Lecture 10 AutoEncoders (AE) and Generative Adversarial Networks (GAN)

Shiwei Lan[1]

[1]School of Mathematical and Statistical Sciences
Arizona State University

STP598 Machine Learning and Deep Learning
Fall 2021

Table of Contents

AutoEncoder

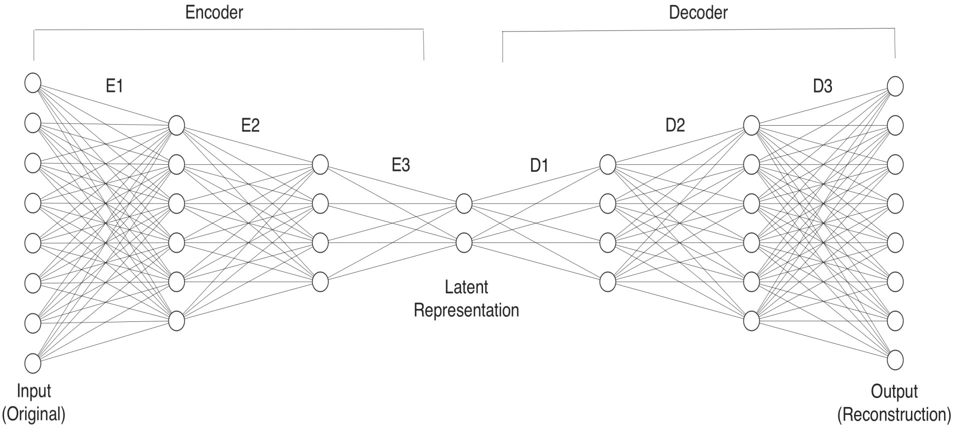


Figure: A typical architecture of autoencoder (AE) neural network.

AutoEncoders

- An autoEncoder (AE) is a neural network that is trained to attempt to copy its input to its output.
- The network consists of two parts:
  1. **encoder**: $f : x \mapsto h$
  2. **decoder**: $g : h \mapsto r$
- The network is trrained to approximately recover (copy) x, i.e. "$r \approx x$".
- The goal of AE is not to perfectly copy, but rather to learn useful (latent) properties of the data!

AutoEncoders

- If the hidden (latent) dimension is smaller than the input dimension, then the AE is called *undercomplete*; otherwise, it is called *overcomplete*.
- The learning process involves minimizing a loss function as follows

$$\min L(x, g(f(x))) \tag{1}$$

  where L is the loss penalizing $g(f(x))$ deviating from x, e.g. mean squared error.
- When the decoder is linear and L is the mean squared error, an undercomplete AE is equivalent to PCA (latent space is spanned by principal directions).
- When f, g are allowed to be nonlinear without constraint, the latent space can be meaningless.

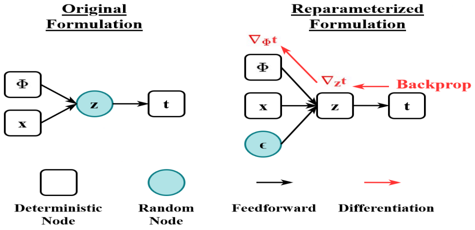
Regularized AutoEncoders

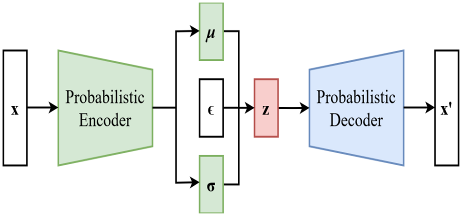
- **Sparse AE** adds sparsity penalty $\Omega(h)$ to the loss: $L(x, g(f(x))) + \Omega(h)$.
- Typical choice of $\Omega(h)$ could be from Laplace prior $p(h; \lambda) = \lambda/2 \exp(-\lambda|h|)$: $\Omega(h) = -\log p(h; \lambda) = \lambda \sum_i h_i$.
- **Denoising AE** minimizes $L(x, g(f(\tilde{x})))$ with $\tilde{x}$ being a copy of x corrupted by some noise and tries to undo such corruption.
- **Contractive AE** regularizes AE with a penalty on the gradients of decoder to learn the distribution of training data:

$$L(x, g(f(x))) + \Omega(h, x), \quad \Omega(h, x) = \lambda \sum_i \|\nabla_x h_i\|^2. \tag{2}$$

Variational AutoEncoder (VAE)

- **Variational AutoEncoder (VAE)** (Kingma and Welling 2014) is probabilistic model for variational Bayesian inference.
- The goal is to approximate posterior distribution $p_\theta(z|x)$ with $q_\Phi(z|x)$ (part of VAE) by minimizing evidence lower bound (ELBO) loss (variation of Kullback–Leibler divergence).
- It reduces to construct a *probabilistic encoder* $q_\Phi(z|x)$ and a *probabilistic decoder* $p_\theta(x|z)$.





