Assignment 2

Harshal Chalke
MS in Artificial Intelligence
Rochester Institute of Technology
Rochester, NY, USA
hc4293@rit.edu

***Abstract*—This assignment explores Generative Adversarial Networks (GANs) and Variational Autoencoders (VAEs), focusing on their theoretical foundations, training methodologies, and applications within imaging science. Emphasis is placed on GANs' min-max objective, discriminator-generator interplay, and DCGAN advances, alongside VAE latent space structure and ELBO optimization. Real-world applications demonstrate GANs' and VAEs' utility in data augmentation, anomaly detection, and image generation.**

***Index Terms*—Generative Adversarial Networks, Variational Autoencoders, Image Generation, Anomaly Detection, Deep Learning.**

I. Generative Adversarial Networks (GANs)

A. Theory and Objective of GANs:

1) Objective Function in GANs and Its Min-Max Structure

The primary objective function of Generative Adversarial Networks (GANs) is structured as a min-max optimization problem between two competing neural networks: the generator (G) and the discriminator (D). This objective is defined as:

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{\text{data}}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1-D(G(z)))] \quad (1)$$

This min-max adversarial framework drives both G and D to iteratively improve, resulting in the generator producing increasingly realistic samples as it learns from the discriminator's feedback.

- The **discriminator** D attempts to correctly classify real data from the true data distribution $p_{\text{data}}(x)$ as real, outputting values close to 1. For generated samples $G(z)$ (from noise z), D tries to classify them as fake, outputting values close to 0.
- The **generator** G aims to produce realistic samples that "fool" D, maximizing $D(G(z))$ so that it approaches 1.

2) Roles of the Generator and Discriminator in GANs

In GANs, the generator and discriminator have distinct yet interdependent roles:

- The **generator** G creates data from random noise z, sampled from a distribution p_z, and learns to generate samples similar to real data. It minimizes its loss by generating samples that D classifies as real.
- The **discriminator** D evaluates both real and generated data, outputting a probability that a given sample is real, effectively learning to distinguish between real and fake samples.

Through this adversarial process, the generator iteratively improves its output quality, while the discriminator becomes increasingly adept at distinguishing real from generated samples.

3) Challenges When the Discriminator Becomes Too Powerful

A common issue in GAN training arises when the discriminator D becomes too powerful, accurately distinguishing real from fake samples with ease. This imbalance provides minimal gradient information to G, effectively "winning" the game and halting the generator's improvement. This issue can lead to mode collapse, where the generator produces limited variations of samples due to insufficient learning feedback.

Several techniques can mitigate this issue:

- **Label Smoothing**: Adjusts the labels for real and fake samples to soften the discriminator's learning, preventing overconfidence.
- **Modified Loss Functions**: Using alternative objective functions, such as Wasserstein GAN (WGAN), improves stability by altering the optimization landscape.
- **Training Adjustments**: Techniques like alternating updates, reducing D's learning rate, adding dropout between layers or employing batch normalization help maintain balanced learning dynamics between D and G.

B. Training Strategies for GANs

1) Steps Involved in Training a GAN and Optimization of Losses

The following describes the core steps and optimization process:

1) **Sample Real and Noise Data**: Real samples are drawn from the actual data distribution p_{data}. Noise samples, drawn from a prior distribution p_z (e.g., Gaussian), are used as input to the generator to produce synthetic data.
2) **Update the Discriminator**: The discriminator is trained to distinguish between real samples and fake samples generated by G. This is done by maximizing the probability of correctly classifying real samples as real and

generated samples as fake. The discriminator's loss L_D is given by:

$$L_D = -\left(\mathbb{E}_{x\sim p_{\text{data}}}[\log D(x)] + \mathbb{E}_{z\sim p_z}[\log(1 - D(G(z)))]\right) \tag{2}$$

3) **Update the Generator**: After updating D, the generator is updated to produce samples that maximize the probability of being classified as real by D. This is achieved by minimizing the generator's loss function L_G:

$$L_G = -\mathbb{E}_{z\sim p_z}[\log D(G(z))] \tag{3}$$

4) **Iterate**: Steps 2 and 3 are repeated iteratively. As the generator improves, the discriminator faces increasingly challenging tasks in distinguishing real from fake samples, driving both models to continuously refine their capabilities.

