images seem to have a greenish cast compared to the ground truth? | David Bollo |
| If you have a noisy image, but you don't have a ground truth, how can you denoise it? | Neel Sandell |

# Convolutional Neural Network (CNN)



# Convolution (Extract High-Level Features)



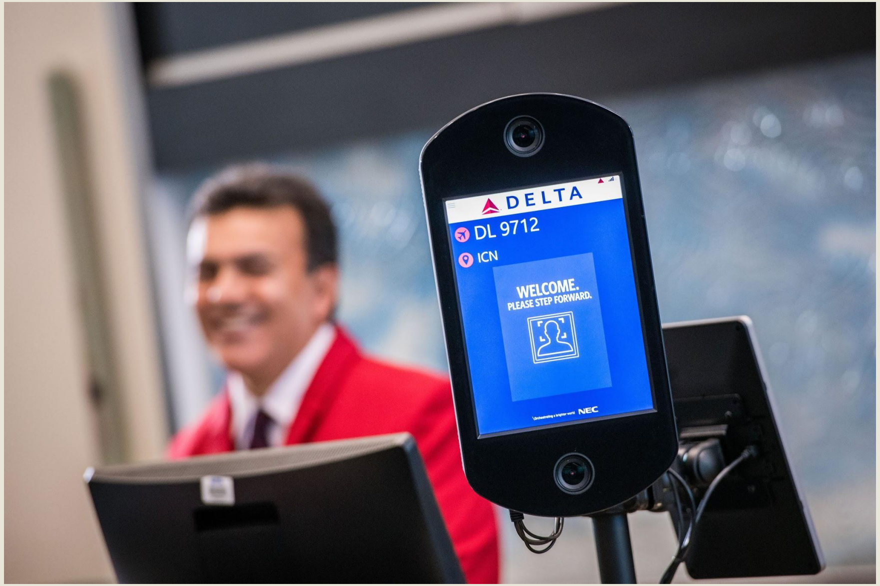
# Homework

```
def saturate(original, factor=1.5):
    saturated = tf.image.adjust_saturation(original, factor)
    # return both the saturated and the normal image
    tensor_tuple = (saturated, original)
    return tensor_tuple

def downres(original):
    scaled_down = tf.image.resize(original, size=[100,100], method=tf.image.ResizeMethod.NEAREST_NEIGHBOR)
    downres = tf.image.resize(scaled_down, size=[PATCH_HEIGHT,PATCH_WIDTH],
                                           method=tf.image.ResizeMethod.NEAREST_NEIGHBOR)
    # return both the downres'd and the normal image
    tensor_tuple = (downres, original)
    return tensor_tuple
```

# Transfer Learning

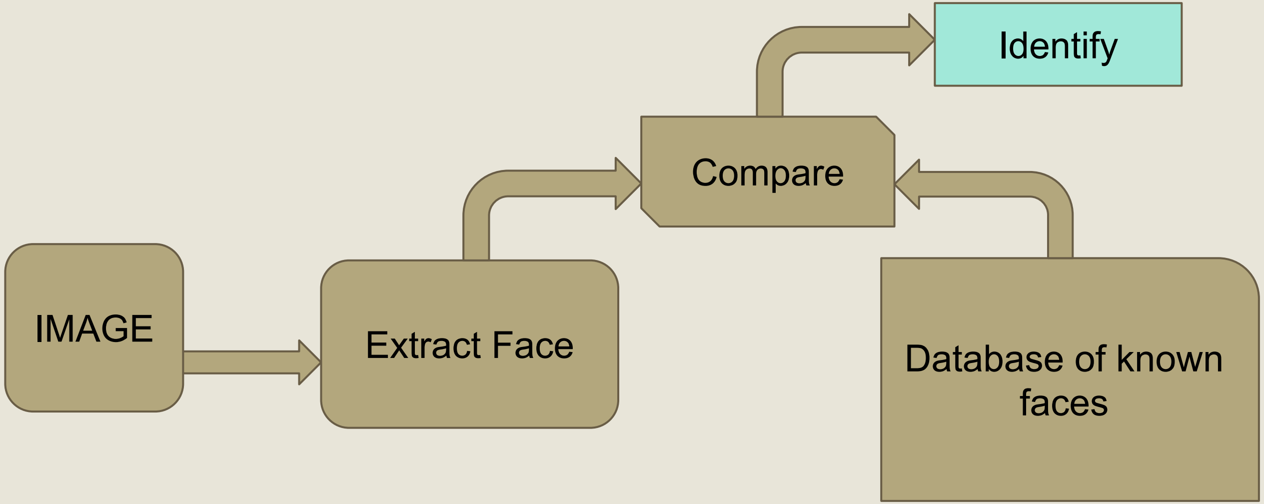
Build a Facial Recognition System




*Delta News Hub / CC BY (https://creativecommons.org/licenses/by/2.0)*

# Transfer Learning

## Build a Facial Recognition System



# End to End System - Transfer Learning

-- Read Camera Input Stream
-- Isolate Faces
-- Compare with stored ground truth
-- Identify person

