Neural Networks for Unsupervised Learning

Antonio Fonseca - June 11th 2024

Agenda

- Introduction:
  - When should I use unsupervised learning
- Neural networks for unsupervised learning
  - Autoencoders
  - Variational Autoencoder
- Tutorial
  - Handwritten digits (MNIST)
  - Satelitte image dataset (SAT-6)

Frameworks for Machine Learning

Most real-world problems fall into the category of unsupervised learning

- **"Pure" Reinforcement Learning (cherry)**
  - The machine predicts a scalar reward given once in a while.
  - **A few bits for some samples**
- **Supervised Learning (icing)**
  - The machine predicts a category or a few numbers for each input
  - Predicting human-supplied data
  - **10→10,000 bits per sample**
- **Unsupervised/Predictive Learning (cake)**
  - The machine predicts any part of its input for any observed part.
  - Predicts future frames in videos
  - **Millions of bits per sample**

- **(Yes, I know, this picture is slightly offensive to RL folks. But I'll make it up)**

Original LeCun cake analogy slide presented at NIPS 2016

Extracting knowledge from unlabeled data

- **Understanding Data without Labels**
  - Many real-world datasets are unlabeled
  - Labeling data can be expensive and time-consuming
- **Discovering Hidden Patterns**
  - Reveals underlying structure in data
  - Useful for exploratory data analysis
- **Dimensionality Reduction**
  - Simplifies data for better visualization
  - Reduces computational complexity
- **Data Preprocessing**
  - Helps in feature engineering
  - Improves the performance of supervised learning models

Unsupervised learning techniques

Clustering

- **K-means**
- **Hierarchical Clustering**

Dimensionality Reduction

- **Principal Component Analysis (PCA)**
- **t-Distributed Stochastic Neighbor Embedding (t-SNE)**
- **Uniform Manifold Approximation and Projection (UMAP)**

Autoencoders

- **Autoencoder**
  - Encodes input to a latent space
  - Reconstructs the input from latent space
- **Variational Autoencoder (VAE)**
  - Generates new data points
  - Learns the distribution of input data
  - Reconstructs the input from latent space

Autoencoder (AE)