Through this adversarial optimization, both models enhance their performance over time.

2) Role of Strided Convolutions in Deep Convolutional GANs (DCGANs)

Deep Convolutional GANs (DCGANs) enhance traditional GANs by integrating convolutional layers, specifically using strided convolutions instead of fully connected or pooling layers. Strided convolutions offer the following benefits:

- **Improved Feature Extraction**: Strided convolutions enable the network to down-sample and up-sample, capturing spatial hierarchies in the data more effectively than pooling layers. This improves the generator's ability to create fine-grained details in generated images.
- **Training Stability**: By preserving spatial relationships in the data, DCGANs improve training stability and reduce artifacts in generated images. This is essential in applications like image generation where coherent spatial patterns are crucial.
- **Enhanced Gradient Flow**: Strided convolutions facilitate better gradient propagation throughout the network, which enhances convergence stability and supports the simultaneous improvement of both G and D.

These modifications make DCGANs particularly suited for tasks like image generation, where capturing fine details and maintaining stable training dynamics are necessary for producing high-quality samples.

C. Applications of GANs

1) Real-World Imaging Applications of GANs and Their Advantages

Generative Adversarial Networks (GANs) offer transformative solutions for imaging tasks that require realistic data generation, enhancement, and synthesis. Key applications include:

- **Data Augmentation in Medical Imaging**: GANs create synthetic medical images, like MRI or CT scans, to augment datasets, addressing class imbalance and improving robustness in disease classification, especially when real data is limited.
- **Image-to-Image Translation in Autonomous Driving**: GANs enable conversions such as day-to-night or weather variations in driving scenes, making models more adaptable to diverse conditions. Techniques like Pix2Pix and CycleGAN perform realistic domain translations, enhancing system resilience.
- **Super-Resolution for Satellite Imagery**: GANs, especially SRGAN, improve satellite image resolution, aiding applications in urban planning, agriculture, and disaster management by providing detailed visuals.

GANs thus provide high-quality image generation, domain translation, and resolution enhancement, expanding possibilities in imaging science.

2) Latent Space Interpolation in GANs

GANs use a latent space z that enables controlled variation in generated images. **Latent space interpolation** involves smoothly transitioning between two points z_1 and z_2 in this space, generating intermediate images that blend features from both.

This technique supports applications like style transfer or facial image generation, where interpolation produces a sequence of images showing gradual changes, such as transitioning expressions. Latent space interpolation highlights GANs' ability to generate diverse outputs from a continuous, structured space.

II. Variational Autoencoders (VAEs)

A. Theory of VAEs

1) Differences Between Standard Autoencoders and Variational Autoencoders

Standard autoencoders and variational autoencoders (VAEs) are both used for dimensionality reduction and data generation, but they differ in their approach to the latent space. A standard autoencoder has a deterministic latent space, encoding input data into a fixed representation and reconstructing the original data. It minimizes reconstruction error but lacks a structured, probabilistic latent space, limiting its generative capabilities.

In contrast, a **Variational Autoencoder (VAE)** introduces a probabilistic framework by encoding input data as a distribution (mean μ and variance σ^2) in the latent space, from which a latent variable z is sampled. This regularization of the latent space enables VAEs to generate new samples by sampling from the learned distribution.

The **latent variable** z serves as a probabilistic representation of the input, capturing underlying features of the data distribution. This design allows VAEs to perform smooth interpolation and variation, as the latent space is continuous and interpretable, supporting generative tasks effectively.

2) Reparameterization Trick in VAEs and Its Purpose

In a VAE, sampling z directly from a learned distribution (defined by μ and σ^2) would make the model's optimiza-

tion non-differentiable, as sampling introduces randomness that disrupts gradient-based backpropagation. To address this, VAEs use the **reparameterization trick**, a technique that enables differentiable sampling by expressing z as:

$$z = \mu + \sigma \cdot \epsilon \tag{4}$$

where $\epsilon \sim \mathcal{N}(0, 1)$ is a random variable sampled from a standard normal distribution. This transformation allows gradients to flow through μ and σ, making it possible to optimize the VAE using standard backpropagation.