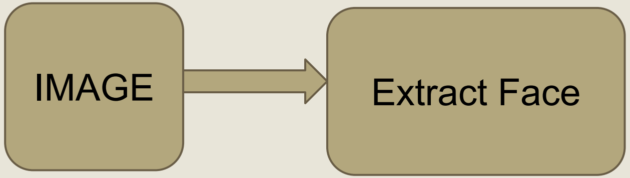
# Extracting Faces -- Haar Cascades

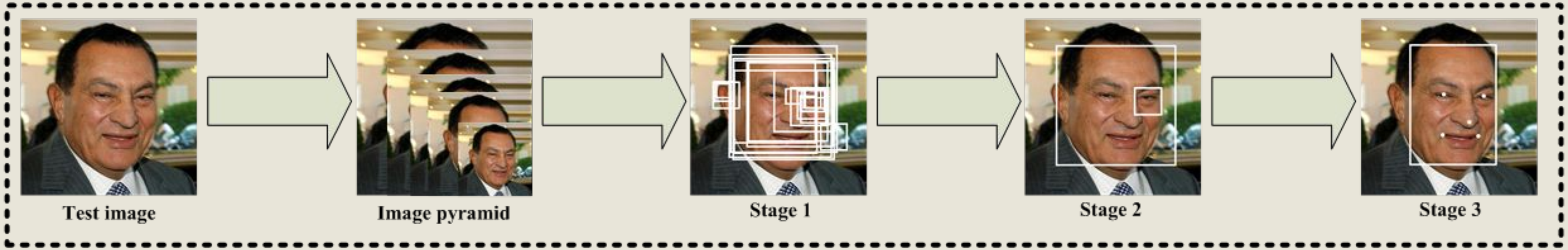
IMAGE → Extract Face

# End to End System - Transfer Learning

-- Use other people's trained network
-- Add in your own data

# Extracting Faces -- MT-CNN



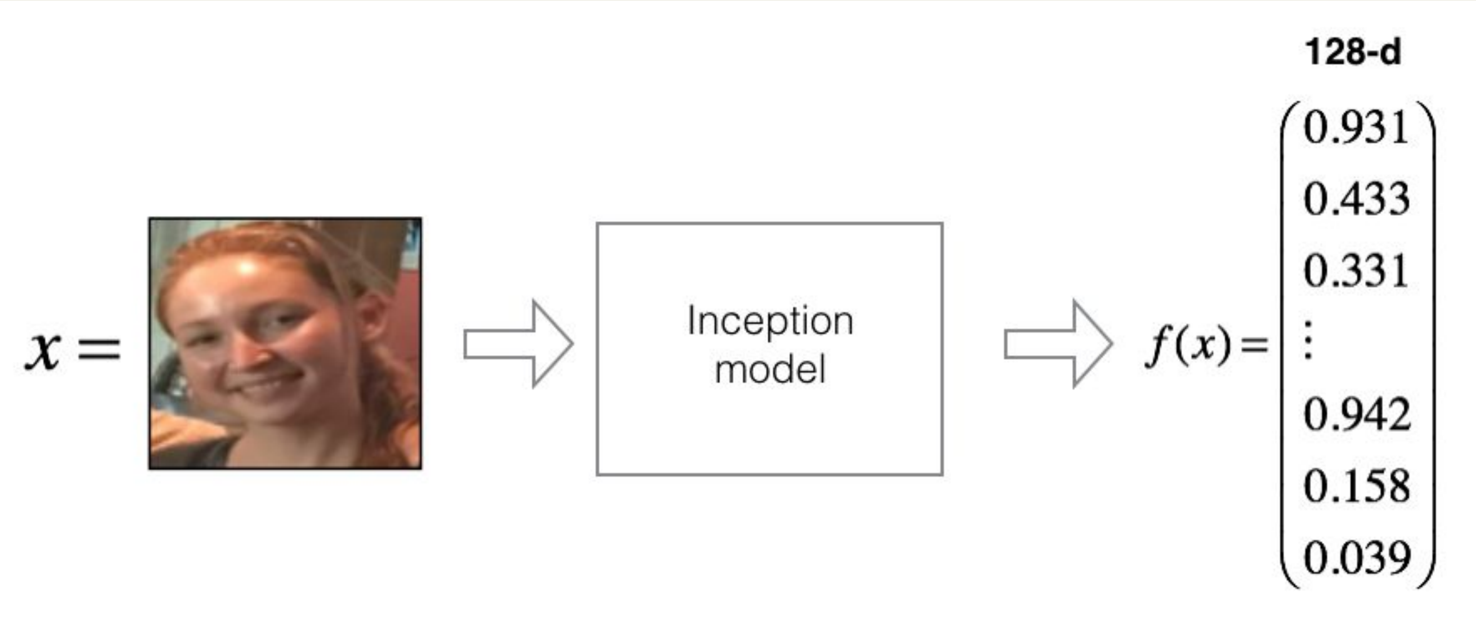
# Hands-on

★ ***facialRecognition01.ipynb***

# End to End System - Transfer Learning

1. Get faces from ground-truth images (MTCNN)
2. Encode ground-truth images (FACENET)
3. Read Camera Input Stream
4. Isolate Faces (MTCNN)
5. Encode input face (FACENET)
6. Compare encoding with stored ground-truth
7. Identify person

