AutoEncoders

CoE197Z/EE298Z (Deep Learning)

Rowel Atienza, Ph.D.
rowel@eee.upd.edu.ph
github.com/roatienza

Exercise

Draw an image of a 100-peso bill

Draw from what you can remember only

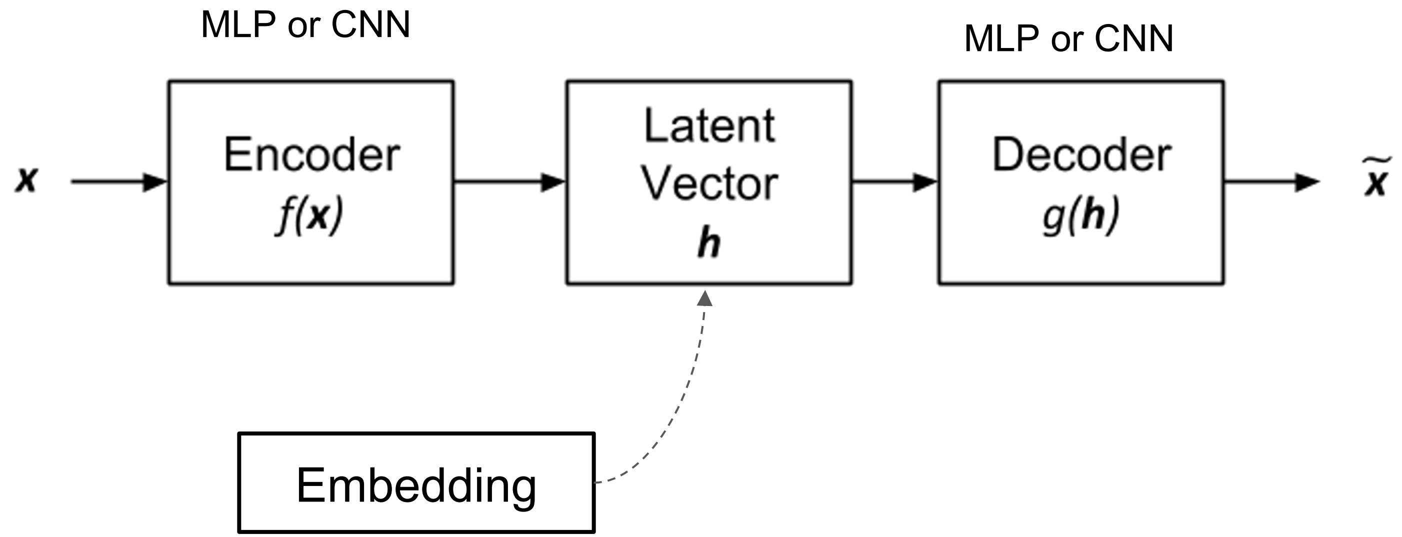
Do not look and copy a 100-peso bill

100
TEXT 1/1X
XX12354F12
100

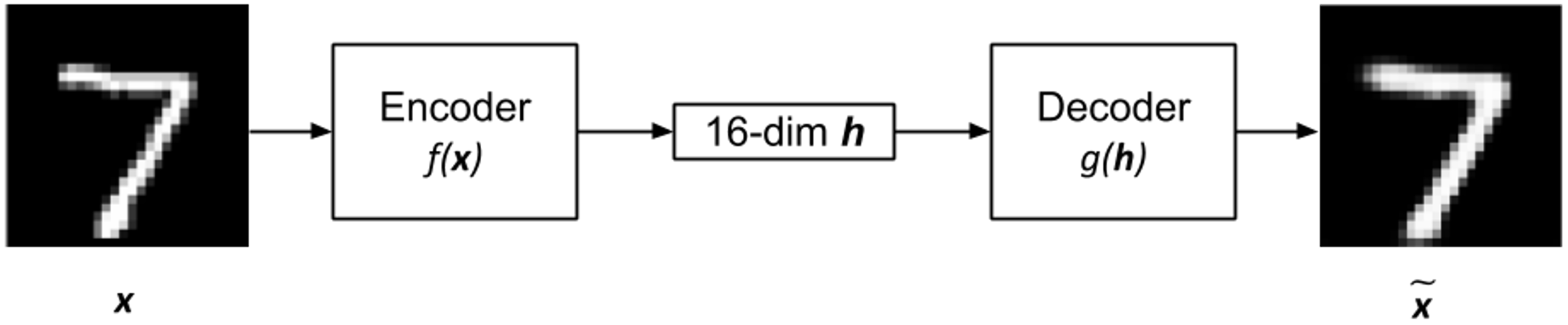
AutoEncoder

- A self-supervised network
  - Supply it with data, it will figure out the mapping from input to output
- A neural network that attempts to copy its input to its output
  - In doing so, it learns a hidden code, $\boldsymbol{h}$, that represents its input
- The network has 2 parts
  - An encoder: $\boldsymbol{h} = f(\boldsymbol{x})$
  - A decoder: $\boldsymbol{r} = g(\boldsymbol{h})$

AutoEncoder



AutoEncoder



AutoEncoder

Autoencoder must not learn: $g(\boldsymbol{h}) = g(f(\boldsymbol{x})) = \boldsymbol{x}$

Instead it must learn to approximate $\boldsymbol{x}$ only; In doing so, it learns special properties of $\boldsymbol{x}$

Roughly, our brain compresses (lossy) information about what it senses

AutoEncoder

For example in MNIST, properties could be writing style, tilt, thickness, roundness of stroke, etc.

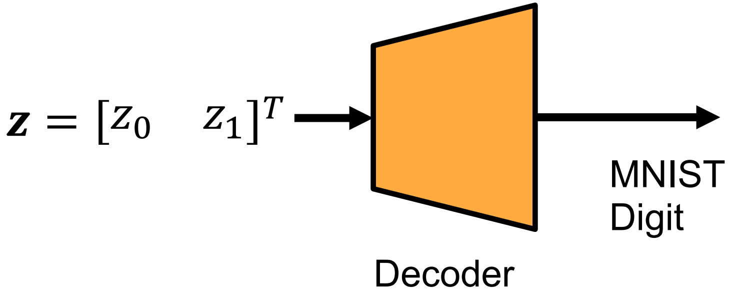
All properties needed to represent digits 0 to 9.

