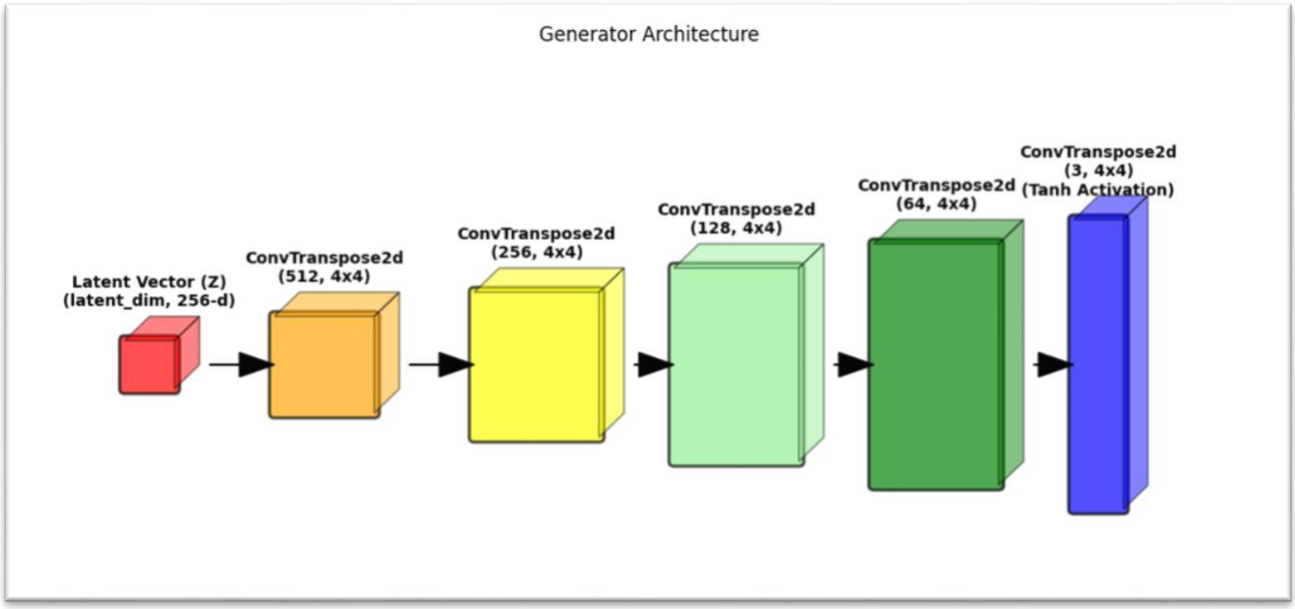
HW4 - Generative Models

1. Model Architecture: Generative Adversarial Network (GAN)

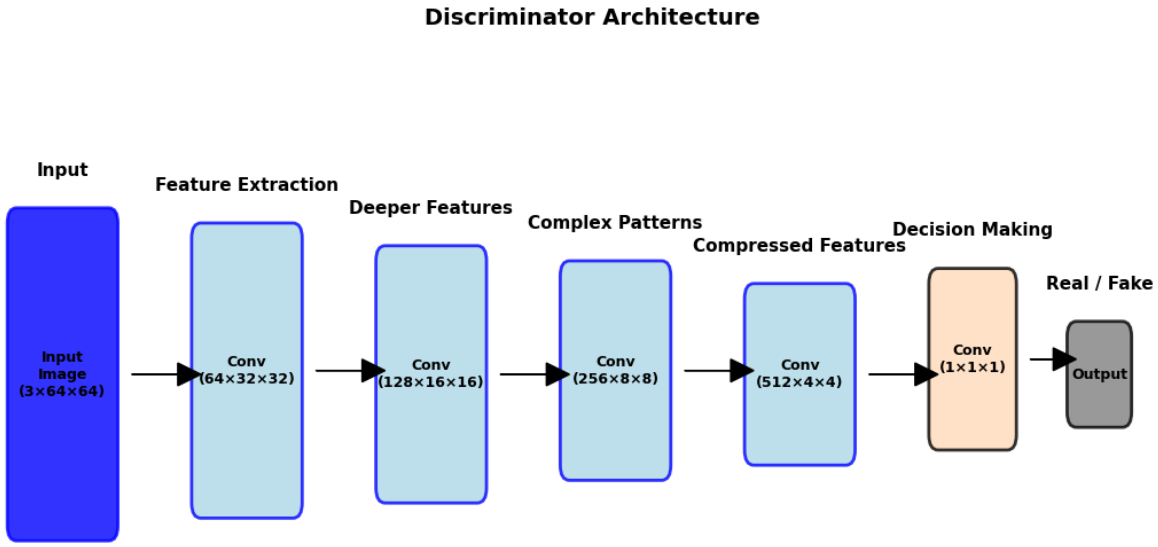
Generator Architecture

- **Input**: A 256-dimensional latent vector Z
- **Deconv 1**: 4×4×512 (ConvTranspose2d + BatchNorm + ReLU)
- **Deconv 2**: 8×8×256 (ConvTranspose2d + BatchNorm + ReLU)
- **Deconv 3**: 16×16×128 (ConvTranspose2d + BatchNorm + ReLU)
- **Deconv 4**: 64×64×32 (ConvTranspose2d + Tanh activation)
- **Output**: A 64×64×3 RGB image



Discriminator Architecture

- **Input**: 64×64×3 RGB image
- **Conv 1**: 32×32×64 (Conv2d + LeakyReLU)
- **Conv 2**: 16×16×128 (Conv2d + BatchNorm + LeakyReLU)
- **Conv 3**: 8×8×256 (Conv2d + BatchNorm + LeakyReLU)
- **Conv 4**: 4×4×512 (Conv2d + BatchNorm + LeakyReLU)
