Project Report: Generating Prototype Images of Notable Figures Using GANs

Course: CSE445, Section 6

Submitted by

Syeda Ramima Rafsana Nadia 2012863042

Ahanaf Tahomid 1831920642

Submitted to

Mohammad Shifat-E-Rabbi

Project Overview

The objective of this project is to generate realistic prototype images of notable figures across diverse fields, such as artists, movie actors, athletes, writers, and scientists, using generative modeling techniques. By training a Generative Adversarial Network (GAN) on a dataset of images from these categories, we aim to produce synthetic images that capture distinct features associated with each professional group. GANs offer a powerful way to synthesize realistic images by learning patterns from existing data, making them a suitable choice for this project.

Project Goals

- **Data Collection and Preparation:** Gather a dataset of notable figures from diverse professions to train our model on various categories.
- **Model Training with GANs:** Use a GAN architecture, specifically a Deep Convolutional GAN (DCGAN), to learn and generate realistic images.
- **Evaluation and Fine-Tuning:** Assess the quality of generated images and iteratively improve model performance.

Challenges and Problem-Solving Approaches

1. **Dataset Challenges**
    - **Problem**: Uneven representation across categories led to imbalanced learning during model training.
    - **Solution**: Additional images were sourced for underrepresented categories. Augmentation techniques like rotations, flips, and brightness adjustments were applied to improve dataset balance.
2. **Model Convergence Issues**
    - **Problem**: Initial GAN training resulted in poor-quality images, with the generator failing to produce realistic outputs.
    - **Solution**: Learning rates were fine-tuned, and the loss functions for both the generator and discriminator were stabilized. A batch normalization layer was added to improve gradient flow.
3. **Overfitting in Discriminator**
    - **Problem**: The discriminator overpowered the generator, leading to mode collapse where the generator produced repetitive images.
    - **Solution**: Techniques such as label smoothing and reducing the discriminator's learning rate were implemented to balance adversarial training.
4. **Computational Limitations**
    - **Problem**: Training on high-resolution images required significant computational power, causing delays.
    - **Solution**: The resolution was reduced to 32x32 pixels for initial training, and later experiments were planned for higher resolutions.

Detailed Workflow

1. Dataset Preparation

- Images were categorized into folders based on professions.
- Preprocessing involved resizing all images to 64x64 pixels and normalizing pixel values to the range [-1, 1] for stabilized training.

2. GAN Architecture

- **Generator**: Used transposed convolutional layers with ReLU activations and batch normalization for upsampling and stability.
- **Discriminator**: Employed convolutional layers with Leaky ReLU activations to robustly classify real and generated images.

3. Training Process

- Alternating updates for the generator and discriminator ensured balanced learning.
- Mini-batches of real and generated images were used to minimize adversarial loss, gradually improving image realism.

Mathematical Functions

1. The Generative Adversarial Network (GAN) Framework

A GAN consists of two neural networks: a **Generator (G)** and a **Discriminator (D)**, which compete with each other in a **minimax game**.

Objective Function

The GAN optimization problem is defined as:

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

- $D(x)$:: Probability that x is a real image (output of the Discriminator).
- $G(z)$: Generator's output for noise z sampled from a prior distribution $p_z(z)$(e.g., Gaussian or uniform distribution).
- $p_{data}(x)$: Distribution of real data.

The **Generator** aims to minimize $\log(1 - D(G(z)))$, while the **Discriminator** tries to maximize $\log D(x) + \log(1 - D(G(z)))$.

2. Generator Network

The Generator maps a random noise vector z to an image $G(z)$. The mapping is learned via a neural network that uses **transposed convolutions** to upsample the noise.

Input to Generator:

- $z \sim p_z(z)$: A random noise vector from a prior distribution (e.g., Gaussian).

Generator's Loss Function:

$$L_G = -E_{z \sim p_z(z)}[\log D(G(z))]$$

The loss measures the Generator's success in "fooling" the Discriminator.

Mathematical Steps:

1. Start with a random noise vector $z \in R^n$
2. Apply a series of transposed convolutional layers:
   $G(z) = \sigma(\text{ConvTranspose}(z))$
3. where σ is an activation function (e.g., ReLU for hidden layers, Tanh for output).
4. Output is an image $G(z)$ with pixel values scaled to $[-1, 1]$.

3. Discriminator Network

The Discriminator is a binary classifier that predicts whether an image is real or generated. It uses **convolutional layers** to extract features from the input.

Discriminator's Loss Function:

$$L_D = -E_{x \sim p_{data}(x)}[\log D(x)] - E_{z \sim p_z(z)}[\log(1 - D(G(z)))]$$

This measures the Discriminator's ability to distinguish real images from generated ones.

Mathematical Steps:

1. Input an image $x \in R^{h \times w \times c}$(real or generated).
2. Apply a series of convolutional layers:
   $D(x) = \sigma(\text{Conv}(x))$
   where σ is a sigmoid activation function for the final output.
3. Output is a scalar $D(x) \in [0, 1]$, representing the probability that the image is real.

