



Autoencoder

Prof. Seungchul Lee
Industrial AI Lab.

Autoencoders

- It is like 'deep learning version' of unsupervised learning
- Definition
 - An autoencoder is a neural network that is trained to attempt to copy its input to its output
 - The network consists of two parts: an encoder and a decoder that produce a reconstruction
- Encoder and Decoder
 - Encoder function : $z = f(x)$
 - Decoder function : $x = g(z)$
 - We learn to set $g(f(x)) = x$

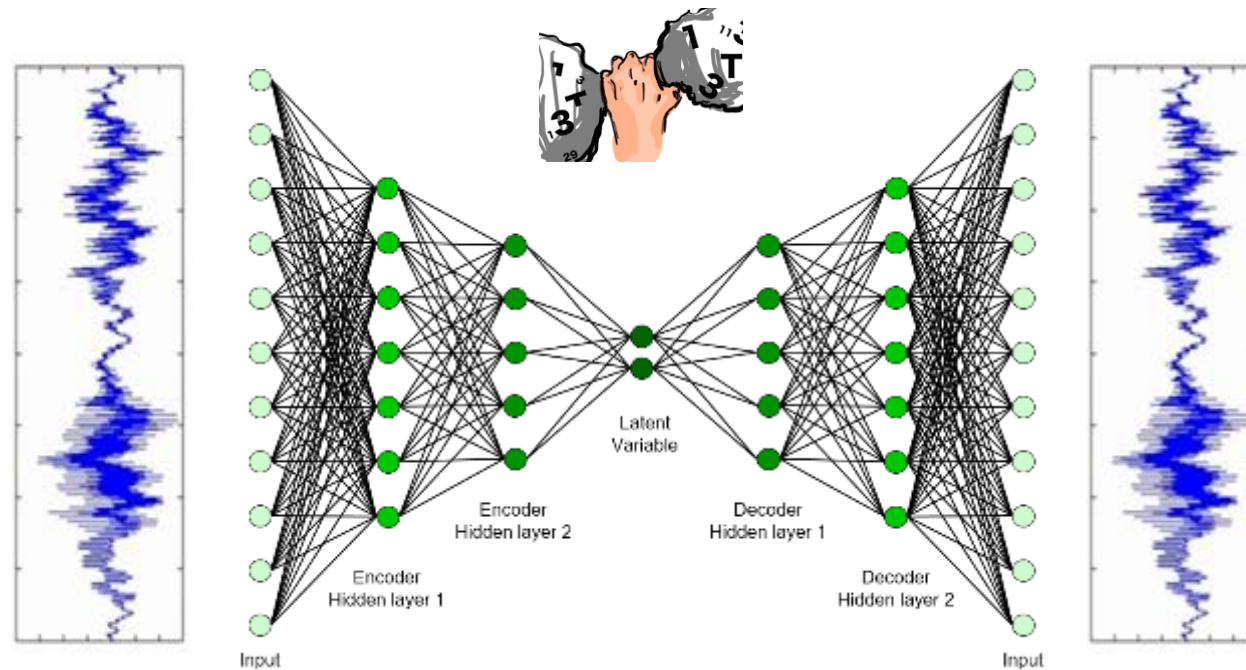
Autoencoder

- Dimension reduction
- Recover the input data



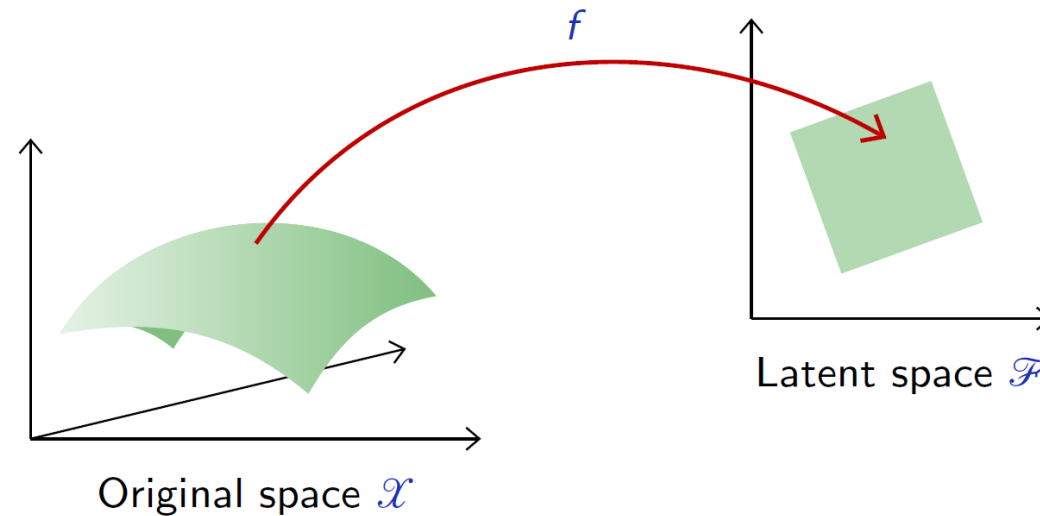
Autoencoder

- Dimension reduction
- Recover the input data
 - Learns an encoding of the inputs so as to recover the original input from the encodings as well as possible