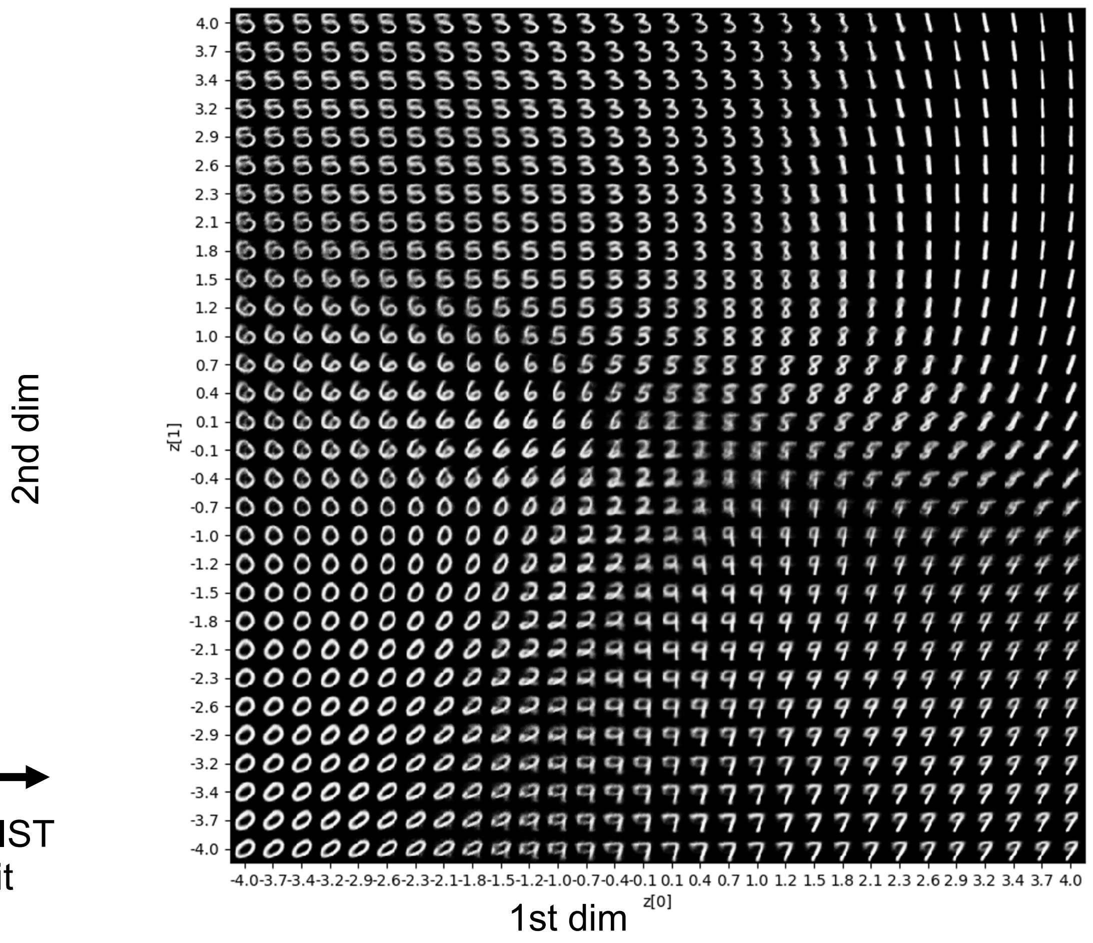
Autoencoder is not an identity function

Let us use 2-dim code on an MNIST AE

Digits over latent code (2-dim)

