

Advanced Computer Vision

Tanveer Hussain

htanveer3797@gmail.com



جامعة الملك عبد الله
للعلوم والتقنية
King Abdullah University of
Science and Technology

KAUST Academy
King Abdullah University of Science and Technology

Course designed by **Naeem Ullah Khan** (naeemullah.khan@kaust.edu.sa), updated by Tanveer Hussain

Supervised Learning

Data: (x, y)

x is data, y is label

Goal: Learn a *function* to map $x \rightarrow y$

Examples: Classification, regression, object detection, semantic segmentation, image captioning, etc.



→ Cat

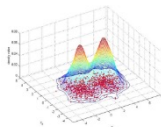
Unsupervised Learning

Data: x

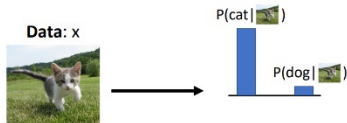
Just data, **no labels!**

Goal: Learn some underlying hidden *structure* of the data

Examples: Clustering, dimensionality reduction, feature learning, density estimation, etc.



Discriminative Model:
Learn a probability
distribution $p(y|x)$



Density Function

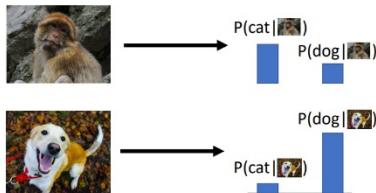
$p(x)$ assigns a positive number to each possible x ; higher numbers mean x is more likely

Density functions are **normalized**:

$$\int_X p(x) dx = 1$$

Different values of x **compete** for density

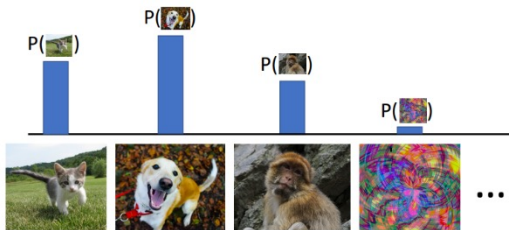
Generative Model:
Learn a probability
distribution $p(x)$



Discriminative model: No way for the model to handle unreasonable inputs; it must give label distributions for all images

Discriminative Model:
Learn a probability
distribution $p(y|x)$

Generative Model:
Learn a probability
distribution $p(x)$



Generative model: All possible images compete with each other for probability mass

Requires deep image understanding! Is a dog more likely to sit or stand? How about 3-legged dog vs 3-armed monkey?

Discriminative Models

Data: (x, y)
 x is data, y is label

Goal: Learn a function to map $x \rightarrow y$

Examples: Logistic Regression, SVM, Neural Networks

Advantages: Higher accuracy, less data needed

Limitations: Cannot generate new data

Generative Models

Data: x
Just data, no labels!

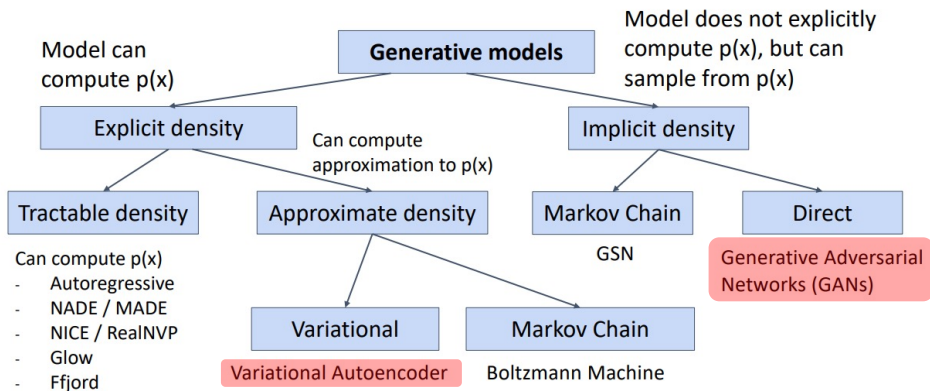
Goal: Learn the underlying data distribution to generate new data