Neural network trained to copy its input ***x*** to its output

Hidden layer ***z*** describes a *code* to represent the input

Two parts

- Encoder: $z = f(x)$
- Decoder: $x' = g(z)$

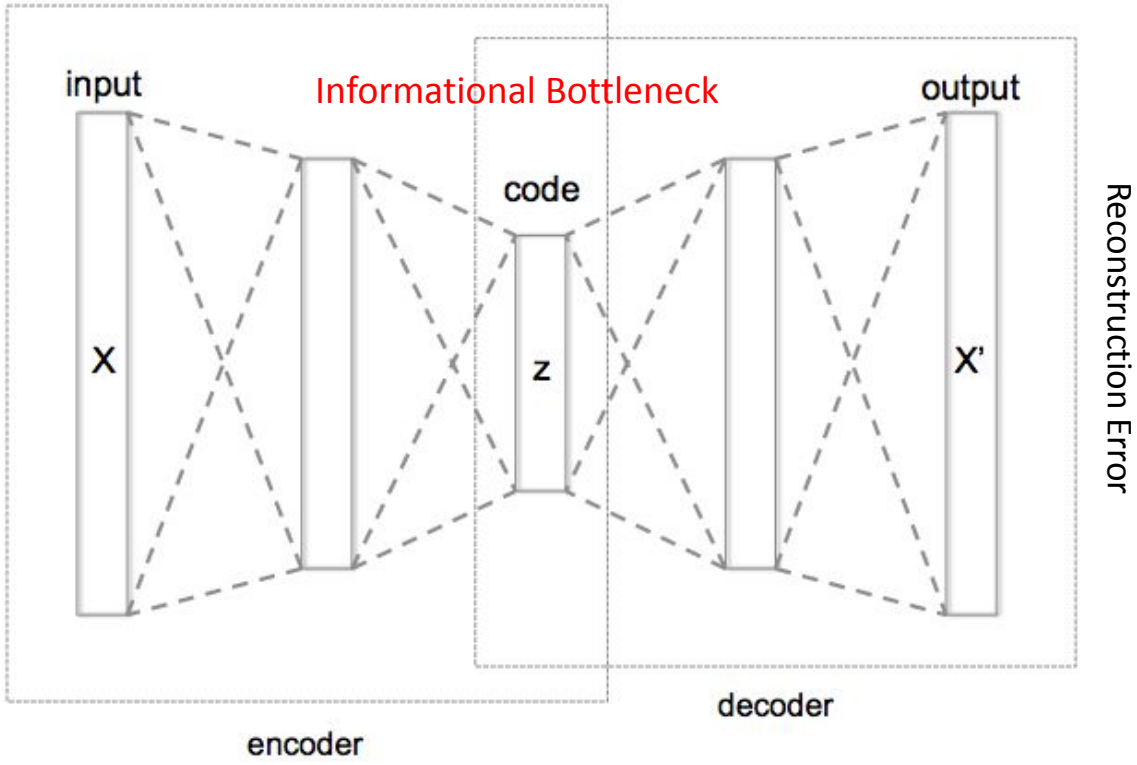
Goal: minimize $L(x, g(f(x)))$

- LLL penalizes $g(f(x))$ for being dissimilar from x
- Example: mean squared error

How do you train an autoencoder?