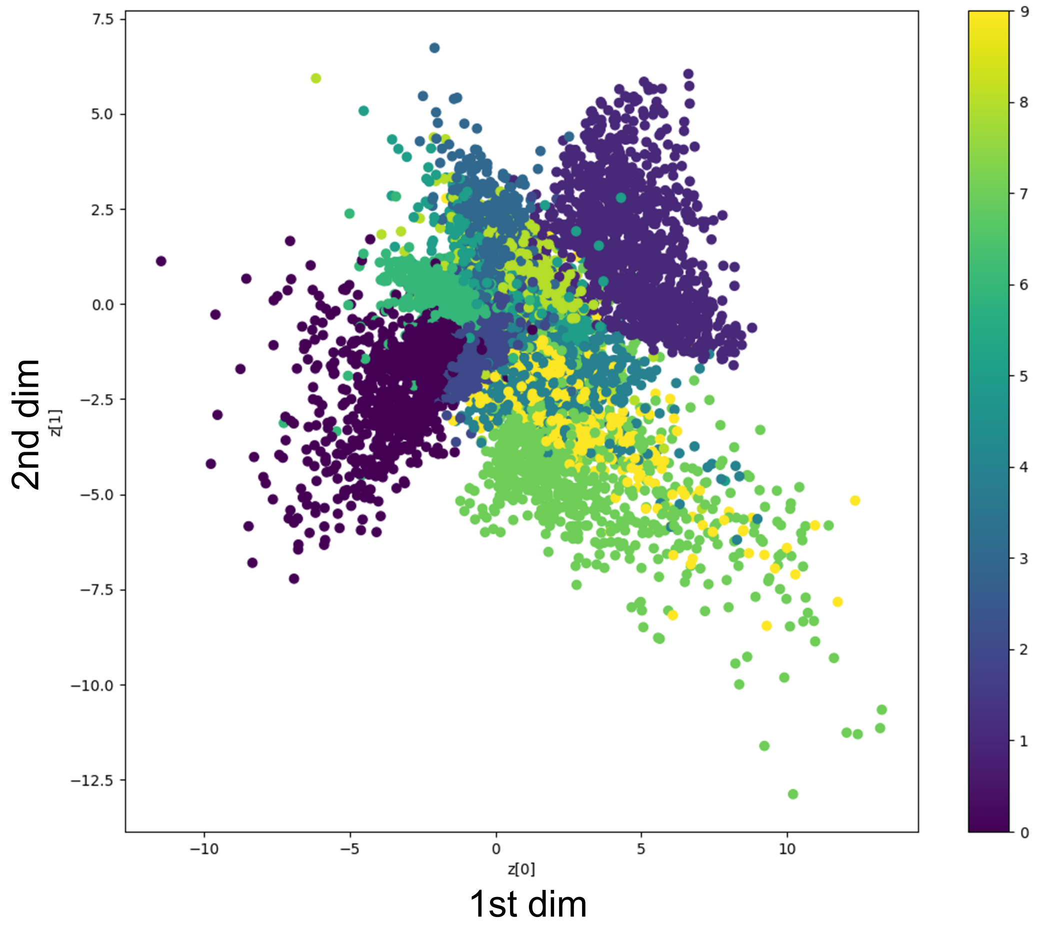
Anywhere we navigate the latent space, there is a corresponding decodable MNIST digit-like image.





