Programming Assignment 5: Deep generative models

Praveen Kandula, ee17d026@smail.iitm.ac.in

Due date: Nov 29th, Friday, 11:55pm

Note:

1. For any questions, please schedule a time with TAs before deadline according to their convenience. Please use moodle dicussion threads for posting your doubts and also check it before mailing to TAs, if the same question has been asked earlier.
2. Submit a single zip file in the moodle named as PA5_Rollno.zip containing report.
3. Read the problem fully to understand the whole procedure.
4. Late submissions will be evaluated for reduced marks and for each day after the deadline we will reduce the weightage by 10%.

Design a generator G that generates data from a desired distribution using multi-layer perceptron (only fully connected layers with non-linearities) which takes a single scalar z as input and outputs a single scalar x. You can use ReLU as the non-linear activation function for the hidden layers, and the output layer does not have any activations.

The input noise value z belongs to the uniform distribution, $z \approx uniform(1, 1)$, andthe generator has to be trained to output a data distribution desired to be Gaussian, i.e. $x_{data} \approx Gaussian(\mu = 2.0, \sigma = 0.2)$.

Complement the generator G with a discriminator D for an adversarial training which also uses multi-layer perceptron (only fully connected layer with non-linearities). D takes a single scalar x or x_{data} as input and outputs a [0,1] value for classification; x is the output from G and x_{data} is a sample from the desired data distribution. You can use ReLU/Tanh as the non-linear activation function for hidden layers and sigmoid for the output layer.

Some tips are given below, but the final choice and tuning is up to the student:

ﬄ The choice of the number of hidden layers is yours; you can try two or three hidden layers for both G and D.

• Use stochastic gradient descent(SGD) to train both G and D.Try a higher learning rate for D than that of G.

• Use minibatch training. Trybatch sizes of 8 or 64.

• Forevery K iterations of descent for D, iterate G once. This "K-times-iterate-D and once-iterate-G" for a minibatch constitutes one epoch. G should mostly give good results within 100 epochs. Try K=10.

1 Submissions

Submit a zip that contains your code with all parameter values, a PDF with your observations, and the plot images aftereach epoch(separately and also as a GIF preferably). A single plot image constitutes all the following plots from (a) to (e) after an epoch. A sample plot after an epoch is given in Figure 1.

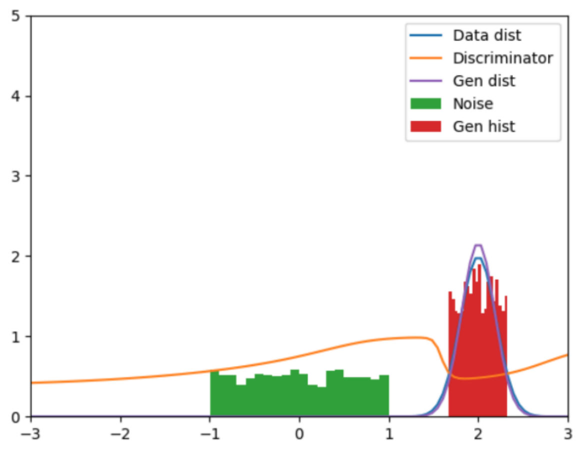
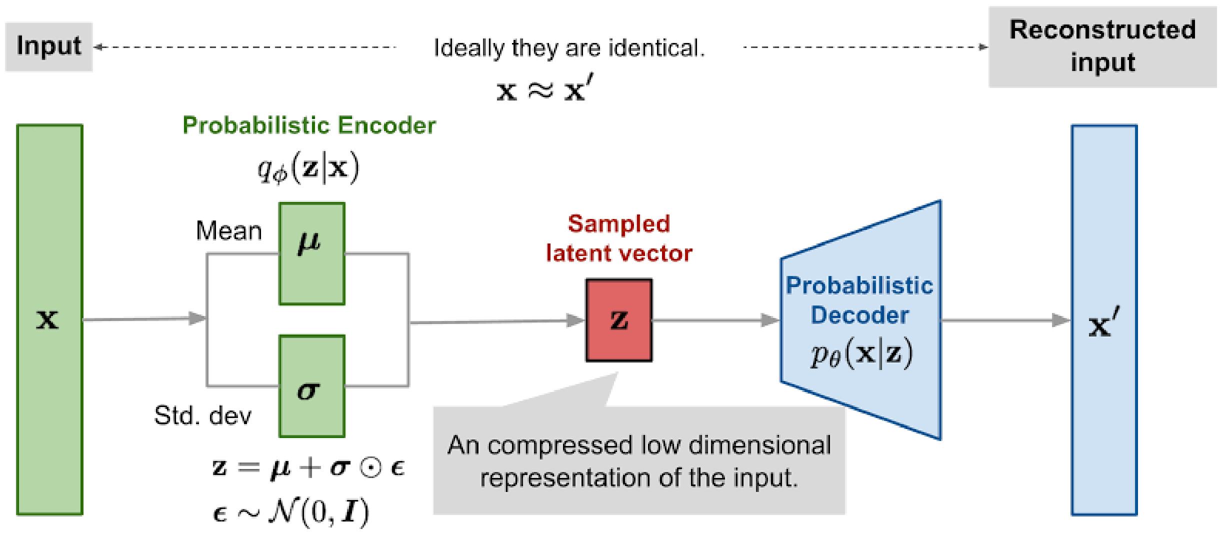