Examples: GMM, HMM, GANs, VAEs

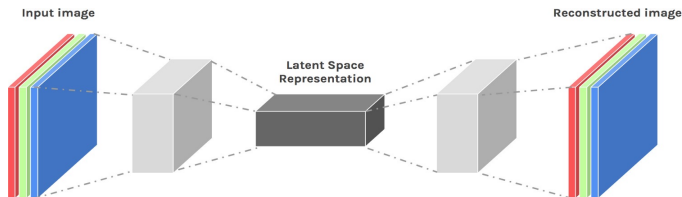
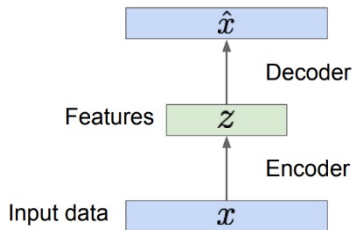
Advantages: Generates new data, useful for unsupervised learning

Limitations: Requires more data, complex to train

- Machine learning models that learn to generate new data samples similar to the training data.

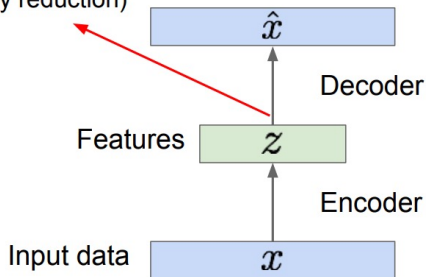


- Autoencoder is an unsupervised strategy to learn lower-dimensional features representation from unlabelled training data.



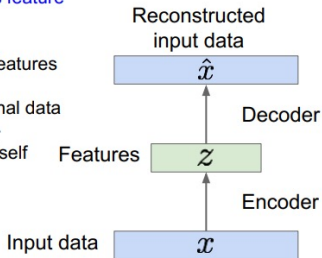
Autoencoder architecture

z usually smaller than x
(dimensionality reduction)



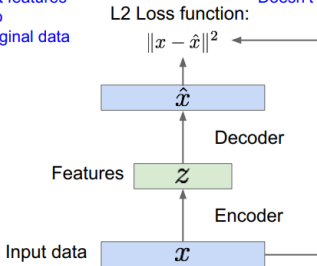
How to learn this feature representation?

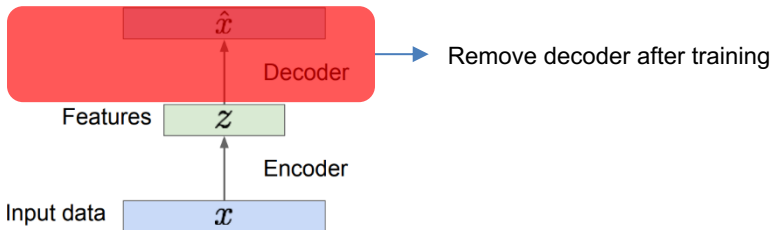
Train such that features can be used to reconstruct original data "Autoencoding" - encoding input itself



Train such that features can be used to reconstruct original data

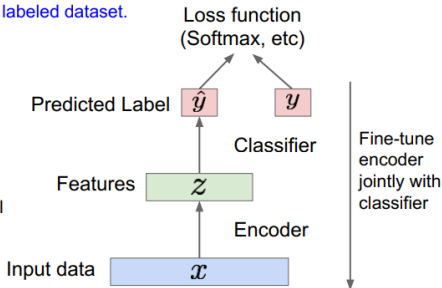
Doesn't use labels!

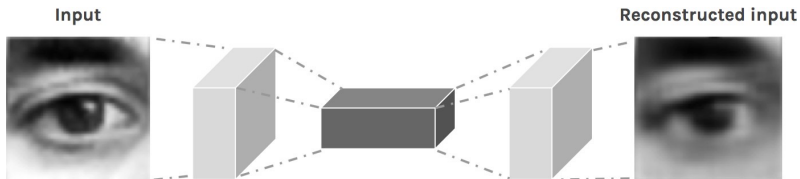




Transfer from large, unlabeled dataset to small, labeled dataset.