# Facenet - triplet loss: Paper

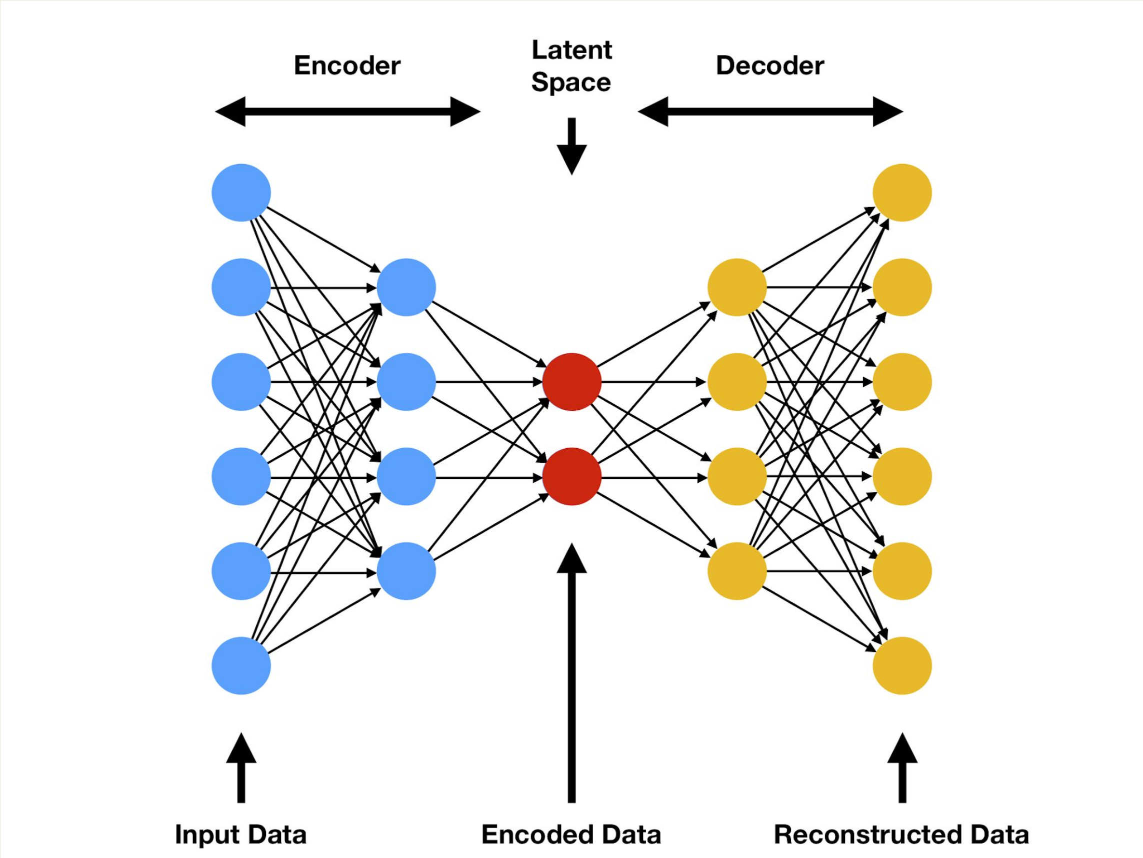



$$Loss = \sum_{i=1}^{N} \left[ \|f_i^a - f_i^p\|_2^2 - \|f_i^a - f_i^n\|_2^2 + \alpha \right]_+$$

