CPSC 8430 Homework 4: GANs

Rayid

1 Introduction

In this assignment, I implemented and compared three different Generative Adversarial Network (GAN) architectures: Deep Convolutional GAN (DCGAN), Wasserstein GAN (WGAN), and Auxiliary Classifier GAN (ACGAN). The goal was to generate high-quality images using these GAN variants and analyze their performance. The code for this assignment can be found on my GitHub repository: https://github.com/rayidali/CPSC8430-HW4.git

2 Methodology

2.1 DCGAN

- Implemented the generator and discriminator networks using deep convolutional layers.
- Trained the networks adversarially to generate realistic images.
- Analyzed the loss curves and generated image quality during training.

2.2 WGAN

- Modified the GAN architecture to use the Wasserstein loss function.
- Implemented gradient penalty to stabilize training and improve convergence.
- Evaluated the generated image quality and compared it with DCGAN.

2.3 ACGAN

- Extended the GAN architecture by incorporating an auxiliary classifier in the discriminator.
- Trained the generator to generate images conditioned on class labels.
- Analyzed the impact of the auxiliary classifier on image quality and diversity.

3 Results and Analysis

3.1 DCGAN

- The training loss for the generator and discriminator converged to 1.2367 and 0.6752, respectively.
- The minimum generator loss achieved was 0.3925, and the minimum discriminator loss was 0.0051.
- The Fréchet Inception Distance (FID) score ranged from 206.1921 to 439.1560, with a mean of 267.6772.
- The final FID score was 246.7291, indicating a reasonable level of image quality.

3.2 ACGAN

- The training loss for the generator and discriminator converged to 3.4066 and 1.2583, respectively.
- The minimum generator loss achieved was 1.6765, and the minimum discriminator loss was 0.8862.
- The FID score ranged from 281.9992 to 455.3881, with a mean of 351.5907.
- The FID score for the last 5 epochs was 324.3262, suggesting a higher level of image quality compared to DCGAN.

4 Optimization Strategies

- Experimented with different learning rates and batch sizes to stabilize training and improve convergence.
- Used techniques like batch normalization and leaky ReLU activations to enhance the generator's performance.
- Implemented gradient clipping and regularization techniques to prevent mode collapse and improve diversity.

5 Implementation Details

- Utilized PyTorch framework for implementing the GAN architectures.
- Employed convolutional layers in the generator and discriminator networks for efficient image generation and discrimination.
- Used the CIFAR-10 dataset for training and evaluation.

6 Comparison Metrics

- Evaluated the generated image quality using the Fréchet Inception Distance (FID) metric.
- Compared the loss curves and convergence behavior of the generator and discriminator networks.
- Analyzed the diversity and realism of the generated images through visual inspection.

7 Conclusion

In this assignment, I successfully implemented and compared three GAN variants: DCGAN, WGAN, and ACGAN. The results demonstrate the effectiveness of these architectures in generating realistic images. ACGAN achieved a higher level of image quality compared to DCGAN, as evident from the lower FID scores.