Encoder can be used to initialize a **supervised** model

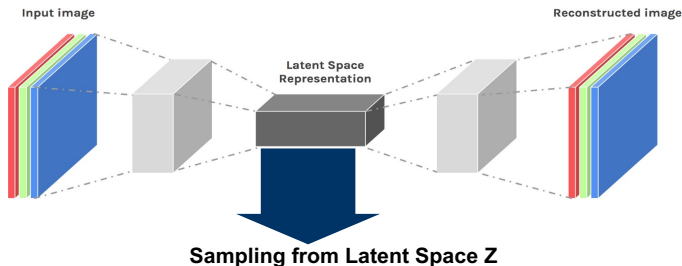




Sample Autoencoder

<https://douglasduhaime.com/posts/visualizing-latent-spaces.html>

- ▶ Autoencoders project data into a latent space Z .
- ▶ What if we sample a new embedding vector from Z and then have the decoder reconstruct the image from it?
- ▶ **Does not work.** Autoencoders just learn a function that maps input to output. The learned latent space is too discontinuous to work as a generative model.



Concept: If we could sample a new embedding vector from the latent space Z , and then use the decoder to reconstruct the image, we would be able to generate new images.

Challenge: In standard autoencoders, the latent space is often irregular and discontinuous.

This means that sampled vectors may not correspond to meaningful images, making it unsuitable for generative purposes.

Autoencoders as generative models (cont.)

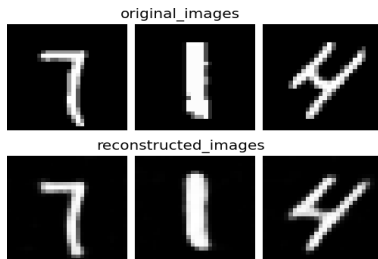










Image reconstruction with autoencoder trained on MNIST digits

Input image	2-D latent space	5-D latent space	10-D latent space
			
			

The table displays four columns of handwritten digits. The first column, 'Input image', shows a 10x10 grid of 100 original MNIST digits. The subsequent three columns show the reconstructed digits using different latent space dimensions: '2-D latent space', '5-D latent space', and '10-D latent space'. Each column also displays a 10x10 grid of 100 reconstructed digits. The 10-D latent space reconstructions appear slightly more similar to the input images than the 2-D and 5-D reconstructions.

Impact of size of latent representation on reconstruction

generated_images

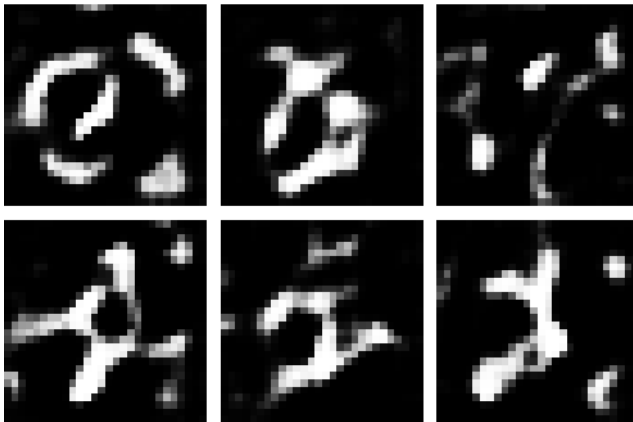


Image generation with autoencoder trained on MNIST digits. Encoding vector sampled from latent space Z and the passed to decoder.

Why Low Performance?

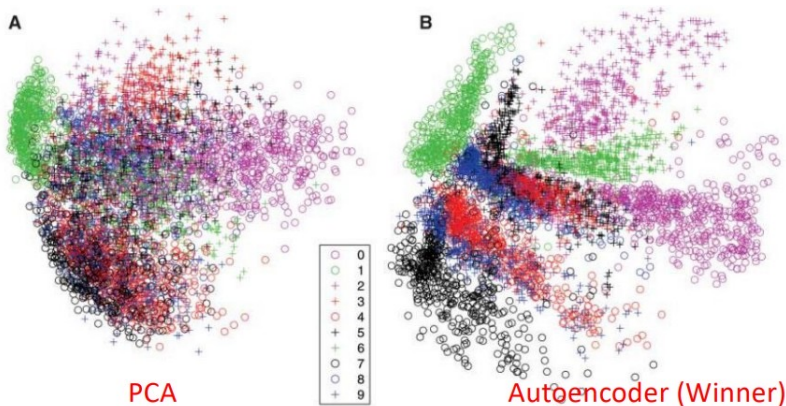
Irregular Latent Space

- ❑ The autoencoder's latent space is learned in such a way that it minimizes reconstruction error for the training data.
- ❑ This does not ensure a smooth or continuous latent space, making random samples likely to be invalid or nonsensical.