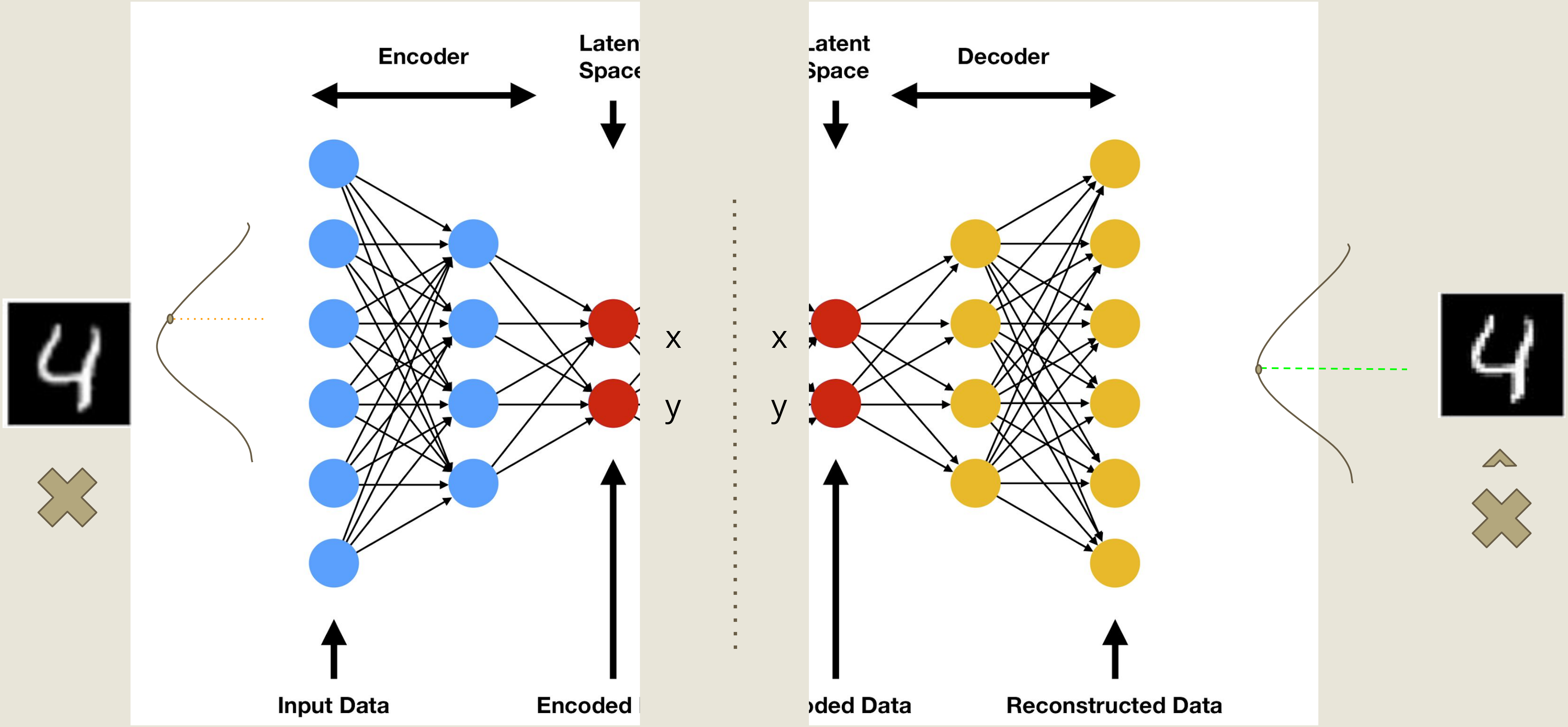
# Hands-on

- ***facialRecognition02.ipynb***

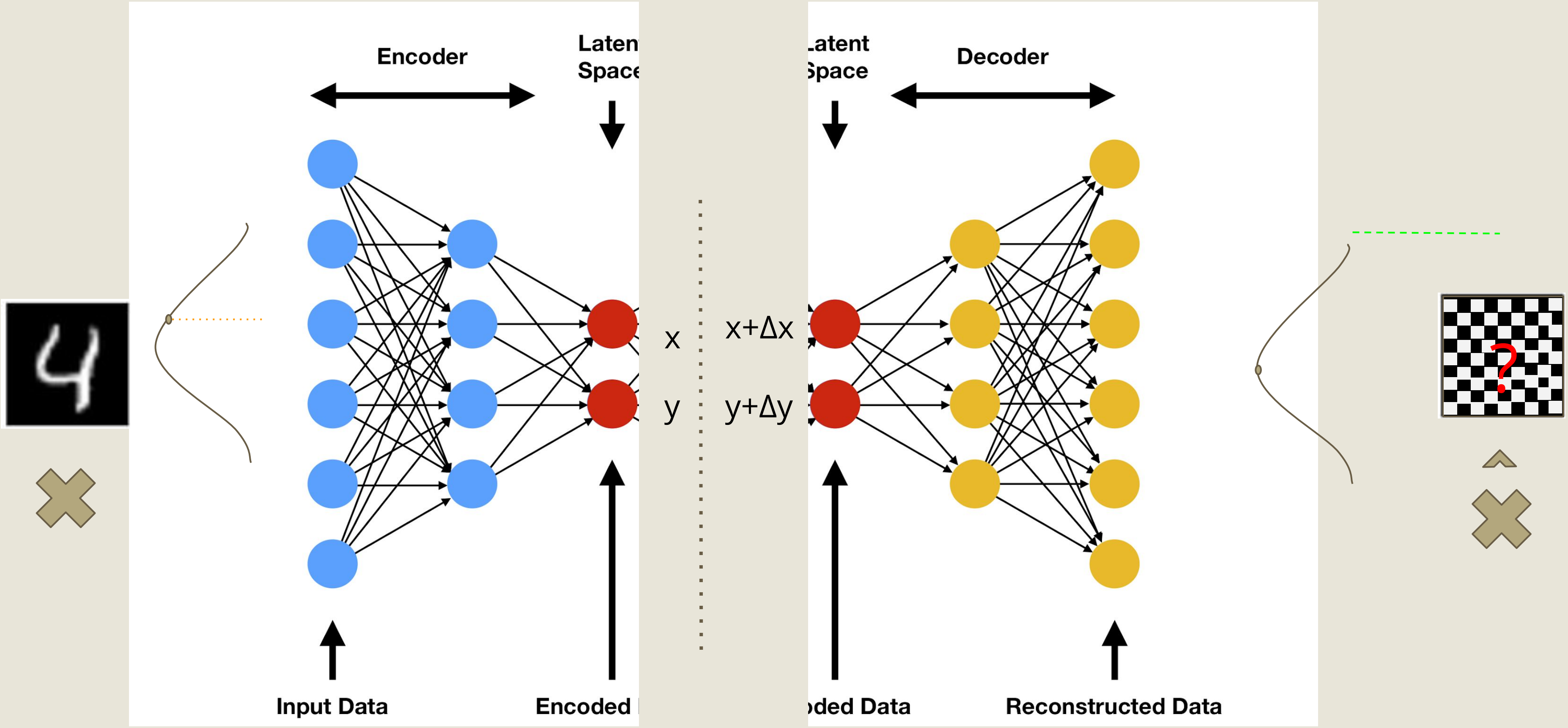
# Autoencoder



# Autoencoder