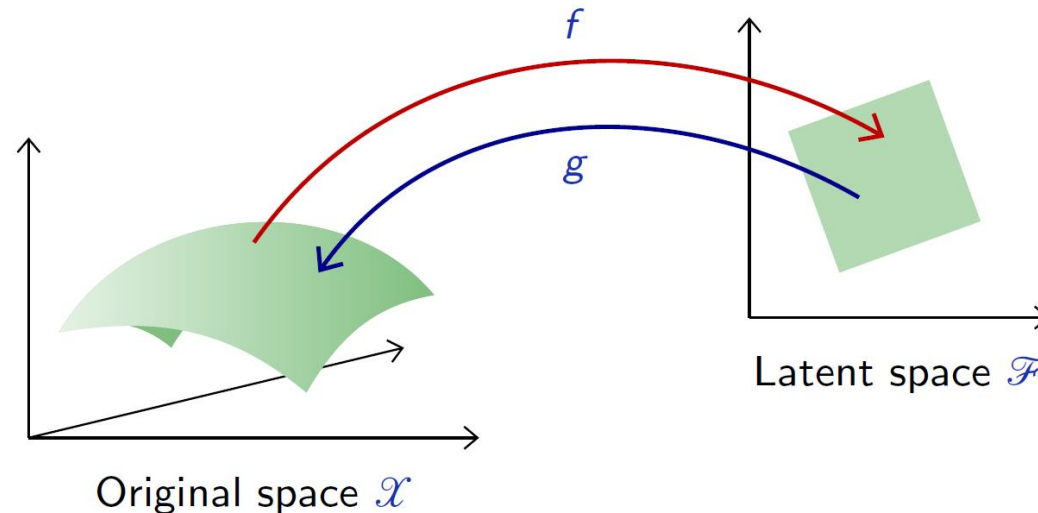
Autoencoder

- Autoencoder combines an encoder f from the original space \mathcal{X} to a latent space \mathcal{F} , and a decoder g to map back to \mathcal{X} , such that $g \circ f$ is [close to] the identity on the data



Autoencoder

- Autoencoder combines an encoder f from the original space \mathcal{X} to a latent space \mathcal{F} , and a decoder g to map back to \mathcal{X} , such that $g \circ f$ is [close to] the identity on the data

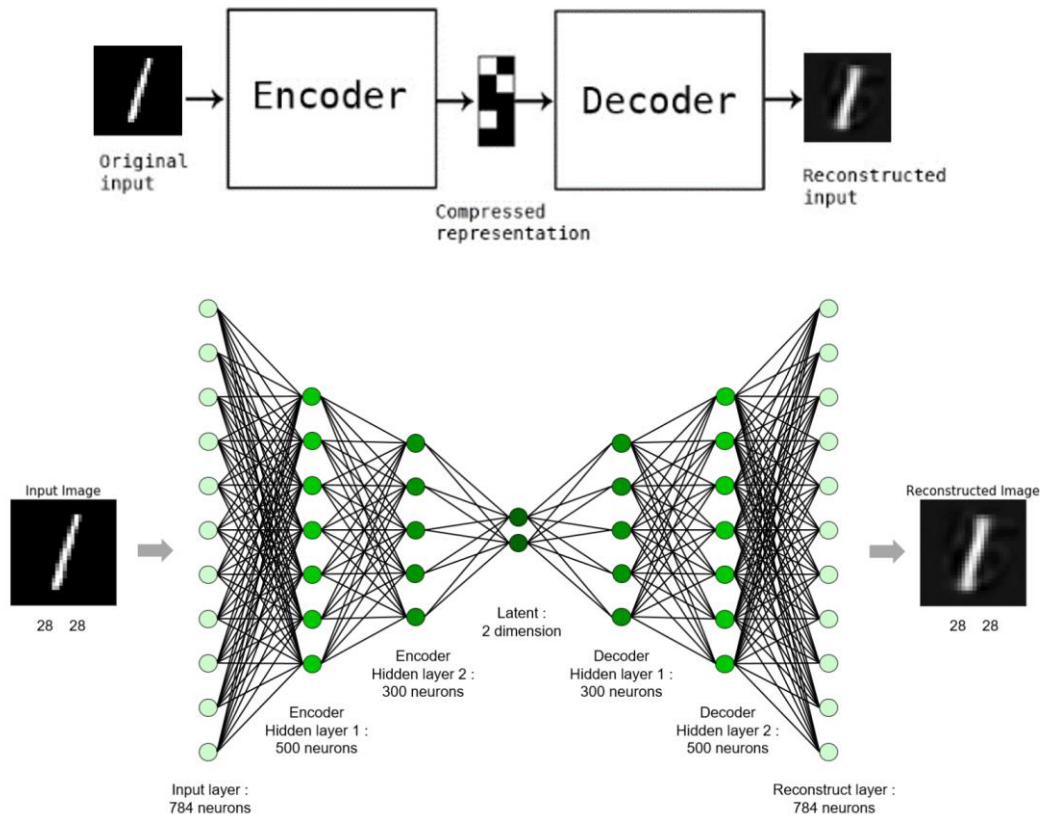


- A proper autoencoder has to capture a "good" parametrization of the signal, and in particular the statistical dependencies between the signal components.