Generative Model Requirements

- ❑ For a generative model, the latent space needs to follow a known, continuous distribution (e.g., Gaussian) to ensure that any sample from this space can be decoded into a meaningful output.

- ▶ While autoencoders themselves have very low generative power, we will soon talk about a type of autoencoders called **Variational Autoencoders** which are specifically designed for generative modeling.
- ▶ Other use cases of Autoencoders include:
 - Data encoding and dimensionality reduction
 - Image denoising and super-resolution
 - Image completion
 - Image colorization



t-SNE visualization on MNIST digits dataset. PCA vs. Autoencoders. The image vector is projected into \mathbb{R}^2 .



Image super-resolution using Autoencoders

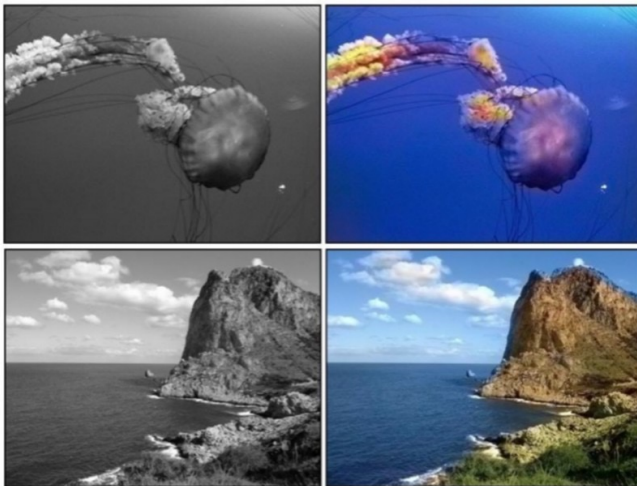


Image colorization using Autoencoders

- ▶ Autoencoders don't work as generative models because the latent space Z is too discontinuous.
- ▶ **Solution:** Let's rectify that.
- ▶ **Variational Autoencoders:** Probabilistic spin on autoencoders that will let us sample from the model to generate data.

Traditional AE

- ❑ Encoder: Maps the input data x to a latent space z .
- ❑ Decoder: Maps the latent space z back to the reconstructed input x .
- ❑ Loss Function: Typically, the reconstruction error (e.g., Mean Squared Error).

Variational AE

- ❑ Encoder: Instead of mapping the input x directly to a latent variable z , the encoder maps x to a distribution over the latent space. Specifically, it outputs the parameters of a Gaussian distribution: a mean vector μ and a standard deviation vector σ .
- ❑ Latent space sampling: A latent variable z is sampled from this Gaussian distribution using the reparameterization trick to ensure backpropagation can be used.
- ❑ Decoder: Maps the sampled latent variable z back to the reconstructed input x^\wedge .
- ❑ Loss Function: Combines the reconstruction error with a regularization term (Kullback-Leibler divergence) to ensure the latent space follows a Gaussian distribution.

Latent Space Representation:

- ❑ In VAEs, the encoder outputs the parameters of a Gaussian distribution (μ and σ) rather than a single point.
- ❑ This makes the latent space continuous and ensures that nearby points in the latent space correspond to similar outputs, which is crucial for generating new data.

Latent Space Representation:

- ❑ In VAEs, the encoder outputs the parameters of a Gaussian distribution (μ and σ) rather than a single point.
- ❑ This makes the latent space continuous and ensures that nearby points in the latent space correspond to similar outputs, which is crucial for generating new data.

Reparameterization Trick:

- ❑ To allow backpropagation through the sampling process, VAEs use the reparameterization trick: $z = \mu + \sigma \odot \epsilon$, where ϵ is sampled from a standard normal distribution.
- ❑ This trick makes the sampling step differentiable, enabling gradient descent optimization.