Digits over latent code (2-dim)

Similar digits appear to cluster in one region.



AutoEncoder Applications

Dimensionality reduction (compression)

Feature learning

Generative modeling

Unsupervised learning

Many more...

The hidden code, $\boldsymbol{h}$

$\boldsymbol{h}$ is also known as latent representation

Dimensionality is much less than $\boldsymbol{x}$

For example, MNIST $\boldsymbol{x}$ could be 28 x 28 = 784 but $\boldsymbol{h}$ could be 16

The small dimension of $\boldsymbol{h}$ forces the neural network to learn only the most important features of $\boldsymbol{x}$

The hidden code, $\boldsymbol{h}$

The learning process is described by minimizing a loss function

$$L(\boldsymbol{x}, g(f(\boldsymbol{x})))$$

L penalizes g as being dissimilar from $\boldsymbol{x}$

Such as mean squared error (MSE)

The hidden code, $\boldsymbol{h}$ and Regularized AutoEncoder

Principal Component Analysis (PCA) or SVD can only learn linear feature representation

Autoencoder can learn both linear and non-linear feature representations

Stochastic Encoder and Decoder

In AutoEncoder, the input is also the output

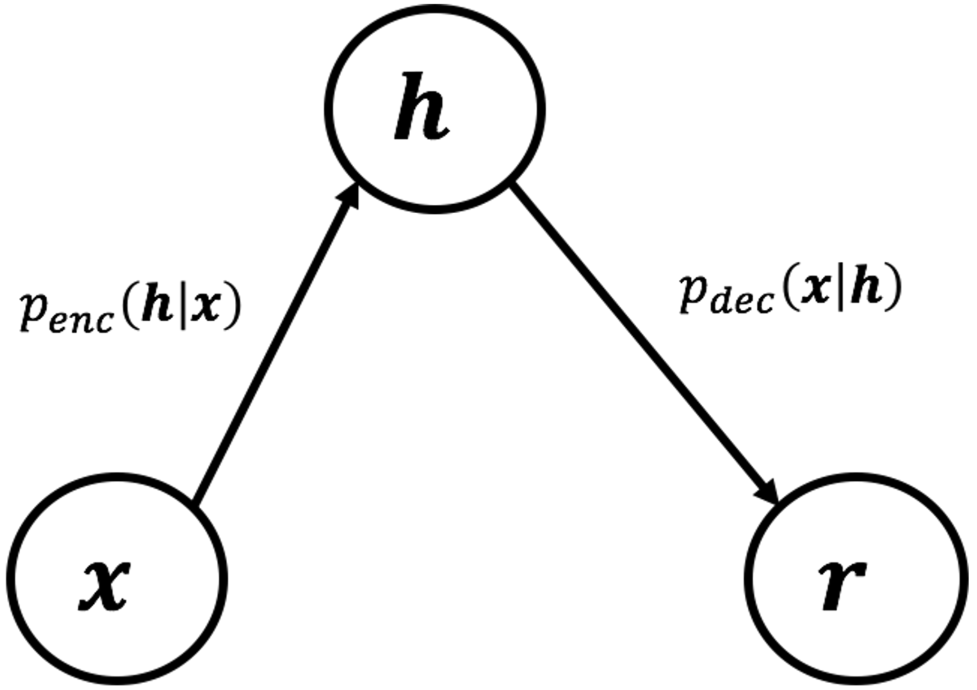
Encoder: $p_{enc}\left(\boldsymbol{h} \middle| \boldsymbol{x}\right)$