Autoencoder with MNIST

Autoencoder with TensorFlow

- MNIST example
- Use only (1, 5, 6) digits to visualize in 2-D



$$\frac{1}{m} \sum_{i=1}^m (t_i - y_i)^2$$

Import Libraries and Load MNIST Data

```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

from sklearn.neural_network import MLPRegressor
from sklearn.metrics import accuracy_score
```

- Only use (1, 5, 6) digits to visualize latent space in 2-D

```
train_x = np.load('./data_files/mnist_train_images.npy')
train_y = np.load('./data_files/mnist_train_labels.npy')
test_x = np.load('./data_files/mnist_test_images.npy')
test_y = np.load('./data_files/mnist_test_labels.npy')

n_train = train_x.shape[0]
n_test = test_x.shape[0]

print ("The number of training images : {}, shape : {}".format(n_train, train_x.shape))
print ("The number of testing images : {}, shape : {}".format(n_test, test_x.shape))
```

```
The number of training images : 16583, shape : (16583, 784)
The number of testing images : 2985, shape : (2985, 784)
```

Structure of Autoencoder

- Input shape and latent variable shape
- Encoder shape
- Decoder shape

```
# Shape of input and latent variable
```

```
n_input = 28*28
```

```
# Encoder structure
```

```
n_encoder1 = 500
```

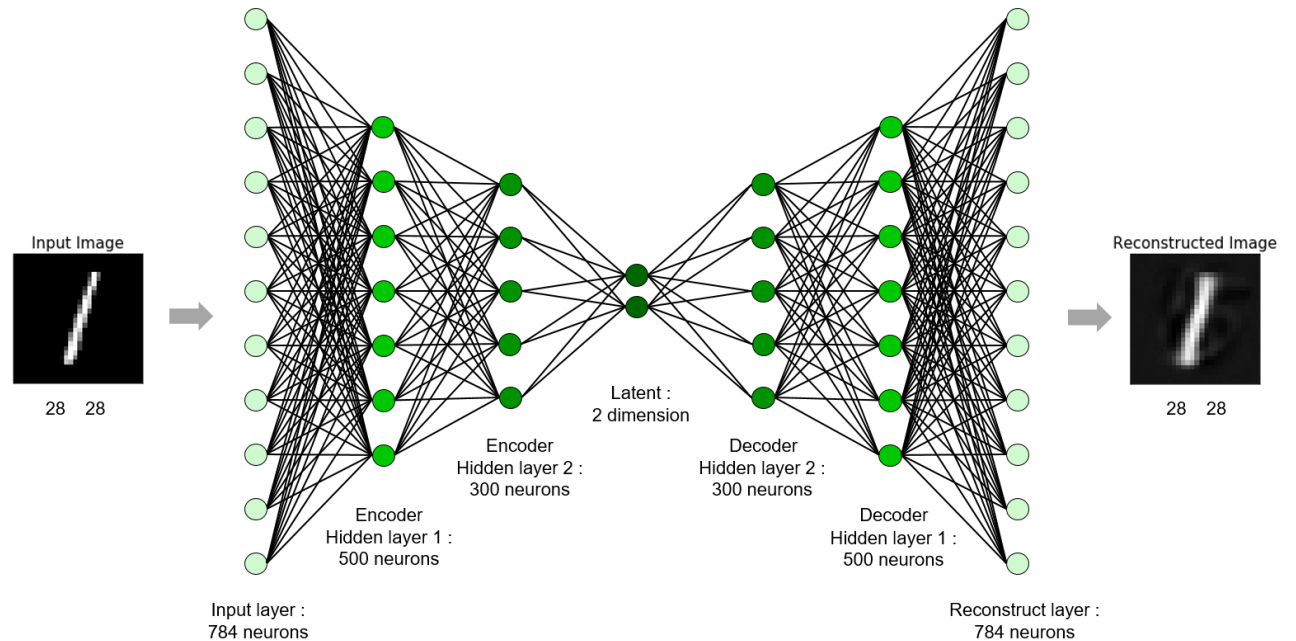
```
n_encoder2 = 300
```

```
n_latent = 2
```