- Backpropagation

AE architecture

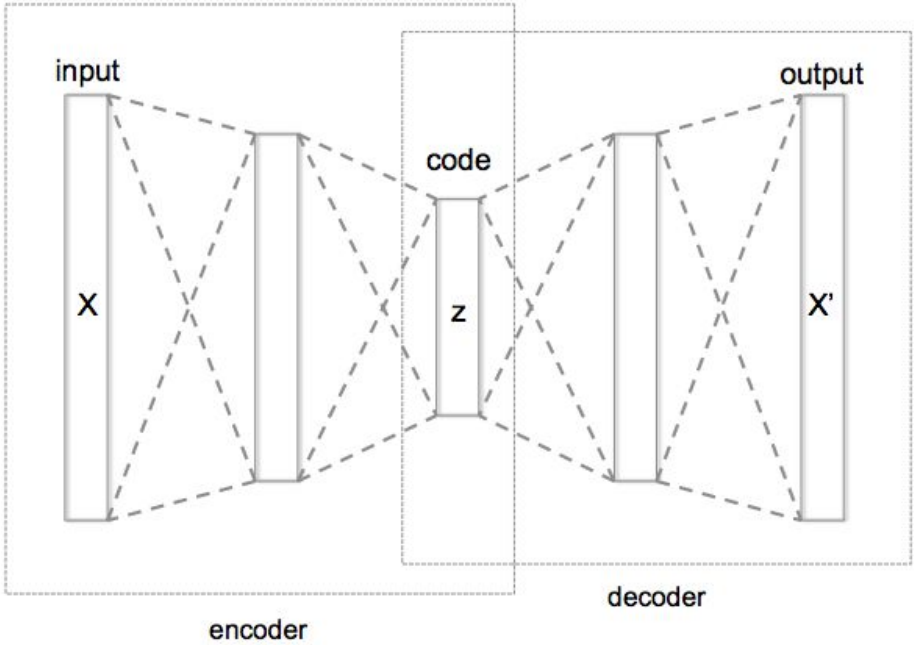


The AE rationale

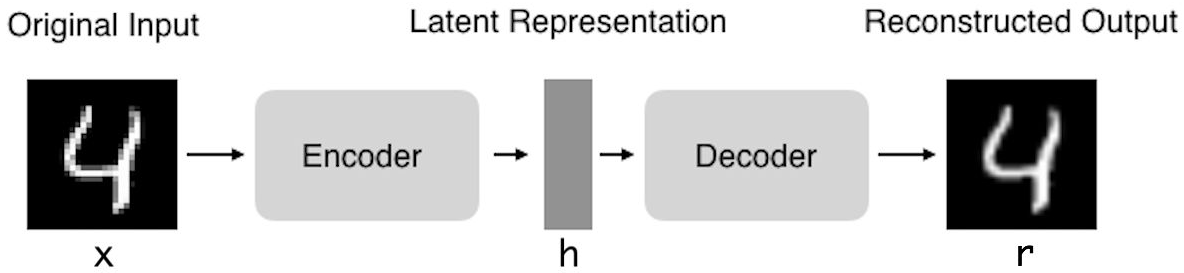
- Can’t we just set $g(f(x)) = x$ everywhere?
  - Not a useful model
- Autoencoders have opposing forces!
  - Reconstruction error drives the encoding to be like the data
  - A different penalty/feature steers it away from copying input
- Design the AE so it **can’t** learn to copy the input perfectly
  - Restrict the AE to copy only approximately
  - Can prioritize certain aspects of the input
- Examples
  - Restrict the size of the code (i.e. add a bottleneck)
  - Add regularization

Autoencoder Applications

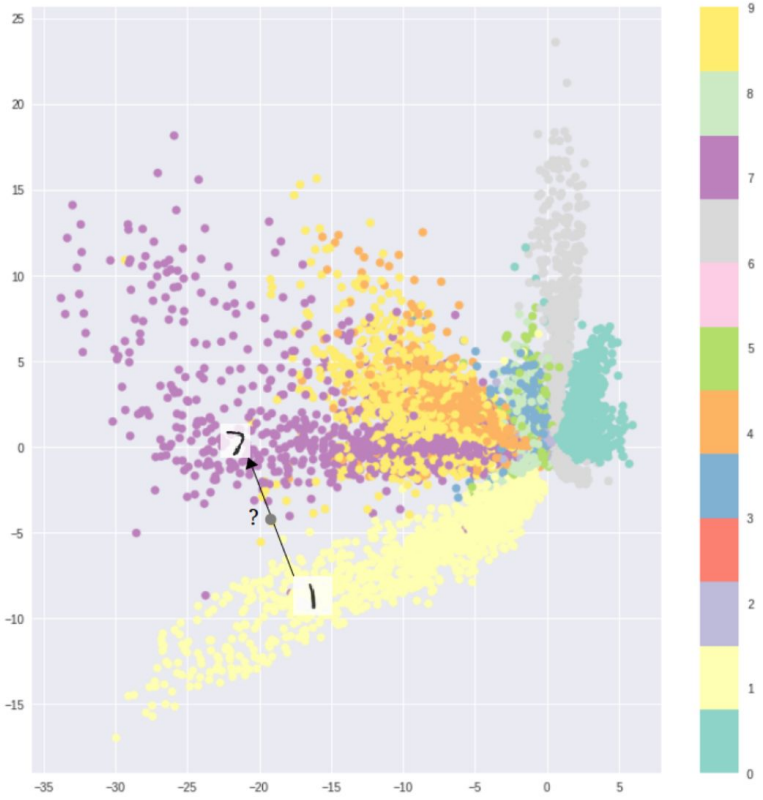
- Dimensionality reduction
  - Including visualization
- Feature learning
- Sparse coding
- Information retrieval
- Data compression
- Data denoising
- Data generation