Decoder: $p_{dec}\left(\boldsymbol{x} \middle| \boldsymbol{h}\right)$

Similar to other networks, the goal of stochastic autoencoder is to minimize

$$-\log p_{dec}(\boldsymbol{x}|\boldsymbol{h})$$

Encoding/Decoding Distribution Graphical Model

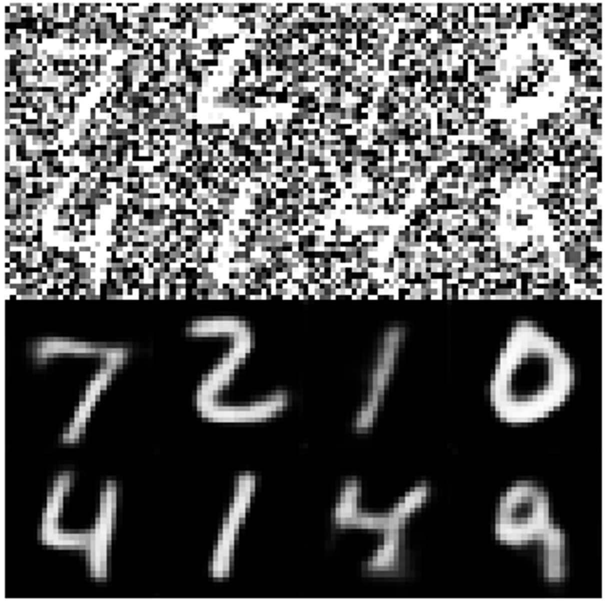

$\boldsymbol{h}$
$p_{enc}(\boldsymbol{h}|\boldsymbol{x})$
$p_{dec}(\boldsymbol{x}|\boldsymbol{h})$
$\boldsymbol{x}$
$\boldsymbol{r}$

Applications

Denoising AutoEncoder

Can we recover clean data given noised data?

Corrupted Input: 1st 2 rows, Denoised Output: last 2 rows



Denoising AutoEncoder

The idea is to recover the true distribution from a corrupted input:

$$L\left(\boldsymbol{x}, g\left(f\left(\widetilde{\boldsymbol{x}}\right)\right)\right)$$

$\widetilde{\boldsymbol{x}}$ is the corrupted version of $\boldsymbol{x}$

The network receives a corrupted input and generates a clean output (clean or uncorrupted version of input)

Noise can also be loss of color (BW to Color), missing part (an old photo with some part of the image torn)

Denoising AutoEncoder Graphical Model

Get latent given noised input

Reconstruct original image given latent

$$\begin{tikzcd}
 & \boldsymbol{h} \arrow[dr, "p_{dec}(\boldsymbol{x}|\boldsymbol{h})"] & \\
\widetilde{\boldsymbol{x}} \arrow[ur, "p_{enc}(\boldsymbol{h}|\widetilde{\boldsymbol{x}})"] & & L \\
 & \boldsymbol{x} \arrow[ul, "C(\widetilde{\boldsymbol{x}}|\boldsymbol{x})"] \arrow[ur] &
\end{tikzcd}$$

Loss function (e.g. MSE)