Figure 1: Sample plot after an epoch.

(a) Plot the data distribution Gaussian(μ= 2.0, σ= 0.2) in the range [-3,3]. See blue in Figure1.
(b) Plot the discriminator output for 100 equispaced samples in the range[-3,3].See orange in Figure 1.
• Generate 1000 random noise points as input $z \sim uniform$(-1,1). Generate these points every time newly in eachepoch.
(c) Plot these 1000 random noise points as a histogram with 20 bins. See green in Figure1.
(d) Pass these points through G and plot the histogram (with 20 bins) of its outputs. See red in Figure1.
(e) Calculate the mean and standard deviation for these generator outputs and plot a Gaussian. See purple in Figure1.
• Limit the horizontal axis in [-3,3] and vertical axis in [0,5].

2 Variational auto encoders

Variational autoencoder (VAE) generates data from the desired distribution of ϵ. Below is the design of VAE and loss function to be used for training. The first term in the loss function is the reconstruction loss ensuring the input and generated images are the same. The second term ensures the abstract representation of encoder follows the desired distribution using KL divergence loss.


Input
Ideally they are identical.
$\mathbf{x} \approx \mathbf{x}'$
Reconstructed input
Probabilistic Encoder
$q_\phi(\mathbf{z}|\mathbf{x})$
Mean
$\boldsymbol{\mu}$
$\boldsymbol{\sigma}$
Std. dev
$\mathbf{x}$
Sampled latent vector
$\mathbf{z}$
Probabilistic Decoder
$p_\theta(\mathbf{x}|\mathbf{z})$
$\mathbf{x}'$
An compressed low dimensional representation of the input.
$\mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}$
$\boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{I})$

Structure of variational auto encoder

$$l = -E_{z\sim q(z|x)}[log(p_\theta(x|z)] + D_{KL}(q_\phi(z|x)||p_\theta(z))$$

Use the MNIST data and code vector of size 20 for all the experiments (the mean and variance from encoder should be of size 20)

Design a VAE using only fully connected layers with two hidden layers of size (100, 100), (200, 200), (300, 300) and plot errors for each of the loss functions. Plot images for each of three experiments by randomly sampling from the distribution ϵ using decoder.

Design a VAE using convolution layers and fully connected layers. Use two Conv layers and two fully connected layers. Plot the graphs and generated images similar to the above. Use suitable activation functions and strides.