# Autoencoder - A variation



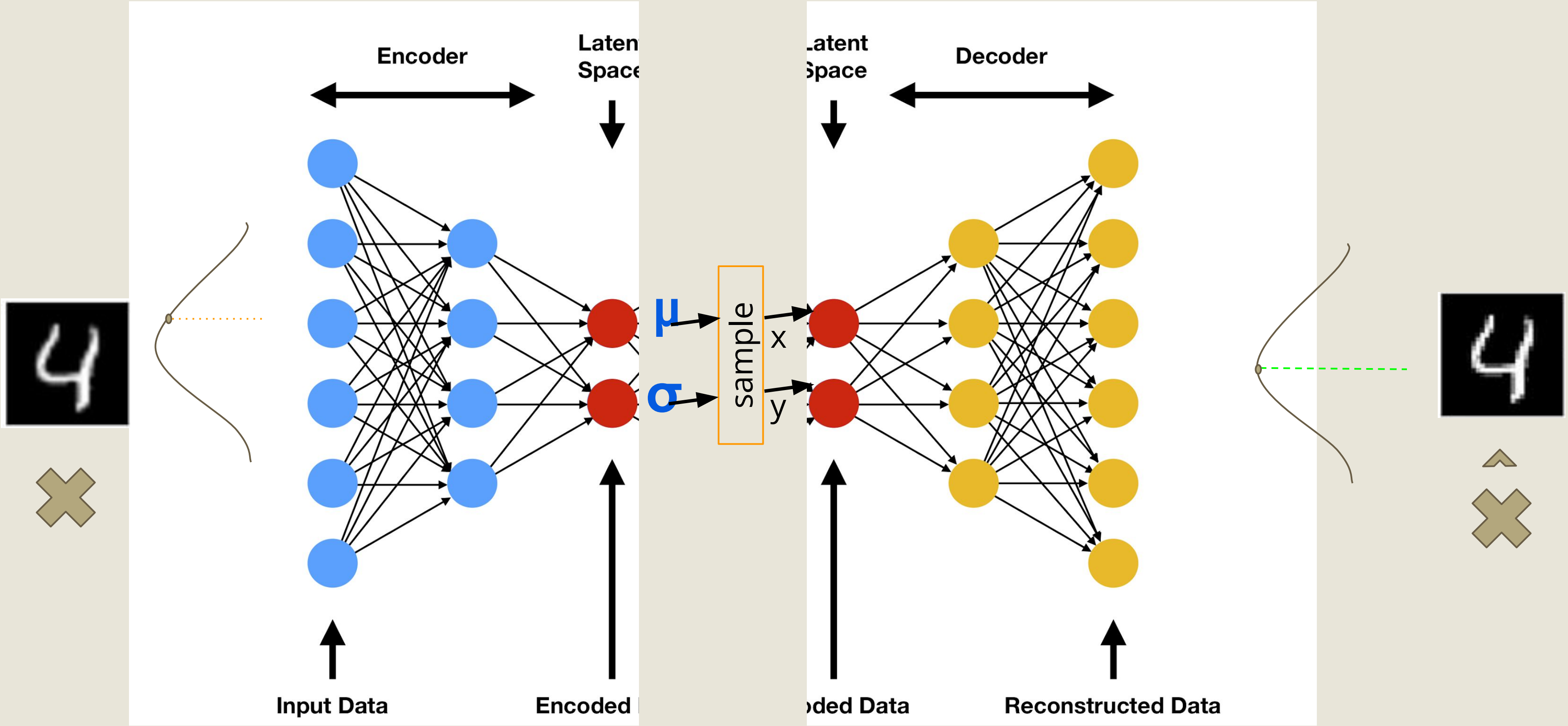
# You don't because

The latent space and the input distributions are different!