Autoencoding MNIST



Latent Space Not Optimal for Generation



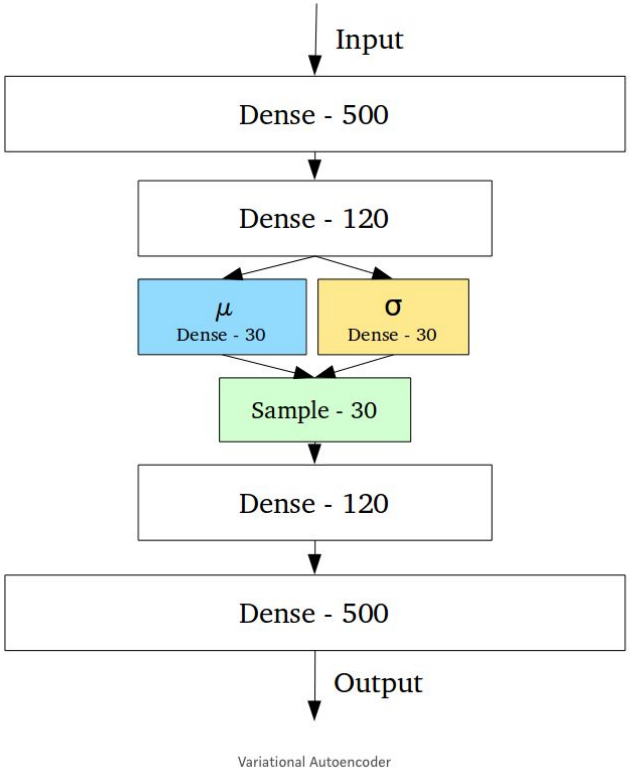
Optimizing purely for reconstruction loss

Formation of clusters helps decoding

Does not help generation

Latent space cannot be interpolated

Variational Autoencoders (VAEs)



Variational Autoencoder

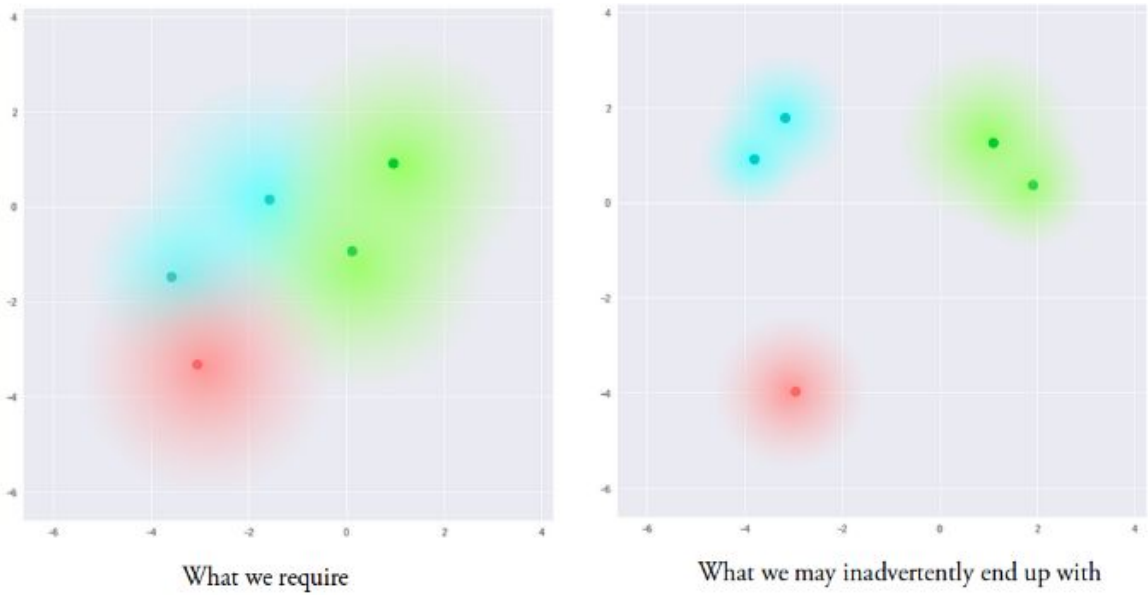
Encoding layer outputs two vectors

Means: $\{\mu_1, \mu_2, \mu_3 \ldots \mu_k\}$
Variances: $\{\sigma_1, \sigma_2, \sigma_3 \ldots \sigma_k\}$

Creates a K-dimensional Gaussian ball for each datapoint

Splits latent space into Gaussian balls

Discouraging Clusters



Solution: Penalize the Gaussian balls if they don’t overlap!