- **Conv 5**: 1×1×1 (Conv2d)
- **Output**: A single value indicating the probability of the input being real or fake.



2. Hyperparameters

- **Latent Space Dimension**: z = 256
- **Batch Size**: 256
- **Number of Epochs**: 300
- **Learning Rate**: 0.0003
- **Adam Optimizer**: o β1 = 0.5
  - β2 = 0.999

3. Optimization & Regularization

- **Weight Initialization**:
  - Normal distribution with mean 0 and std 0.02 for convolutional layers.
  - BatchNorm layers initialized with mean 1 and std 0.02.
- **Leaky ReLU in Discriminator**: Slope = 0.2
- **Batch Normalization**: Applied to stabilize training.

4. Transforms Details

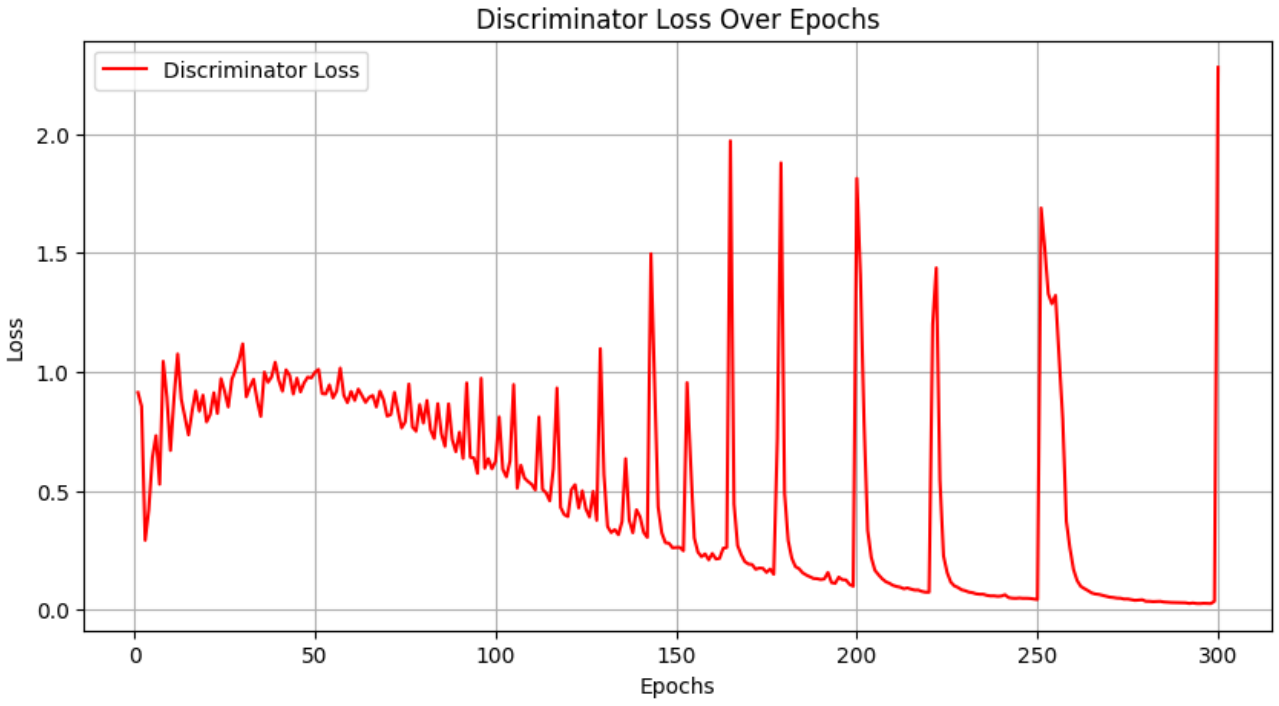
- **Dataset**: Images were preprocessed using Resize (64x64), Normalize ([0.5, 0.5, 0.5]).