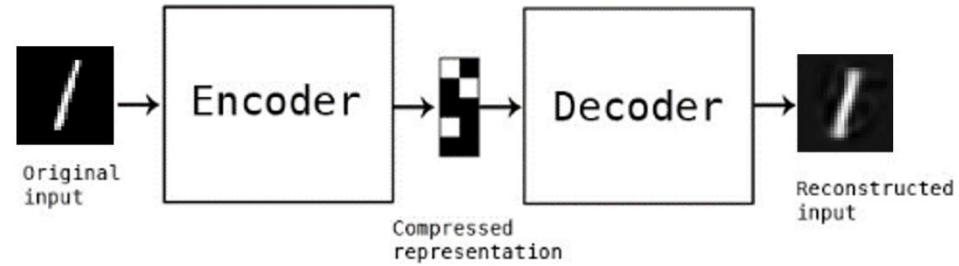
```
# Decoder structure
```

```
n_decoder2 = 300
```

```
n_decoder1 = 500
```



Build a Model

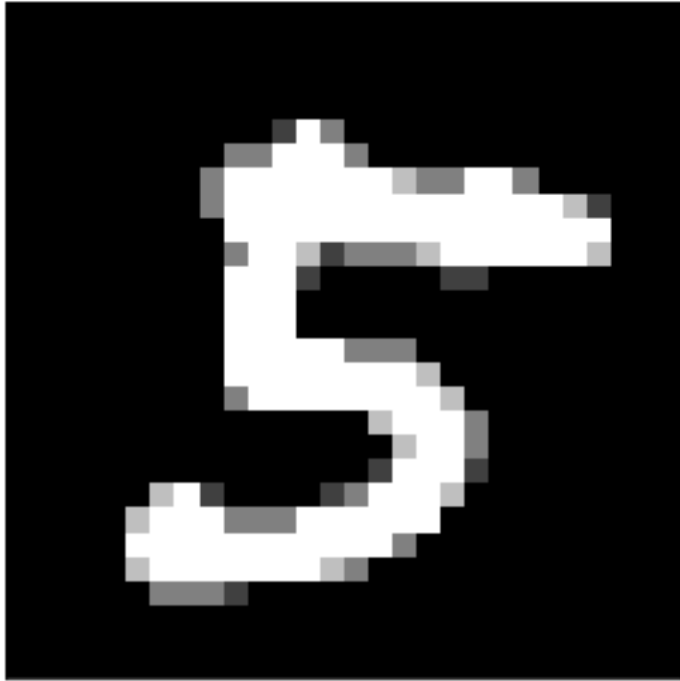


```
reg = MLPRegressor(hidden_layer_sizes = (n_encoder1, n_encoder2, n_latent, n_decoder2, n_decoder1),  
                    activation = 'tanh',  
                    solver = 'adam',  
                    learning_rate_init = 0.0001,  
                    max_iter = 20,  
                    tol = 0.0000001,  
                    verbose = True)
```

Test or Evaluation

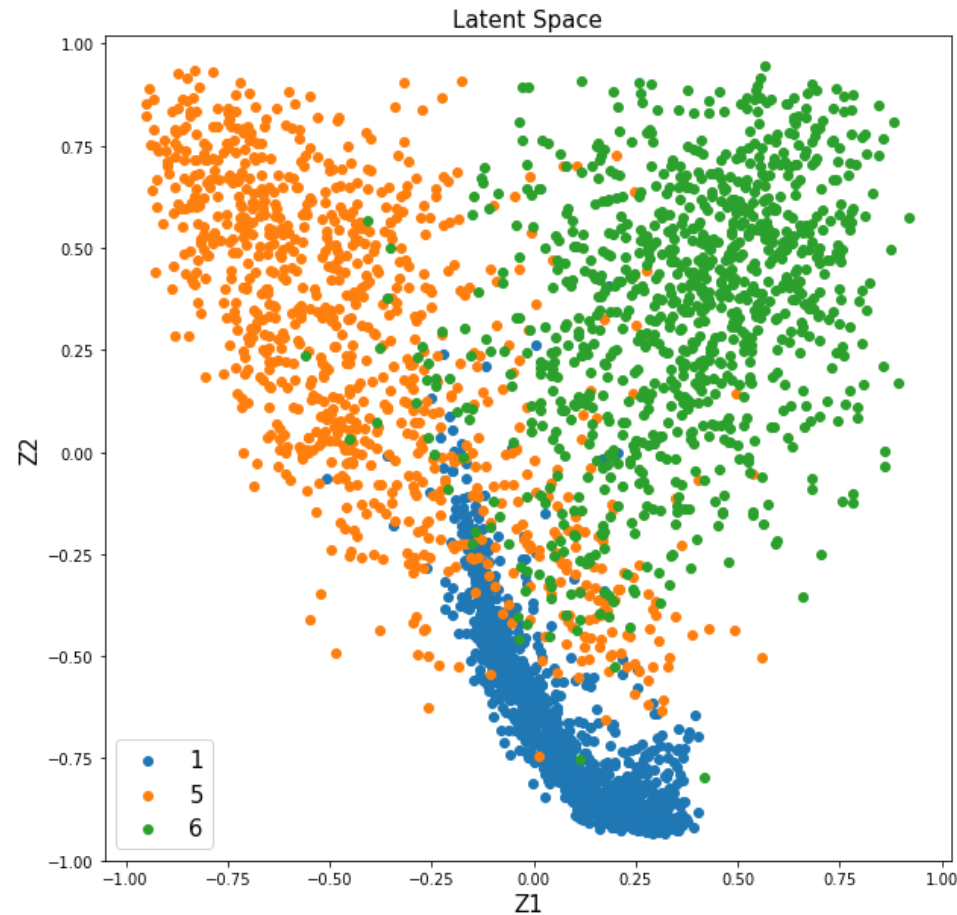
```
idx = np.random.randint(test_x.shape[0])  
x_reconst = reg.predict(test_x[idx].reshape(-1, 784))
```