Add noise

Denoising AutoEncoder

Sample a training example $\boldsymbol{x}$ from training data

Sample a corrupted version of $\boldsymbol{x}$ using corruption process

$$C(\widetilde{\boldsymbol{x}}|\boldsymbol{x})$$

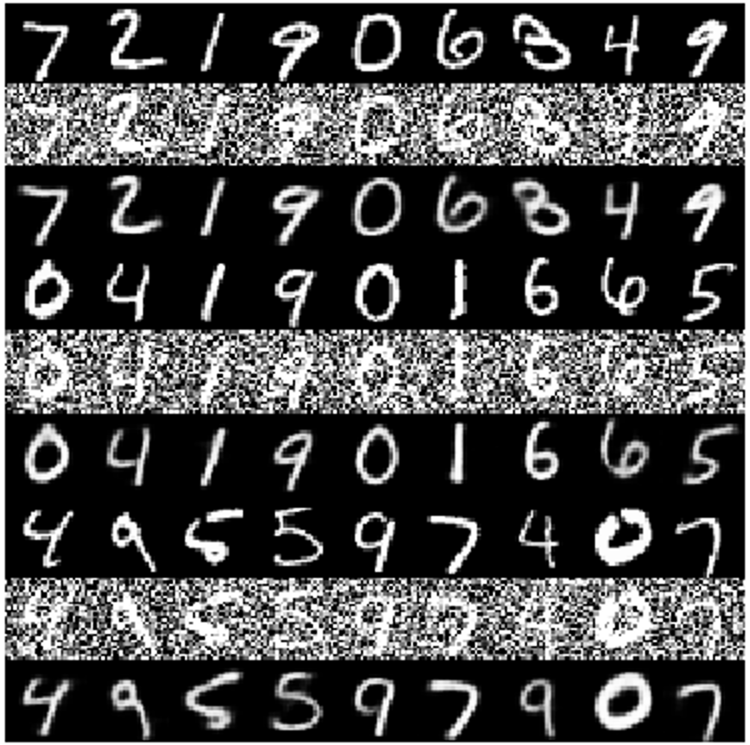
Use data points $(\widetilde{\boldsymbol{x}}, \boldsymbol{x})$to train the network to estimate autoencoder reconstruction distribution

$$p_{recon}(\boldsymbol{x}|\widetilde{\boldsymbol{x}}) = p_{dec}(\boldsymbol{x}|\boldsymbol{h})$$

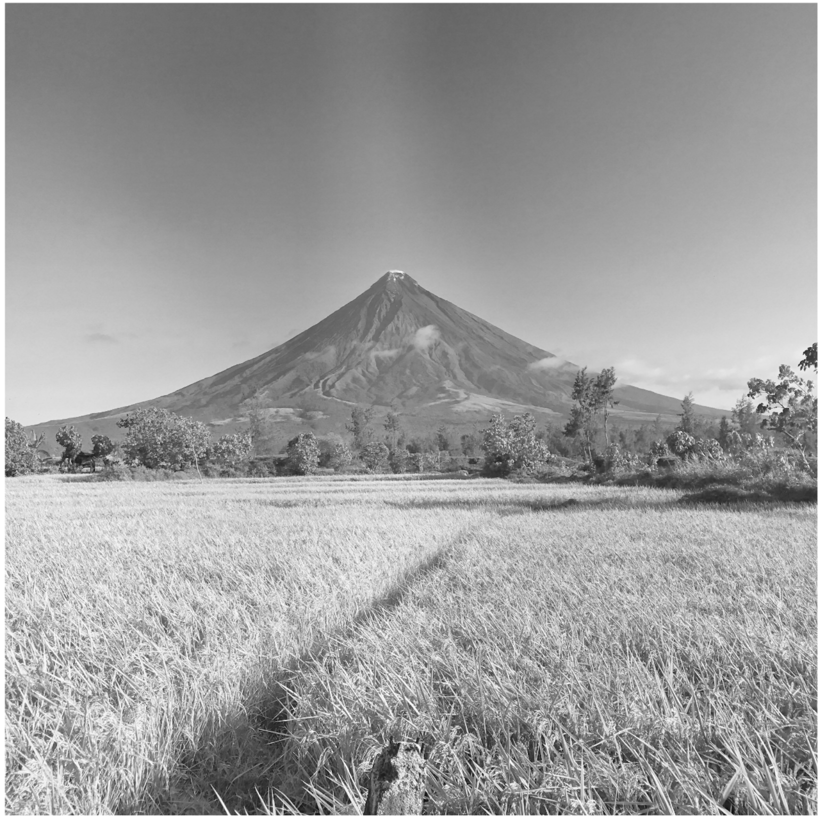
where $\boldsymbol{h}$ is the output of the encoder $\boldsymbol{h} = f(\widetilde{\boldsymbol{x}})$