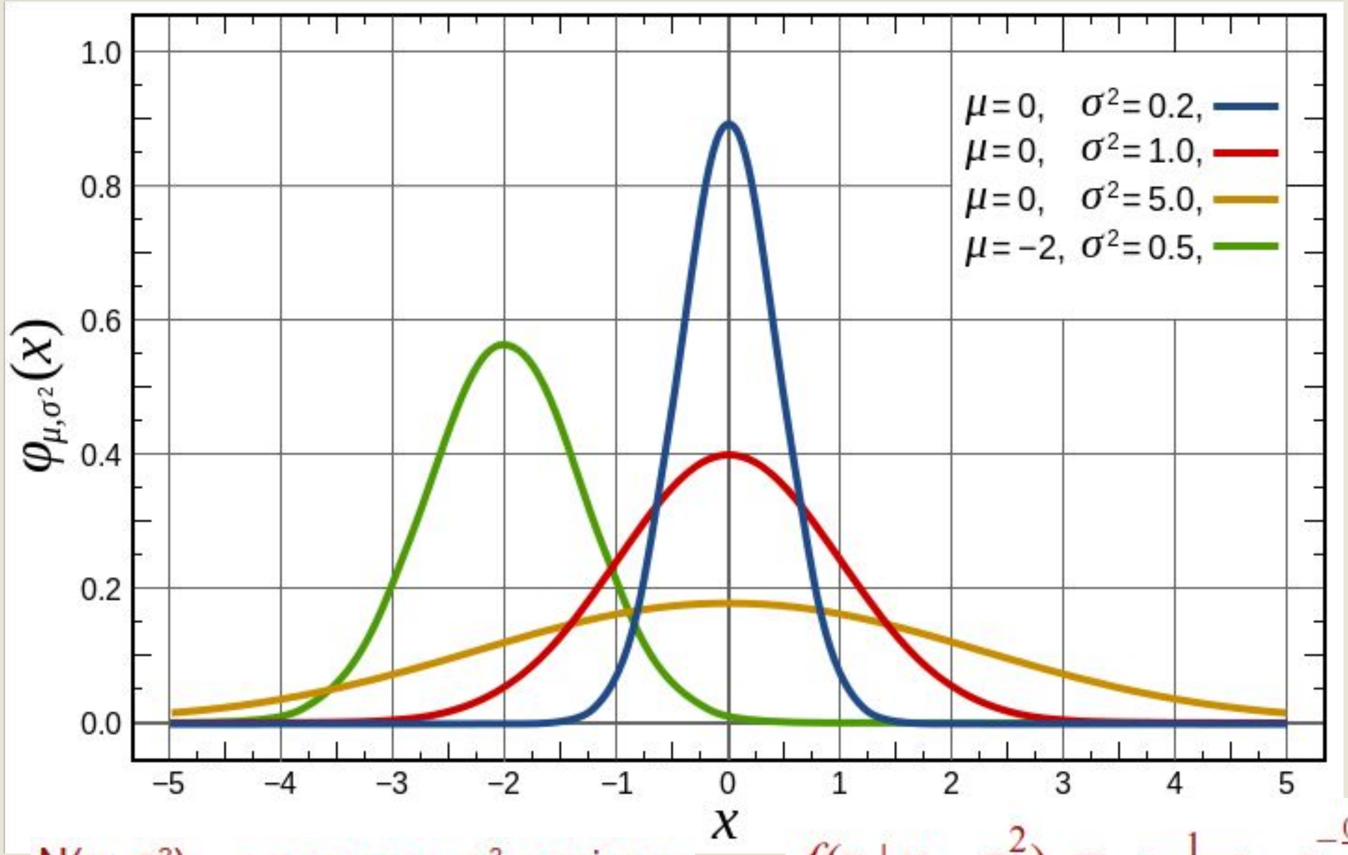
But there is a way:

Treat x,y as μ and σ of a distribution

# Variational Autoencoder



# You get nice continuous distribution for each input




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

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

# Latent Spaces and Embeddings

https://projector.tensorflow.org

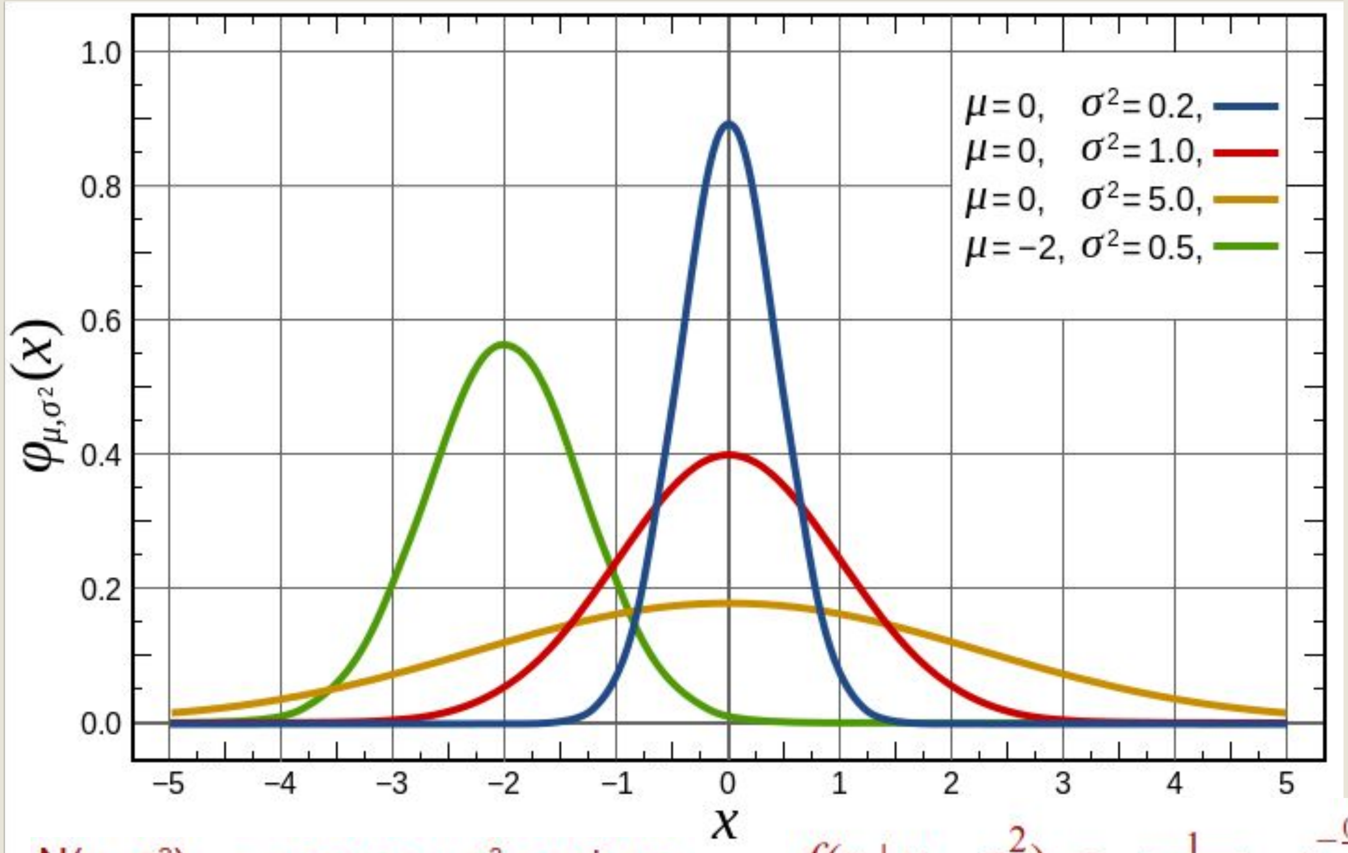
For each input we can generate new ‘fake’ output!

