Master 2 Computer Science

Advanced Neural Networks Architectures

Generative Models

Or how magic happens in AI

Part 1. AutoEncoders

Akka Zemmari

Table of Contents

References

This presentation is based on the following resources:

- Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y., *Generative Adversarial Networks*, NIPS, 2014.
- Goodfellow, I., Bengio, Y., Courville, A., *Deep Learning*, MIT Press, 2016.
- Kingma, D. P., Welling, M., *Auto-Encoding Variational Bayes*, ICLR, 2014.
- Radford, A., Metz, L., Chintala, S., *Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks*, ICLR, 2016.

Introduction to Generative Models

Introduction to Generative Models

Intuition

- Generative models learn to model the distribution of data.
- They can generate new samples that resemble the training data.
- Applications: Image generation, data augmentation, creative AI.

Generative vs Discriminative Models

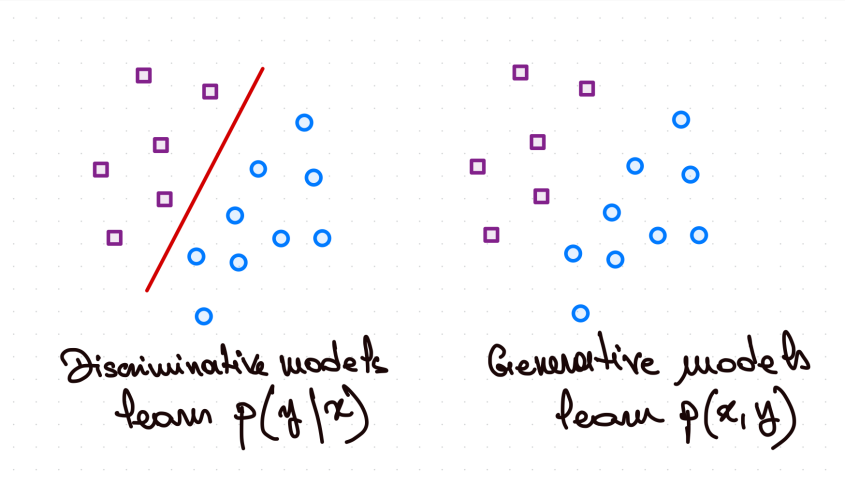

Figure 1: Generative vs Discriminative Models

Key difference:
Given data x and labels y, a generative model captures $p(x, y)$, while a discriminative model focuses on $p(y|x)$.

AutoEncoders (AE)

AutoEncoders (AE)

Goal: Learn to compress and reconstruct input data.

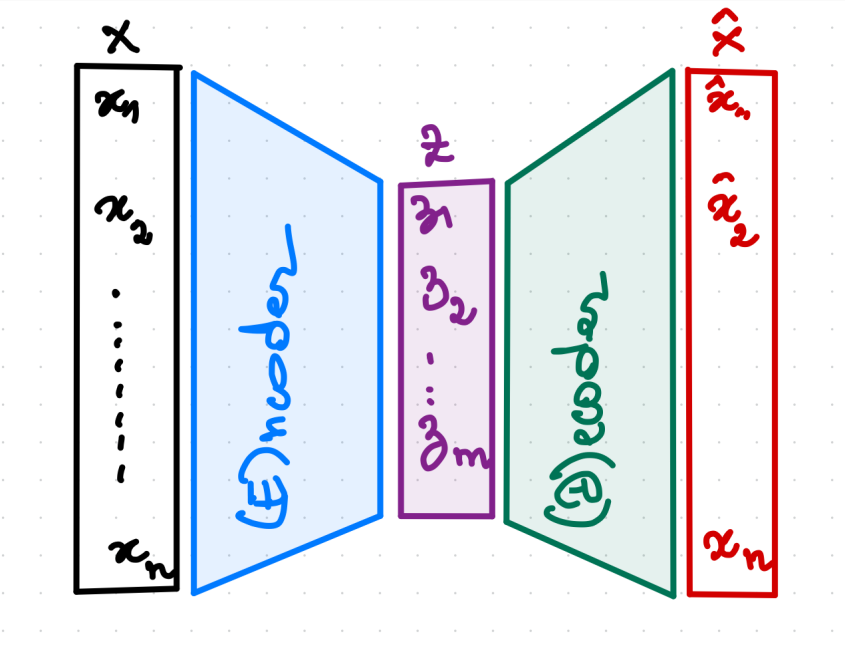

Figure 2: AutoEncoder Architecture

AutoEncoders (AE)

Mathematical Formulation

- An AutoEncoder consists of :
  - $E : \mathbb{R}^n \to \mathbb{R}^m$ (Encoder)
  - $D : \mathbb{R}^m \to \mathbb{R}^n$ (Decoder).
- The goal is to minimize the reconstruction error between x and its reconstruction $\hat{x} = D(E(x))$.
- Typically, the reconstruction loss is measured by the mean squared error (MSE) or cross-entropy loss.

AutoEncoders (AE)

Mathematical Formulation

- Encoder: $E(x) = f(W_e x + b_e)$, where f is a non-linear activation function.
- Decoder: $D(z) = g(W_d z + b_d)$, where g is another non-linear activation function.
- Reconstruction: $\hat{x} = D(E(x)) = g(W_d f(W_e x + b_e) + b_d)$.
- Objective Function:

$$L(x, \hat{x}) = \frac{1}{N} \sum_{i=1}^{N} ||x^{(i)} - \hat{x}^{(i)}||^2 = \frac{1}{N} \sum_{i=1}^{N} ||x^{(i)} - D(E(x^{(i)}))||^2.$$

AutoEncoders (AE)

Linear AutoEncoders

- When f and g are linear functions, an AutoEncoder is equivalent to Principal Component Analysis (PCA).
- The encoder and decoder matrices can be viewed as the principal components of the data.
- **Theorem (PCA and Linear AutoEncoders)** Let X be a centered data matrix. A linear AutoEncoder minimizes the reconstruction loss if, and only if, the weight matrix W_e of the encoder is the matrix of principal components of X.

AutoEncoders (AE)

The Bottleneck Theory

- The "bottleneck" refers to the dimensionality reduction in the latent space $z = E(x)$.
- The dimensionality of z controls the trade-off between information loss and generalization.

Theorem (Information Bottleneck)
Let $Z = E(X)$ be the latent variable.

For any stochastic AutoEncoder, the optimal encoding maximizes the mutual information $I(X; Z)$ subject to a constraint on the complexity of Z.

AutoEncoders (AE)

Hands-on (download the `Autoencoders.py` from the course website):

- Implement a class for the Autoencoders using PyTorch.
- Train the AutoEncoder on a generated synthetic dataset.
- Visualize the latent space and reconstructed data.
- Experiment with different latent space dimensions and activation functions.
- Compare the results with PCA.
- Do the same with a real dataset (e.g., MNIST).

What's Next?

- Variational AutoEncoders (VAE).
- Generative Adversarial Networks (GAN).
- Applications of generative models.