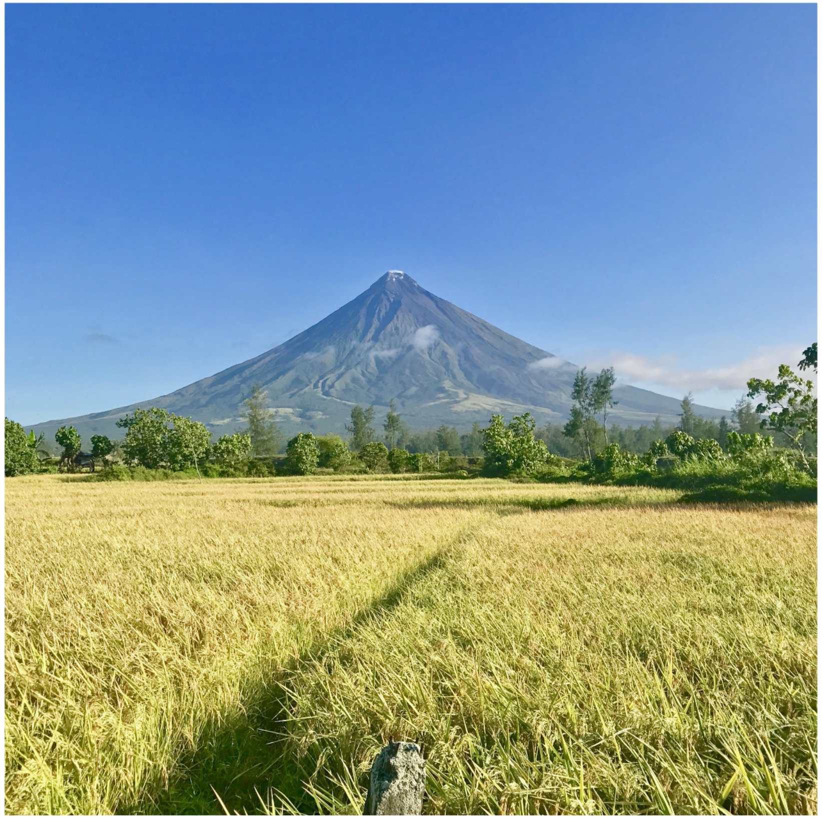
Denoising AutoEncoder on MNIST

Original images: top rows, Corrupted Input: middle rows, Denoised Input: third rows



Colorization





Colorization AutoEncoder

$$\begin{tikzcd}
 & \boldsymbol{h} \arrow[dr, "p_{dec}(\boldsymbol{x}|\boldsymbol{h})"] & \\
\widetilde{\boldsymbol{x}} \arrow[ur, "p_{enc}(\boldsymbol{h}|\widetilde{\boldsymbol{x}})"] & & L \\
 & \boldsymbol{x} \arrow[ul, "C(\widetilde{\boldsymbol{x}}|\boldsymbol{x})"] \arrow[ur] &
\end{tikzcd}$$