Input Image



Distribution in Latent Space

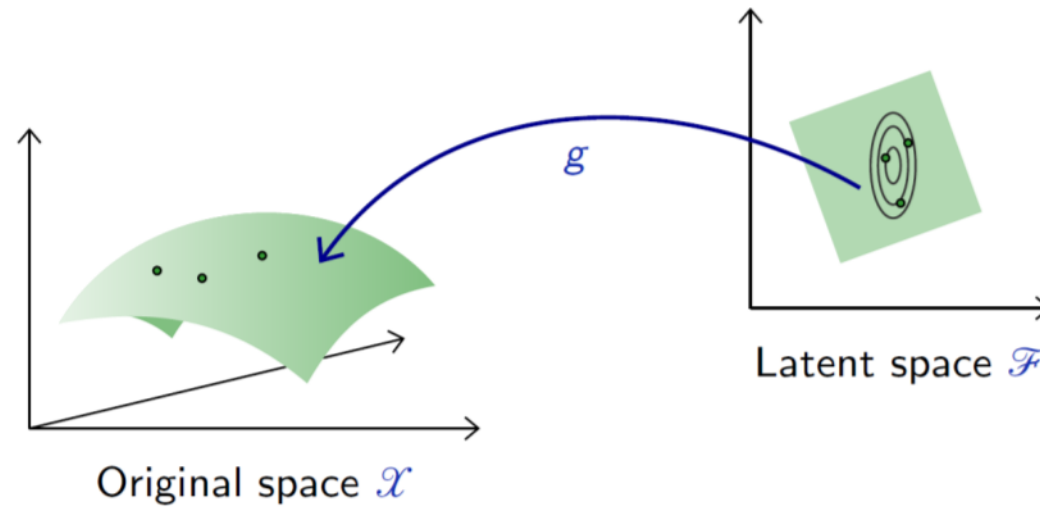
- Make a projection of 784-dim image onto 2-dim latent space