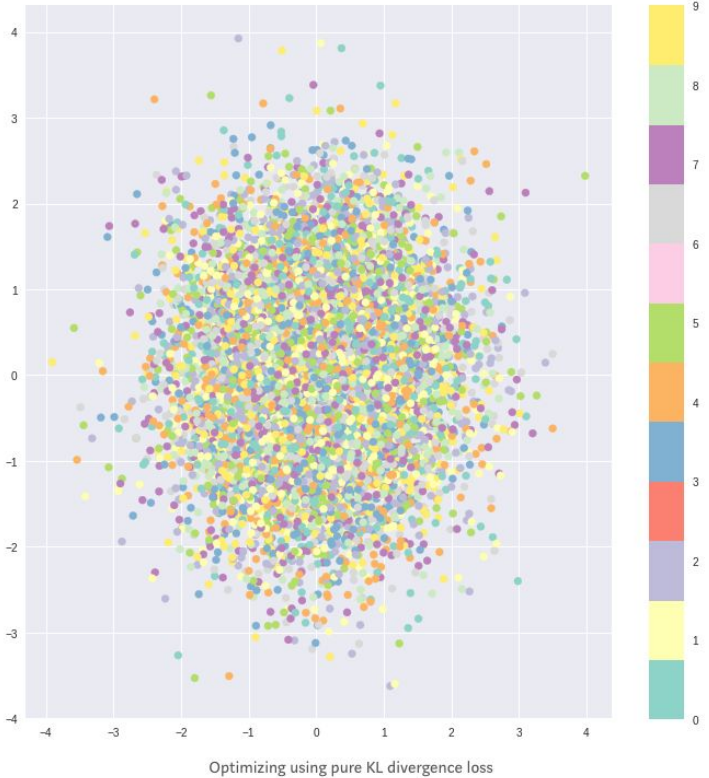
KL Divergence Penalty

$$\mathbb{KL}(q_\theta(z \mid x_i) \mid\mid p(z))$$

Penalizing distributions
In the latent space from being
too far from a standard normal

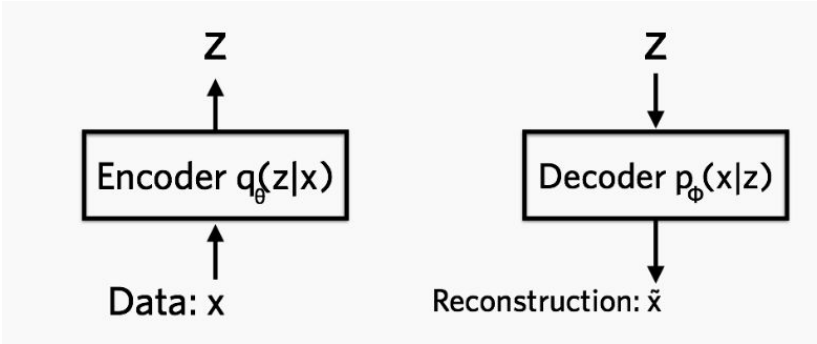
$$p(z) = Normal(0, 1).$$

Encourages every ball to be the same!



Loss Function of VAE

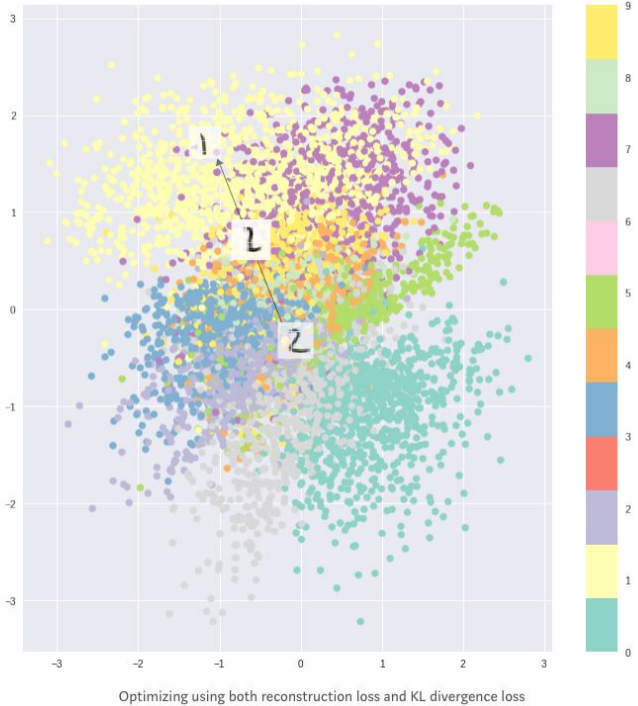
$$l_i(\theta, \phi) = -\mathbb{E}_{z \sim q_\theta(z|x_i)}[\log p_\phi(x_i \mid z)] + \mathbb{KL}(q_\theta(z \mid x_i) \,||\, p(z))$$