The reparameterization trick is essential for training VAEs, as it provides a workaround to the otherwise non-differentiable sampling process, allowing the network to learn meaningful representations in the latent space that are both continuous and structured. This, in turn, improves the VAE's generative capacity, as it enables smooth sampling and interpolation across the latent space.

B. Loss Function of VAEs

1) Derivation of the Evidence Lower Bound (ELBO)

In Variational Autoencoders (VAEs), the objective is to maximize the likelihood $\log p_\theta(x^{(i)})$. Since the true posterior $p_\theta(z|x^{(i)})$ is intractable, we approximate it with $q_\phi(z|x^{(i)})$. Using Jensen's inequality, we obtain the Evidence Lower Bound (ELBO), which serves as a tractable objective function:

$$\log p_\theta(x^{(i)}) = \mathbb{E}_{z\sim q_\phi(z|x^{(i)})}\left[\log \frac{p_\theta(x^{(i)}, z)}{q_\phi(z|x^{(i)})}\right] \tag{5}$$

Applying Bayes' rule and rearranging terms, we get:

$$\begin{aligned}\log p_\theta(x^{(i)}) \geq \mathbb{E}_{z\sim q_\phi(z|x^{(i)})}[\log p_\theta(x^{(i)}|z)] \\ -D_{KL}(q_\phi(z|x^{(i)})||p(z))\end{aligned} \tag{6}$$

The ELBO comprises:

- $\mathbb{E}_{z\sim q_\phi(z|x^{(i)})}[\log p_\theta(x^{(i)}|z)]$: the reconstruction term, which ensures the generated sample resembles the input.

- $D_{KL}(q_\phi(z|x^{(i)})||p(z))$: the KL-divergence, which regularizes $q_\phi(z|x)$ to be close to the prior $p(z)$, typically a standard Gaussian.

2) Importance of KL-Divergence in VAEs and Its Impact on Sample Quality

The KL-divergence term in the ELBO encourages the approximate posterior $q(z|x)$ to be close to the prior distribution $p(z)$, which is typically set as a standard normal distribution. Minimizing KL-divergence has two main benefits:

- **Regularized Latent Space**: By enforcing similarity between $q(z|x)$ and $p(z)$, the KL-divergence regularizes the latent space, making it continuous and smooth. This continuity is essential for generating high-quality samples, as points sampled from $p(z)$ will correspond to realistic data points after decoding.
- **Improved Generalization**: The KL-divergence constraint prevents $q(z|x)$ from diverging too far from $p(z)$, reducing overfitting and improving generalization. This ensures that the VAE can generate meaningful and high-quality samples even when generating data points outside the training distribution.

C. Applications of VAEs

1) Anomaly Detection Using VAEs

VAEs are well-suited for anomaly detection due to their ability to learn a probabilistic representation of normal data patterns. During training, VAEs reconstruct typical input data, encoding it into a structured latent space. At inference, anomalies show high reconstruction errors, as the model cannot represent them accurately. This high error effectively signals outliers in fields like medical imaging, network security, and quality control.

2) Comparison of VAEs and GANs for Image Generation

VAEs and GANs serve different roles in image generation:

- **VAEs**: Preferred for interpretability and continuous latent spaces, enabling smooth interpolations and controlled image generation. Ideal for tasks needing probabilistic representation, such as anomaly detection and semi-supervised learning.
- **GANs**: Known for producing highly realistic images, often superior to VAEs in visual quality. Suitable for applications prioritizing realism, like media and entertainment, where interpretability is less critical.

When image quality is critical, GANs are favored; VAEs are preferred for tasks needing structured latent spaces or probabilistic inference.

III. Part 2

A. Generative Adversarial Networks (GANs)

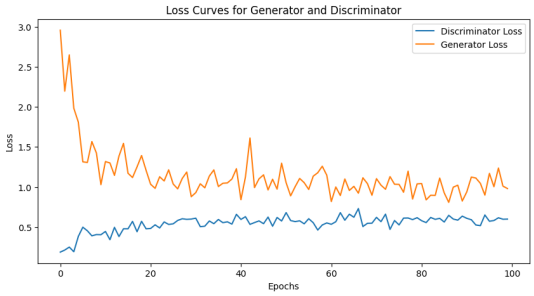

Fig. 1. Generator and Discriminator loss curves for fcGAN

Fig. 1 highlights the sensitivity of GANs to hyperparameters, with noticeable instability in training. The generator's loss decreases over time, indicating improvement, while the discriminator's loss remains low but fluctuates, reflecting the adversarial nature of GAN training.