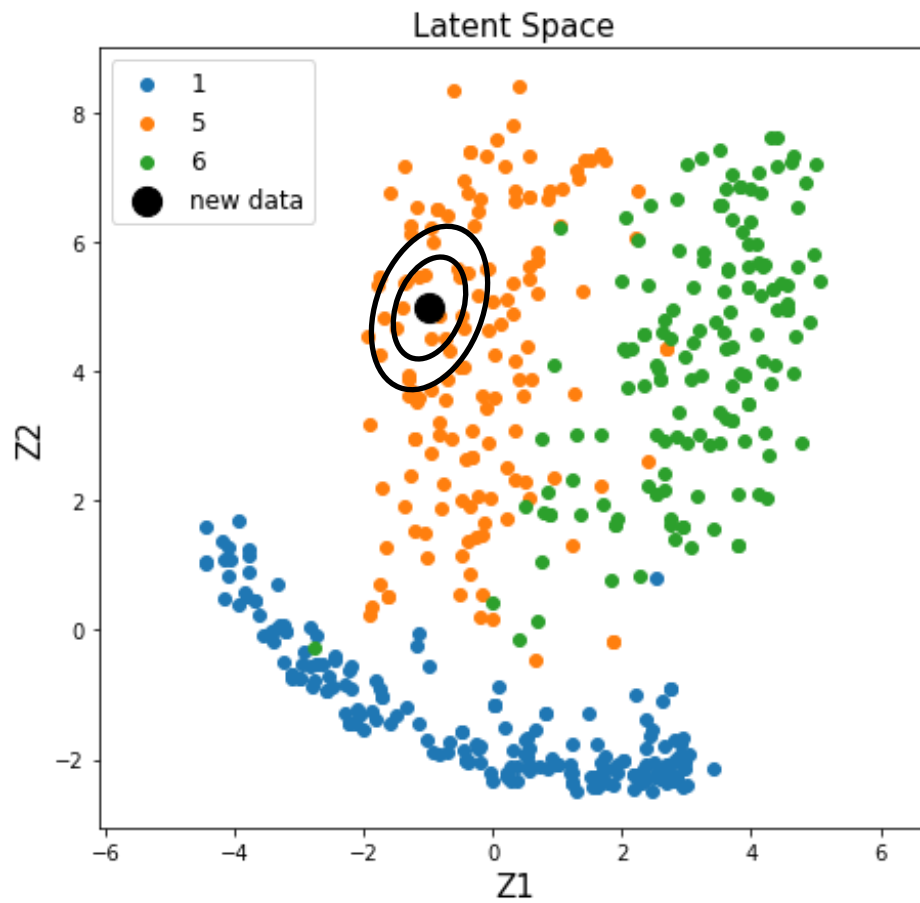
Autoencoder as Generative Model

Generative Capabilities

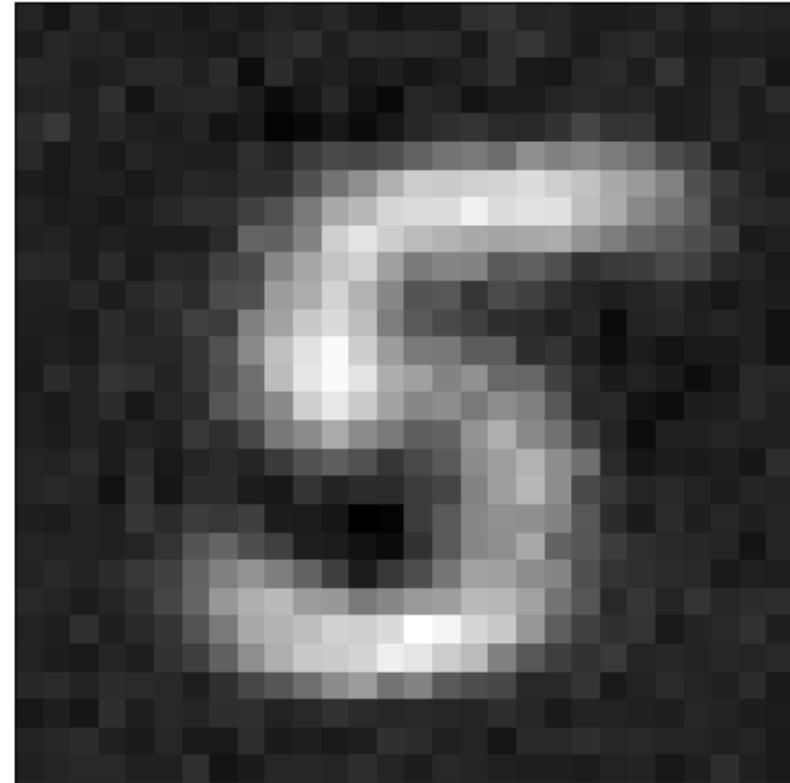
- We can assess the generative capabilities of the decoder g by introducing a [simple] density model q^Z over the latent space \mathcal{F} , sample there, and map the samples into the image space \mathcal{X} with g .



