# CSDS 600: Deep Generative Models

**Homework 1**

**Instructor:** Yu Yin

**Due Date**: **11:59 pm Sept. 24**

**Submission**: Canvas

**1. Autoregressive Models (20/100)**

a. Write out the chain rule for a distribution over $x_1, x_2, \ldots, x_n$.

b. Draw an example of a Masked Autoregressive Distribution Estimation (MADE) model and illustrate the key characteristic that qualify it as proper probability model.

**2. KL-Divergence (20/100)**

Let $\mu: \chi \to \mathbb{R}^k, \sigma: \chi \to \mathbb{R}^k$ and $q(z|x) = N(z; \mu(x), \text{diag}(\sigma^2(x)))$. Suppose that $p(z) = N(z; 0, I)$, show that

$$D(q(z|x) \,||\, p(z)) = \frac{1}{2}\sum_i \sigma_i^2(x) + \mu(x)_i^2 - \log\sigma_i^2(x) - 1.$$

**3. Normalizing Flows (20/100)**

Let $q_0$ be a probability distribution on $\mathcal{Z}$, and define $\boldsymbol{z}_s = g_s(\boldsymbol{z}_{t-1})$ where $g_s: \mathcal{Z} \to \mathcal{Z}$ are invertible functions. Prove that the pushforward distribution on $\boldsymbol{z}_t = g_t \circ \ldots \circ g_1(\boldsymbol{z}_0)$ is given by $q_t$, where

$$\log q_t(\boldsymbol{z}_t) = \log q_0(\boldsymbol{z}_0) - \sum_{s=1}^{t} \text{logdet}\left(\frac{\partial g_s(\boldsymbol{z}_{s-1})}{\partial \boldsymbol{z}_{s-1}}\right).$$

Hint: consider using the inverse function theorem.

**4. Variational Autoencoders (40/100)**

In this question, you will train a VAE model on the MNIST dataset. This dataset consists of 28 × 28 grayscale images. Please implement a standard VAE with the following characteristics:

a. 16-dim latent variables $z$ with standard normal prior $p(z) = N(0, I)$.

b. An approximate posterior $q_\theta(z|x) = N(z; \mu_\theta(x), \sum_\theta(x))$, where $\mu_\theta(x)$ is the mean vector, and $\sum_\theta(x)$ is a diagonal covariance matrix.

c. A decoder $p(x|z) = N(x; \mu_\emptyset(z), I)$, where $\mu_\emptyset(z)$ is the mean vector. (We are not learning the covariance of the decoder)

**Request deliverables**:

a. Record the average full negative ELBO, reconstruction loss, and KL term of the training data (per minibatch) and test data (for your entire test set). Code is provided that automatically plots the training curves.

b. Report the final test set performance of your final model.

c. 100 samples from your trained VAE (put all sample in one figure).

d. 50 real-image / reconstruction pairs (put all sample in one figure).

e. Interpolations of length 10 between 10 pairs of test images from your VAE (100 images total)

Helpful Tips:

- When computing reconstruction loss and KL loss, average over the batch dimension and sum over the feature dimension
- When computing reconstruction loss, it suffices to just compute MSE between the reconstructed and true images. (you can compute the extra constants if you want)

- Use batch size 128, learning rate $10^{-3}$, and an Adam optimizer
- You can play around with different architectures and try for better results, but the following encoder / decoder architecture below suffices.

```
Encoder
    conv2d(3, 32, 3, 1, 1)
    relu()
    conv2d(32, 64, 3, 2, 1) # 16 x 16
    relu()
    conv2d(64, 128, 3, 2, 1) # 8 x 8
    relu()
    conv2d(128, 256, 3, 2, 1) # 4 x 4
    relu()
    flatten()
    linear(4 * 4 * 256, 2 * latent_dim)

Decoder
    linear(latent_dim, 4 * 4 * 128)
    relu()
    reshape(4, 4, 128)
    transpose_conv2d(128, 128, 4, 2, 1) # 8 x 8
    relu()
    transpose_conv2d(128, 64, 4, 2, 1) # 16 x 16
    relu()
    transpose_conv2d(64, 32, 4, 2, 1) # 32 x 32
    relu()
    conv2d(32, 3, 3, 1, 1)
```