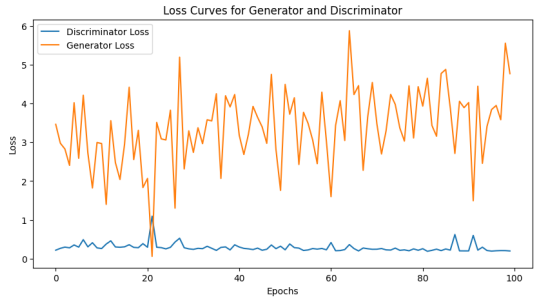

Fig. 2. Generator vs. Discriminator loss for dcGAN

Fig. 2 displays greater instability compared to the fully connected GAN, with a stable discriminator loss and random fluctuations in the generator loss. A challenge encountered was that increasing layers in the generator reduced feature richness, leading to degraded output quality.

Fig. 3. Latent space Interpolation for fcGAN (MNIST)

Fig. 3 demonstrates latent space interpolation in the fcGAN. The interpolation smoothly transitions between different digit shapes, reflecting the continuity of the learned latent space.

B. Variational Autoencoders (VAE)

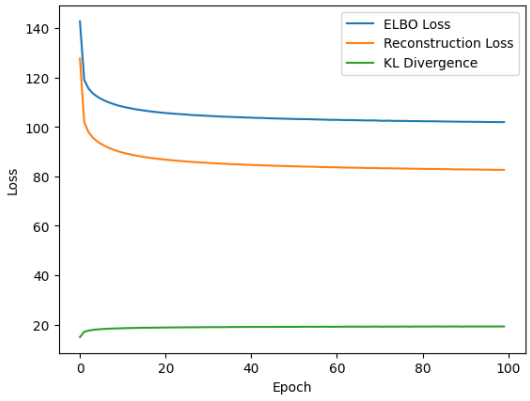

Fig. 4. VAE losses

Fig. 4 shows the ELBO loss decomposed into reconstruction loss and KL divergence. The reconstruction loss converges quickly, while the KL divergence stabilizes, ensuring a structured latent space for effective generation.

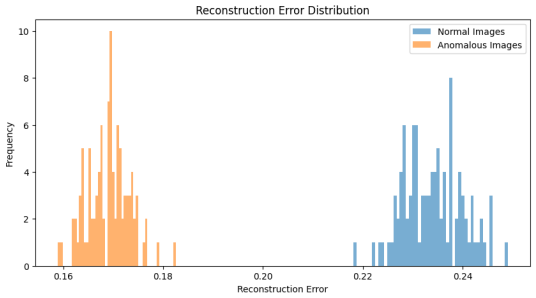

Fig. 5. Anamoly Detection

Fig. 5 anomalies have higher reconstruction errors, aiding in detection.

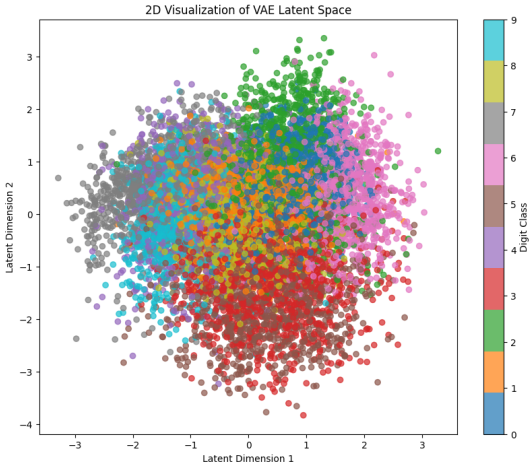

Fig. 6. Latent Space Visualization

Fig. 6 presents the 2D latent space of the VAE, with clear clustering by digit class.

C. Project Repository

The implementation of GANs and VAEs used in this report can be found on GitHub:

IMGS Assignment 2