MNIST Example

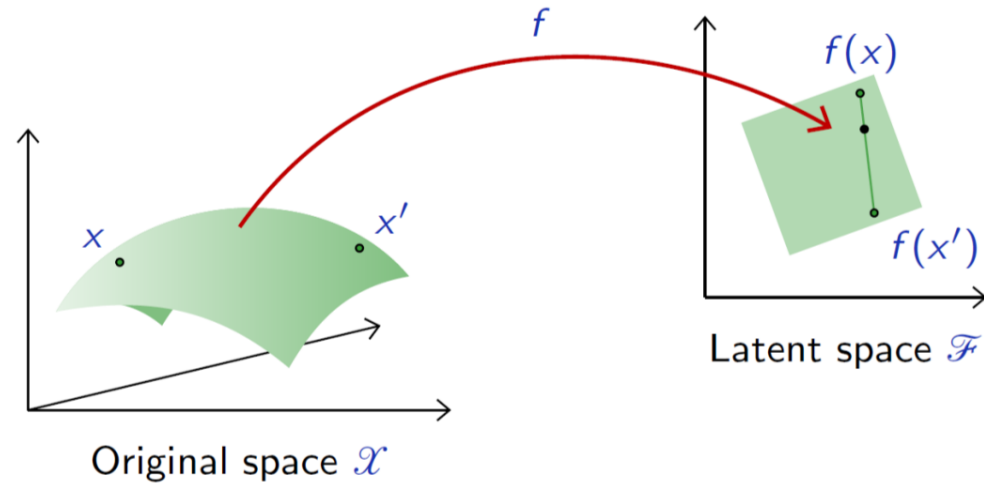


Generated Fake Image



Latent Representation

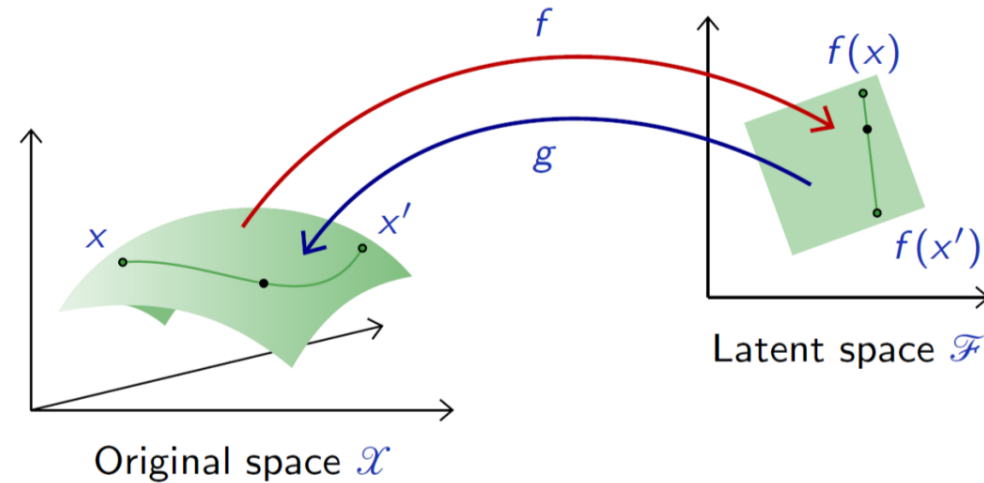
- To get an intuition of the latent representation, we can pick two samples x and x' at random and interpolate samples along the line in the latent space



Latent Representation

- To get an intuition of the latent representation, we can pick two samples x and x' at random and interpolate samples along the line in the latent space

$$g((1 - \alpha)f(x) + \alpha f(x'))$$

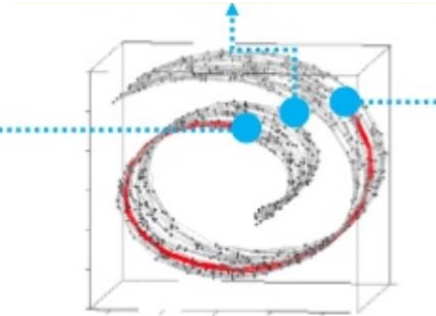


Interpolation in High Dimension

Reasonable distance metric



Interpolation in high dimension



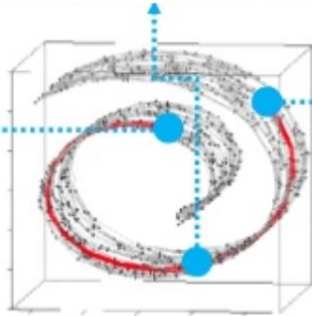
<https://www.cs.cmu.edu/~efros/courses/AP06/presentations/ThompsonDimensionalityReduction.pdf>