Discriminator Loss Analysis

Initially, the loss starts at a relatively high value, indicating that the discriminator struggles to differentiate between real and generated samples. As training progresses, the loss shows a general decreasing trend, reflecting the generator's improvement in producing more realistic samples that challenge the discriminator.

However, the loss exhibits significant fluctuations and periodic spikes. These spikes often occur when the discriminator temporarily dominates, likely due to changes in the generator's output that the discriminator can easily identify as fake. This behavior is common in GAN training due to the adversarial dynamics between the generator and discriminator.

Despite these fluctuations, the overall trend of the loss moving downward suggests that the generator is improving in fooling the discriminator over time. The presence of large spikes and sharp dips highlights the unstable and oscillatory nature of GAN training, where the two networks continuously compete to outperform each other.

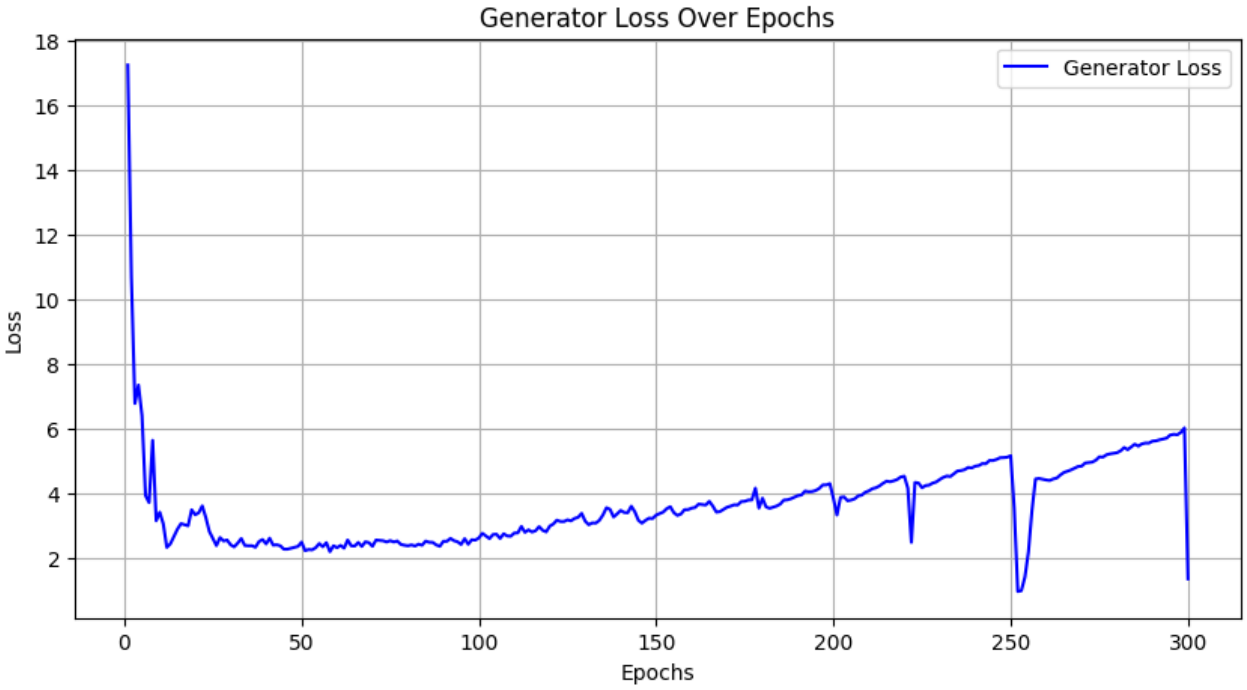


Generator Loss Analysis

At the beginning of the training process, the generator loss starts at a very high value, suggesting that the generator initially struggles to produce outputs that can deceive the discriminator. This high loss rapidly decreases during the first 50 epochs, indicating a period of significant learning and adaptation by the generator as it improves its ability to produce more convincing samples.

After this steep decline, the loss stabilizes, maintaining a relatively steady value. This plateau suggests that the generator's improvements have slowed as it reaches a balance in the adversarial game with the discriminator. However, small fluctuations persist throughout the remaining epochs, which are characteristic of the adversarial training process. These oscillations reflect the dynamic nature of the GAN, where the generator and discriminator continuously challenge each other.