Moreover, we can interpolate!

but, we can do even better!

What about the distributions for other inputs?

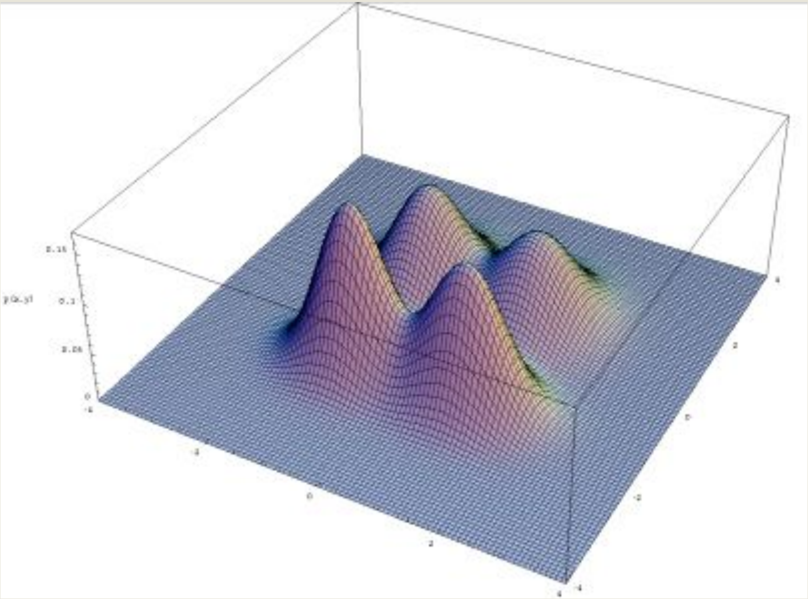
# What if we could take this and turn it into