Interpolation in Manifold

Reasonable distance metric

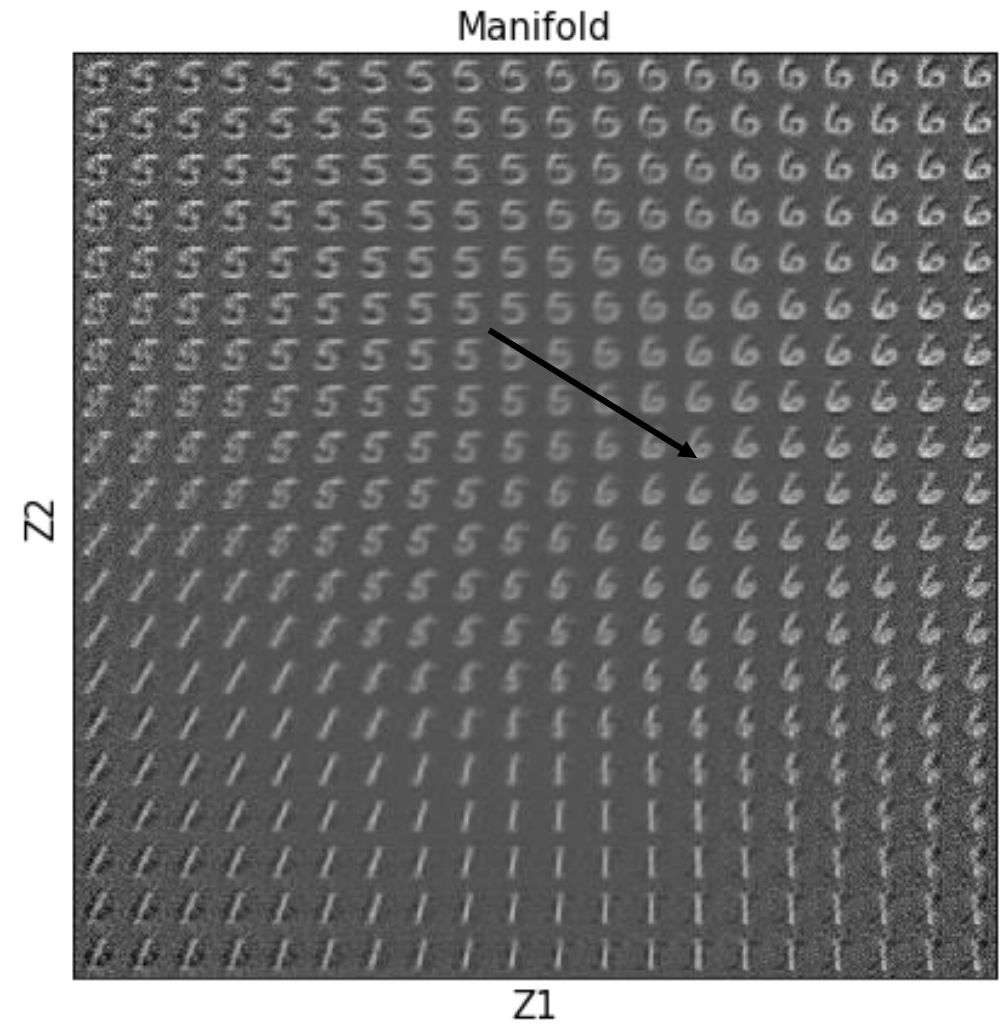
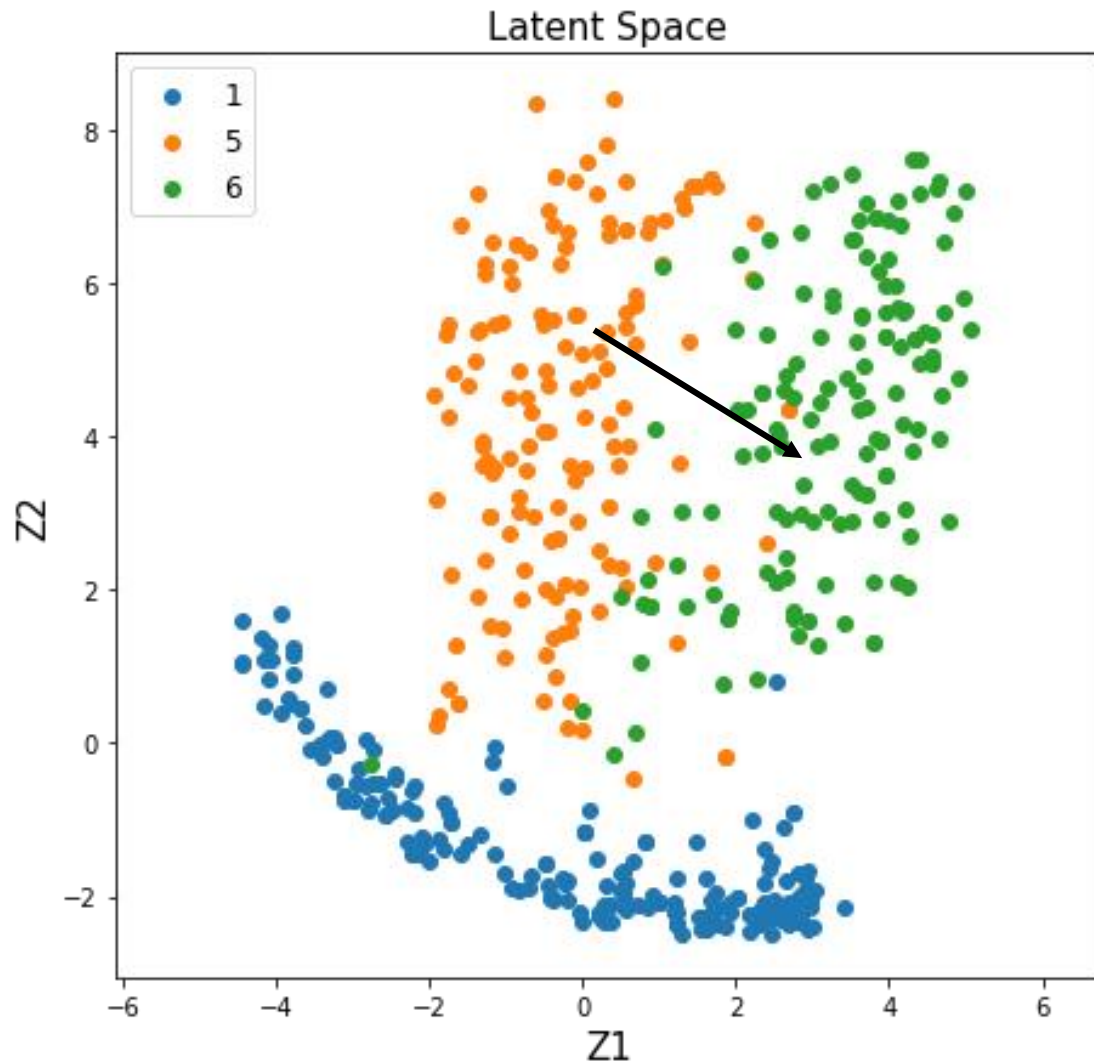


Interpolation in manifold



<https://www.cs.cmu.edu/~efros/courses/AP06/presentations/ThompsonDimensionalityReduction.pdf>

MNIST Example: Walk in the Latent Space



Generative Models

- It generates something that makes sense.
- These results are unsatisfying, because the density model used on the latent space \mathcal{F} is too simple and inadequate.
- Building a “good” model amounts to our original problem of modeling an empirical distribution, although it may now be in a lower dimension space.
- This is a motivation to VAE or GAN.