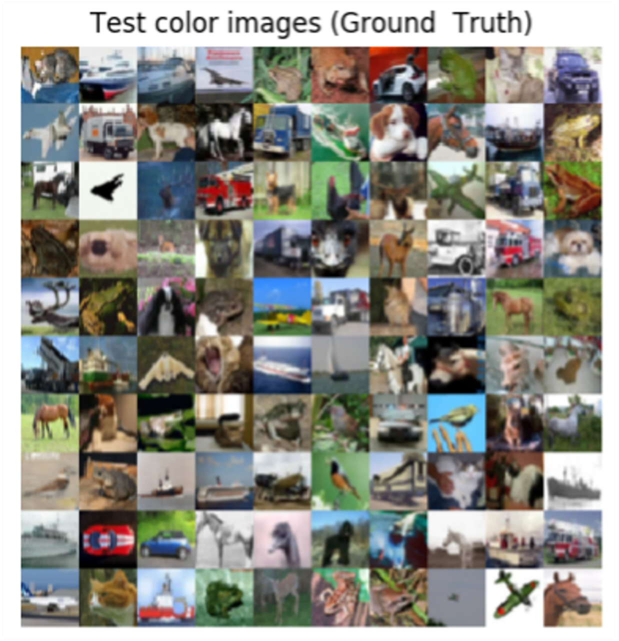
Get latent given grayscale input

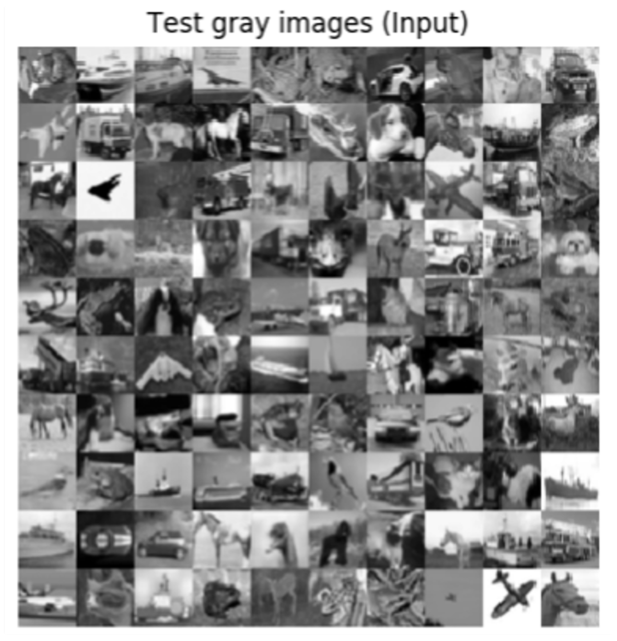
Reconstruct color image given latent

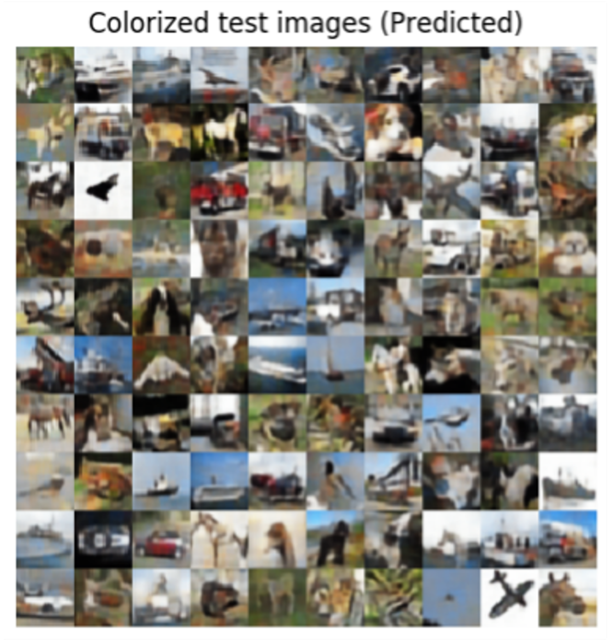
Loss function (e.g. MSE)

Remove color

Colorization AutoEncoder on CIFAR10







Assignment: Due End of Sat Nov 14, 2020

Can we use an autoencoder to build a self-supervised classifier?

After building the AE, use the trained E to generate feature vectors.

Use K-means to cluster the feature vectors

Use Hungarian Algo to assign label to each cluster

Best performing solution has bonus points