# EE641 Homework #2 – Generative Modeling

Author: Zetao Ding   
E-mail: zetaodin@usc.edu  
Date: October 2025

## Overview

This homework explores two deep generative models: (1) a Generative Adversarial Network (GAN) for font image generation, and (2) a Hierarchical Variational Autoencoder (HVAE) for drum pattern generation. Both models demonstrate key principles of probabilistic generation, reconstruction, and representation learning.

## Problem 1 – Font Generation using GAN

The first problem focuses on training a Generative Adversarial Network (GAN) to synthesize font images. The dataset consists of character glyphs in various styles, split into training and validation sets. The generator learns to produce 64×64 grayscale font images conditioned on style and character identity.

### Model Architecture

The GAN consists of a convolutional generator and discriminator. The generator takes random noise and style embeddings to generate character images, while the discriminator learns to distinguish between real and fake images. Both networks are trained in a minimax adversarial setup.

### Training Configuration

Optimizer: Adam (lr=2e-4, β1=0.5, β2=0.999)  
Batch size: 64  
Epochs: 100  
Loss: Binary cross entropy for both generator and discriminator with adversarial updates every batch.

### Results and Discussion

Training stabilized after approximately 80 epochs. The generator progressively learned stroke smoothness, contrast, and overall glyph shape. Validation samples show consistent reconstruction of font style and spacing. Evaluation confirmed realistic image synthesis and mode diversity across different font categories.

### Key Observations

- Early epochs suffered from mode collapse, resolved via balanced training schedule.  
- Applying instance normalization in the generator improved stability.  
- Generated samples exhibit coherent stylistic consistency and legibility.

### Conclusion (Problem 1)

The FontGAN successfully learns to generate diverse and high-quality font glyphs from latent noise. Adversarial training enables the model to generalize across unseen font styles.

## Problem 2 – Hierarchical VAE for Drum Pattern Generation

The second task involves implementing a Hierarchical Variational Autoencoder (HVAE) to model rhythmic patterns. Unlike standard VAEs, this hierarchical version uses two latent spaces to separate global style and local rhythmic detail.

### Model Architecture

Two encoders (high-level and low-level) generate latent variables z\_high and z\_low. The decoder reconstructs 16×9 binary patterns representing drum hits (time × instrument). Reparameterization is applied using z = μ + σ·ε with ε ~ N(0, I).

### Training Setup

Batch size: 128  
Epochs: 100  
Optimizer: Adam (lr=1e-3)  
Hidden size: 256  
z\_high=8, z\_low=16  
KL annealing: linear 0→1  
Loss = BCE + β\_h·KL(z\_h) + β\_l·KL(z\_l).

### Results

The model converged with final metrics: train\_loss=32.73, val\_loss=33.83, recon=31.13, KL\_high=1.26, KL\_low=1.42. Both latent spaces remained active, confirming no KL collapse. Generated drum patterns capture stylistic variation across rock, jazz, hip-hop, and electronic genres.

### Latent Analysis

Interpolation between style prototypes yields smooth transitions, indicating continuous latent manifolds. z\_high captures global style, while z\_low controls per-beat rhythmic nuance.

### Conclusion (Problem 2)

The HVAE effectively learns interpretable hierarchical representations for rhythmic structure. Both quantitative metrics and qualitative inspection confirm strong disentanglement between global and local features.

## Overall Reflection

Across both problems, the experiments demonstrate two key generative modeling paradigms: (1) adversarial image generation with GANs, and (2) probabilistic encoding and decoding with VAEs. Together they highlight the tradeoff between realism (GAN) and interpretability (VAE).