In the later stages of training, noticeable spikes and dips appear, showing moments of temporary instability. These could indicate instances where the generator's adjustments temporarily make its outputs either more or less convincing to the discriminator. Despite these fluctuations, the loss does not return to the initial high levels, indicating overall progress in the generator's ability to refine its outputs over time.



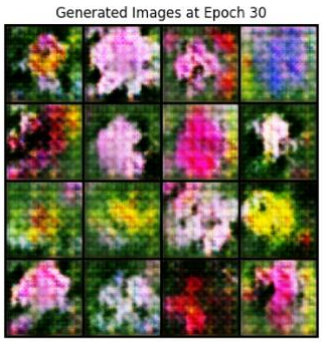
Summary of Attempts and Conclusions

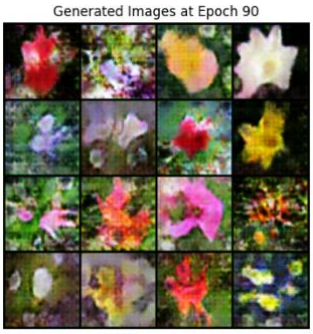
1. Overview

The model was evaluated under two different settings of the **latent space dimension**: **64** and **256**. We will show the differences between them.

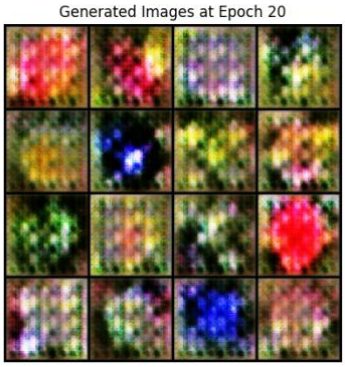
2. Comparison of Results

2.1 Latent Dimension = 64





2.2 Latent Dimension = 256



Latent Space Visualization:

This image demonstrates a latent space interpolation, where the GAN transitions smoothly between two different points in the latent space, resulting in gradual changes in the generated flower. Starting from a light pink flower on the left, the images progressively shift in shape and color, transitioning to a vibrant white flower on the right. This interpolation highlights the GAN's ability to capture a continuous and meaningful representation of the data distribution in its latent space.

Differences between 256-dimensional latent space and 64-dimensional latent space:

The differences between the images generated by the 256-dimensional latent vector model and the 64-dimensional latent vector model can be observed in several aspects, including image quality, diversity, realism, training stability, and computational complexity.

In terms of image quality, the 256-dimensional latent space produces images with significantly more detail and richness. The generated flowers exhibit fine structures, such as petals and textures, which are well-defined and visually appealing. Colors blend smoothly, creating a cohesive and natural look. On the other hand, the images generated by the 64-dimensional latent space lack these intricate details and appear blurrier. The flowers' shapes and structures are less distinct, and the blending of colors is less refined, sometimes resulting in unrealistic or abstract appearances.

Another key difference is the diversity in the generated images. The 256-dimensional model demonstrates a wider range of variations in terms of color combinations, shapes, and textures. The larger latent space allows the generator to capture a broader spectrum of features, leading to more diverse outputs. In contrast, the 64-dimensional model is more constrained by its smaller latent space, resulting in less variety. The images often repeat similar patterns or features, limiting the overall uniqueness of the outputs.

When considering realism, the 256-dimensional latent space excels at producing flower images that resemble real-world representations. The model captures subtle nuances such as shading, proportions, and realistic textures, making the images more believable. However, the 64-dimensional model often struggles to achieve this level of realism. The generated flowers sometimes appear overly abstract or distorted, with forms that deviate significantly from natural flower structures.

Training stability and the capacity to represent features also differ between the two models. The higher-dimensional latent space of the 256-dimensional model gives the generator greater flexibility to capture intricate patterns, resulting in stable and detailed outputs even at higher epochs, such as epoch 300. Conversely, the 64-dimensional model is more prone to limitations in representing complex features, leading to oversimplified outputs. Training instability can also manifest in the form of repetitive patterns or artifacts in the generated images.

Lastly, there is a trade-off in computational complexity between the two models. The 256-dimensional model requires more computational resources for training and inference due to its larger latent space and higher dimensionality. Meanwhile, the 64-dimensional model is less computationally intensive, making it a more efficient choice for scenarios where resources are limited, though this comes at the cost of reduced image quality and diversity.

In summary, the 256-dimensional model excels in generating higher-quality, diverse, and realistic flower images, albeit with greater computational demands. The 64-dimensional model, while more efficient, produces simpler, less diverse, and less detailed outputs.