Latent Space Representation:

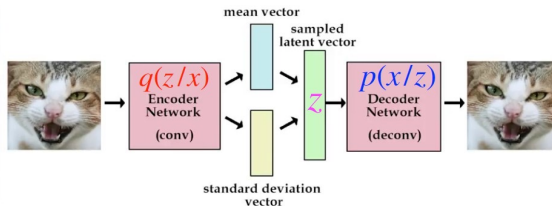
- ❑ In VAEs, the encoder outputs the parameters of a Gaussian distribution (μ and σ) rather than a single point.
- ❑ This makes the latent space continuous and ensures that nearby points in the latent space correspond to similar outputs, which is crucial for generating new data.

Reparameterization Trick:

- ❑ To allow backpropagation through the sampling process, VAEs use the reparameterization trick: $z = \mu + \sigma \odot \epsilon$, where ϵ is sampled from a standard normal distribution.
- ❑ This trick makes the sampling step differentiable, enabling gradient descent optimization.

Loss Function:

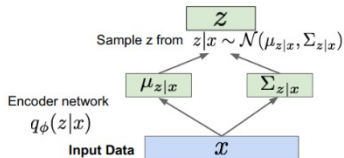
- ❑ **Reconstruction Loss:** Measures how well the decoder reconstructs the input data from the latent variable.
- ❑ **KL Divergence Loss:** Measures how close the learned latent variable distribution is to the prior distribution (typically a standard Gaussian).
- ❑ The total loss is the sum of the reconstruction loss and the KL divergence loss.



Variational Autoencoder Architecture

Sample $\epsilon \sim \mathcal{N}(0, I)$

$$z = \mu_{z|x} + \epsilon \sigma_{z|x}$$



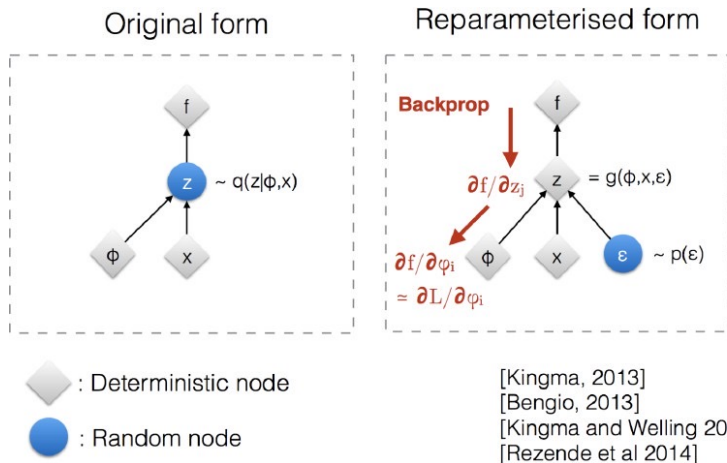
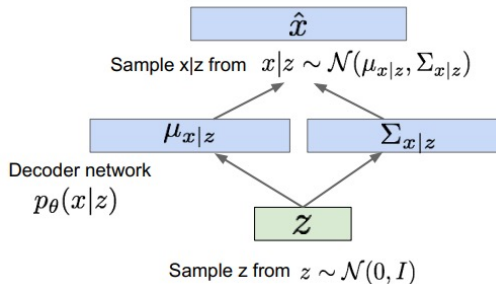
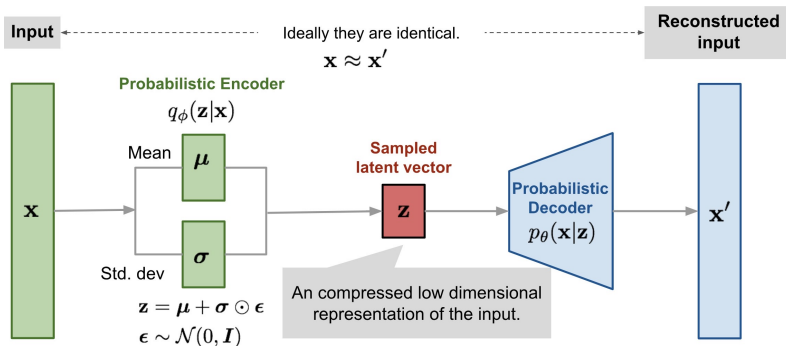


Figure 10: Reparameterization trick to make back propagation possible



For full derivation of KL Loss, read [here](#)

Reparameterization Trick (cont.)