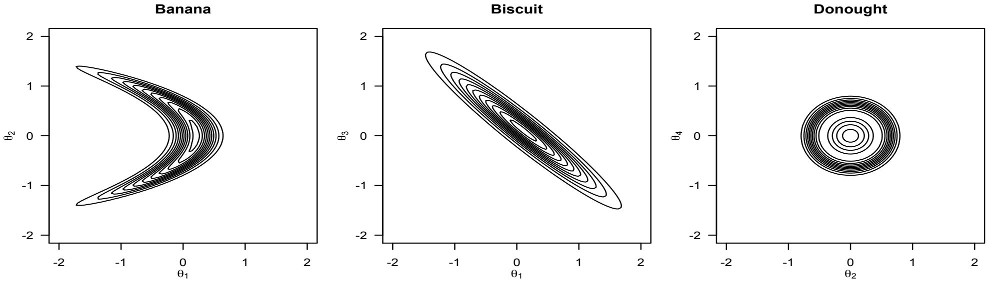
Demonstration: Banana-Biscuit-Doughnut distribution

- Denote parameters $\boldsymbol{\theta} = (\theta_1, \cdots, \theta_D)$. Consider

$$y|\boldsymbol{\theta} \sim \mathcal{N}(\mu_y, \sigma_y^2), \quad \mu_y := \sum_{k=1}^{\lceil D/2 \rceil} \theta_{2k-1} + \sum_{k=1}^{\lfloor D/2 \rfloor} \theta_{2k}^2$$

$$\theta_i \overset{iid}{\sim} \mathcal{N}(0, \sigma_\theta^2), \quad i = 1, \cdots, D$$

- Generate $N = 100$ data points $\{y_n\}_{n=1}^N$ with $\mu_y = 1, \sigma_y^2 = 4$ and $\sigma_\theta^2 = 1$.



Autoencoder HMC Demonstration

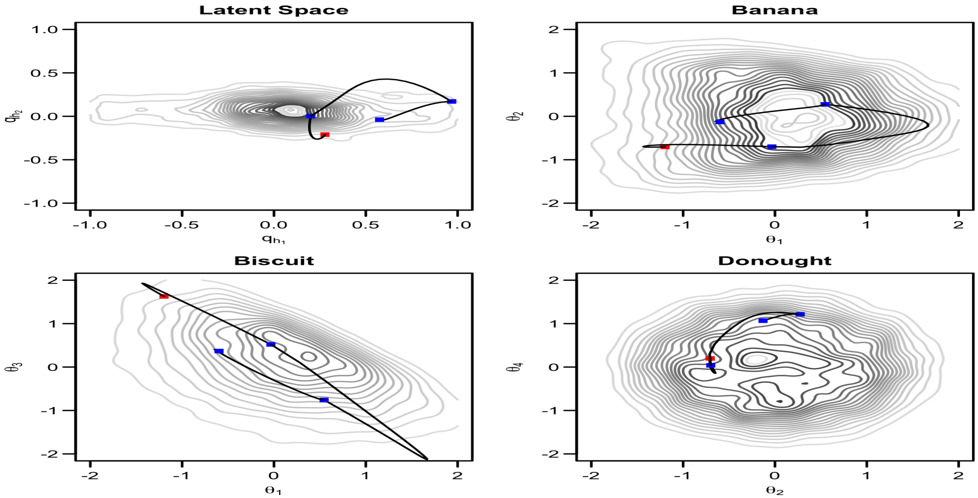


Figure: **Top left:** HMC trajectory in the latent space (2-dimensional); the red square is the initial position, and the blue squares are HMC proposals. **The others:** Trajectories

Table of Contents

Generative Adversarial Networks (GAN)

- A generative adversarial network (GAN) (Goodfellow et al 2014) is a class of machine learning frameworks to generate artificial data that mimic the original data.
- A GAN consists of two neural networks contesting with each other in a zero-sum game (one agent's gain is another agent's loss):
  1. **generator**: $G_\theta(z)$
  2. **discriminator**: $D_\omega(x)$
- Training a GAN reduces to a min-max problem $\inf_\theta \sup_\omega L(\theta,\omega)$. For example, Goodfellow et al (2014) propose the following loss

$$
\begin{aligned}
L(\theta,\omega) &= \mathbb{E}_{X\sim P_r}[\log D_\omega(X)] + \mathbb{E}_{Z\sim P_Z}[\log(1-D_\omega(G_\theta(Z)))] &\quad (3)\\
&= \mathbb{E}_{X\sim P_r}[\log D_\omega(X)] + \mathbb{E}_{X\sim P_{G_\theta}}[\log(1-D_\omega(X))], &\quad (4)
\end{aligned}
$$

- GANs have achieved numerous interesting applications in science, video games, fashion and art, etc..

Generative Adversarial Networks (GAN)

More Reading

- **AE**
  - https://sci2lab.github.io/ml_tutorial/autoencoder/
  - https://towardsdatascience.com/generating-images-with-autoencoders-77fd3a8dd368
  - https://www.tensorflow.org/tutorials/generative/autoencoder
- **GAN**
  - https://machinelearningmastery.com/what-are-generative-adversarial-networks-gans/
  - https://wiki.pathmind.com/generative-adversarial-network-gan
  - https://www.tensorflow.org/tutorials/generative/dcgan