Reconstruction penalty encourages separation

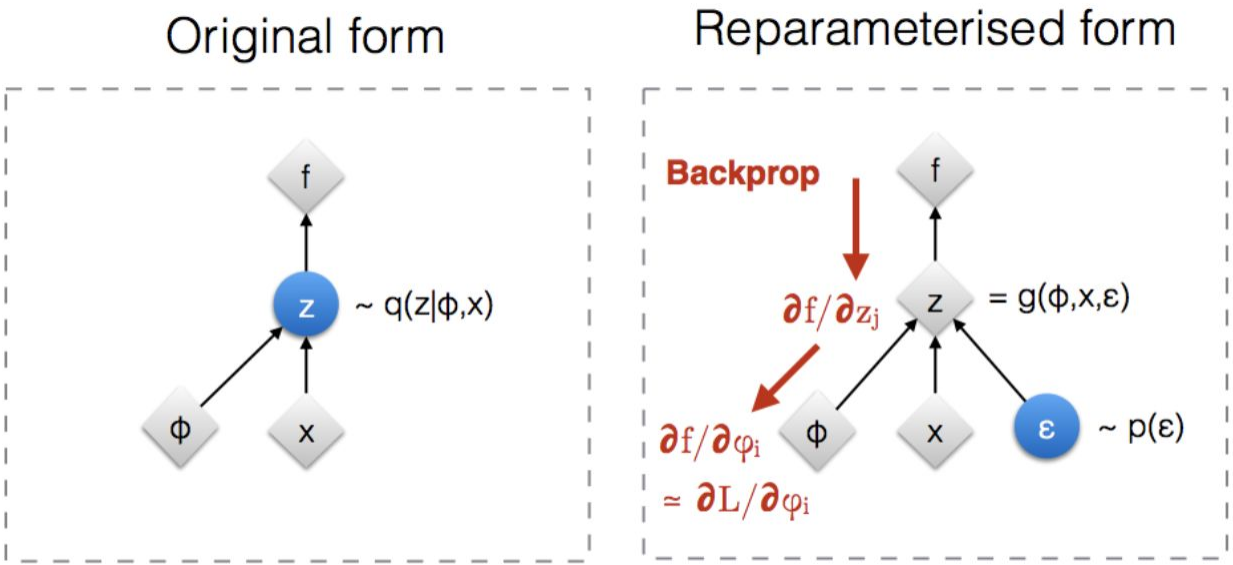
$$-\mathbb{E}_{z \sim q_\theta(z|x_i)}[\log p_\phi(x_i \mid z)]$$

KL+Reconstruction Loss



Optimizing using both reconstruction loss and KL divergence loss

Training a VAE with sampling



Conclusions

Unsupervised Learning

- **No Labeled Data Required**: Efficiently utilizes large datasets without the need for manual labeling.
- **Discover Hidden Patterns**: Identifies underlying structures and relationships in data.

Autoencoders (AEs)

- **Dimensionality Reduction**: Compresses data while retaining essential features.
- **Feature Learning**: Learns representations of data that can be used for other tasks.
- **Reconstruction**: Capable of reconstructing input data from its encoded form.
- **Almost a Generative Model**: Generates new data points within class

Variational Autoencoders (VAEs)

- **Probabilistic Approach**: Models data distribution, providing a probabilistic understanding.
- **Generative Models**: Generates new data points similar to the training data.
- **Interpretable Latent Space**: Enables exploration and manipulation of the latent space for creative applications.