Variational Autoencoder with reparameterization trick

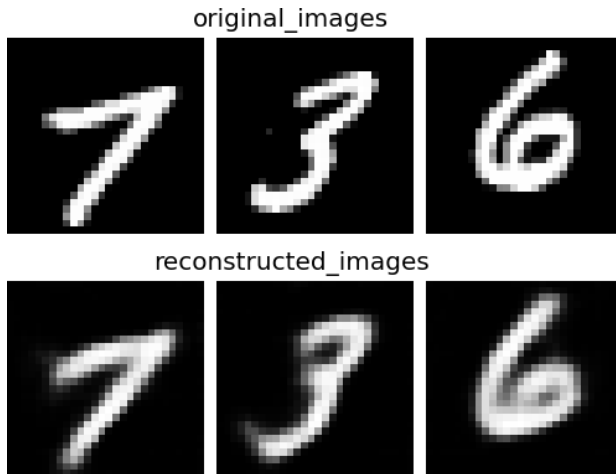


Figure 12: Image reconstruction with variational autoencoders on MNIST digits dataset

generated_images

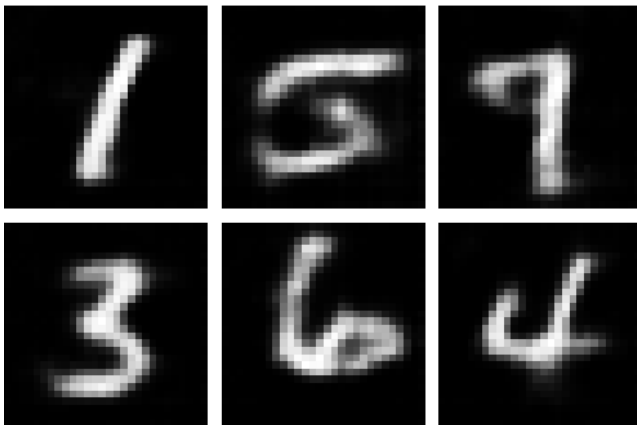


Figure 13: Image generation with variational autoencoders on MNIST digits dataset. Sample an encoding vector from $N(0, 1)$ and passed it through decoder

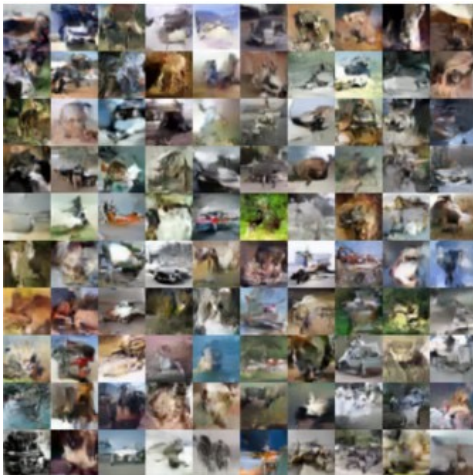


Figure 14: Image generation with variational autoencoders on CIFAR-10 32x32 dataset

- ▶ **Basic Idea:** Different neurons in latent space should be uncorrelated i.e. they all try to learn something different about input data.
- ▶ **Implementation:**

$$L(\theta, \phi; x, z, \beta) = E_{q_{\phi}(z|x)}(\log p_{\theta}(x|z)) - \beta KL(q_{\phi}(z|x)||p(z))$$

- ▶ Increasing the β is forcing variational autoencoder to encode the information in only few latent variables



Figure 15: Azimuthal rotation in β -VAEs and simple VAEs. β -VAEs produce more disentangled rotation whereas some other features also change in simple VAEs.

Reference Slides

- ▶ Fei-Fei Li "Generative Deep Learning" CS231
- ▶ Hao Dong "Deep Generative Models"
- ▶ Hung-Yi Lee "Machine Learning"
- ▶ Francois Fleuret "Deep Learning" EE559
- ▶ Murtaza Taj "Deep Learning" CS437
- ▶ Kreis, Gao & Vahdat [CVPR 2022 Tutorial](#)
- ▶ Rogge & Rasul [The Annotated Diffusion Model](#)