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

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

# What if we could take this and turn it into



# A continuous multi-modal distribution!

# We can then interpolate between two (or more) inputs

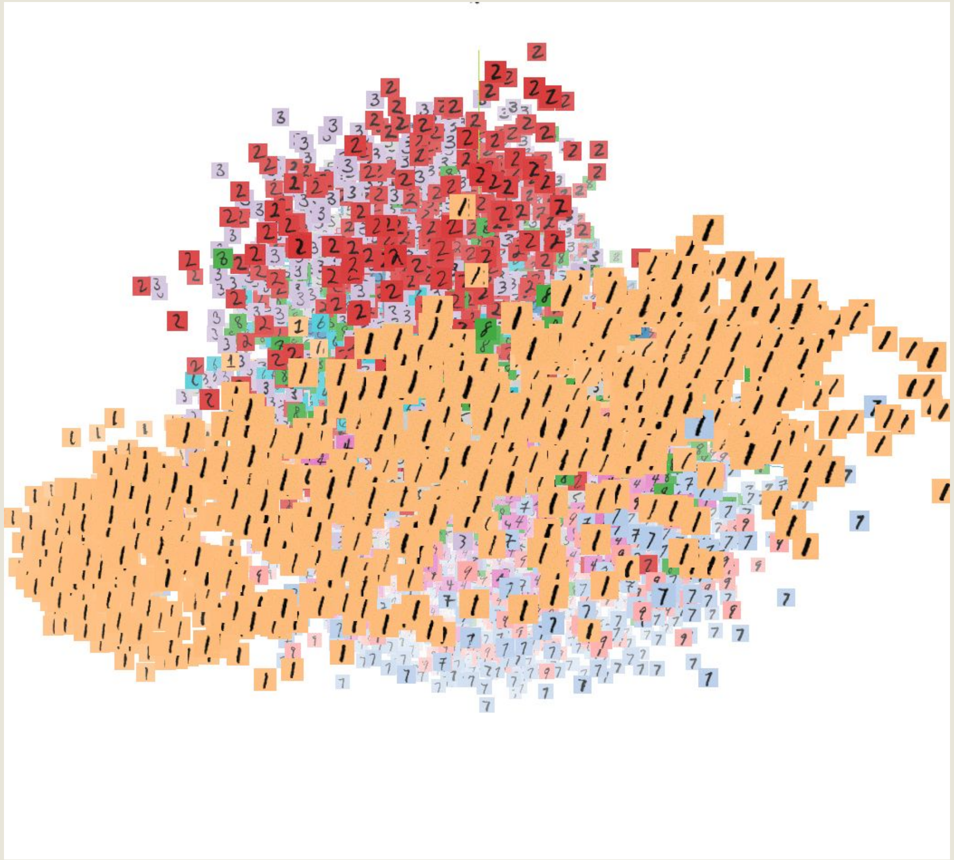
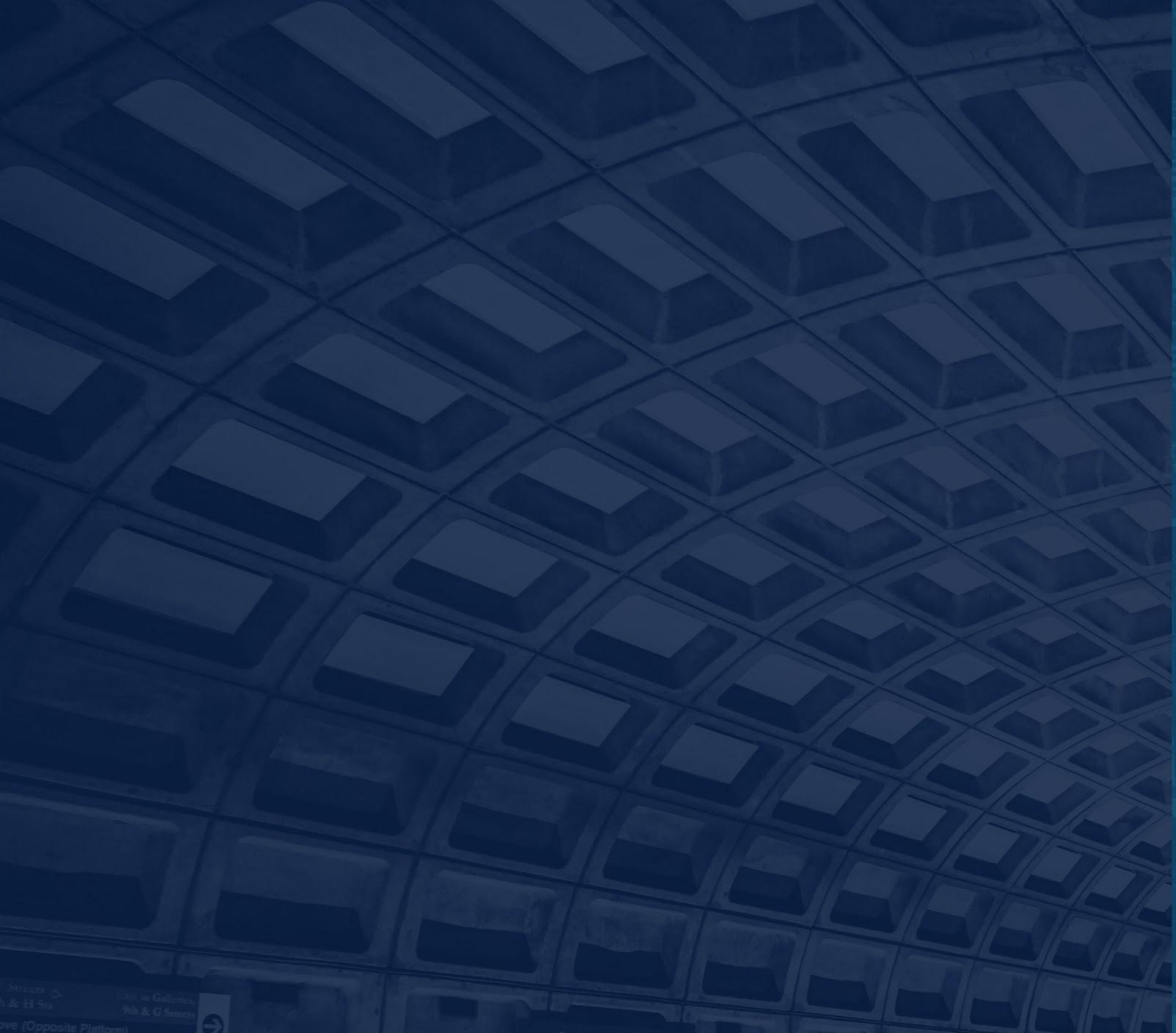


*Figure 6.* Interpolation experiments for celebA

# Next Class

- Finish facial recognition
- Generative Networks: Var-AE vis, GAN
- Homework:
  - Get celebrity images, find encodings
  - Find out who looks like you
- @xarmalarma, #siggraphNOW





# QUESTIONS?

Submit now!

# THANK YOU

## *WANT TO HEAR MORE FROM SIGGRAPH AND DISNEY?*

**Check out SIGGRAPH Now on YouTube**

YouTube.com/user/ACMSIGGRAPH

**Subscribe to the SIGGRAPH Spotlight podcast**

SIGGRAPH SPOTLIGHT

STITCHER

iHeart PODCAST NETWORK

SOUNDCLOUD