4. Combined Loss for GAN Training

During training, G and D are updated alternately:

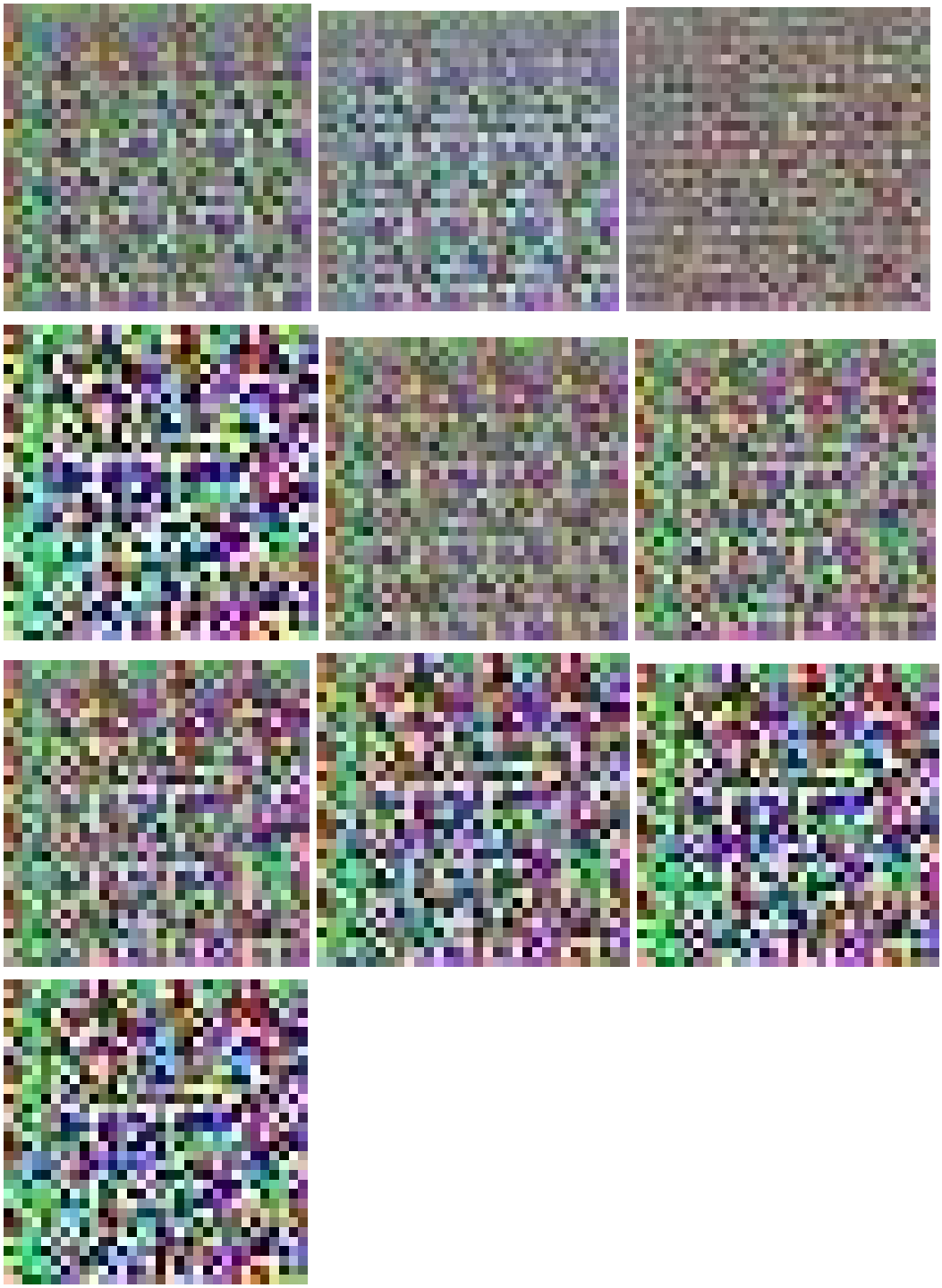
1. **Update D:** Minimize L_D by updating Discriminator weights:
   $\nabla_\theta D L_D = -E_{x \sim p_{data}(x)}[\nabla_\theta D \log D(x)] - E_{z \sim p_z(z)}[\nabla_\theta D \log(1 - D(G(z)))]$
2. **Update G:** Minimize L_G by updating Generator weights:
   $\nabla_\theta G L_G = -E_{z \sim p_z(z)}[\nabla_\theta G \log D(G(z))]$

Future Directions

1. **Enhanced Model Training**
    - Experiment with advanced GAN architectures, such as **StyleGAN** or **BigGAN**, to achieve higher-quality and high-resolution outputs.
    - Incorporate progressive growing techniques to train on increasing resolutions.
2. **Dataset Expansion**
    - Further diversify datasets by including images from less commonly represented professions or regions to improve model generalization.
3. **Evaluation Metrics**
    - Use **perceptual path length** to analyze latent space interpolation and enhance interpretability of generated outputs.
4. **Practical Applications**
    - Extend the model to create composite images for creative industries, enabling applications in entertainment, advertising, and education.

Dataset Preview

Generated Images



Conclusion

This project showcased the power of GANs in synthesizing realistic images, reflecting the characteristics of notable figures across diverse professions. Despite challenges like mode collapse and dataset imbalances, the DCGAN demonstrated robust learning capabilities, producing outputs that closely mirrored real-world patterns.

By refining our model, expanding datasets, and incorporating advanced GAN techniques, we aim to push the boundaries of generative modeling, unlocking new possibilities for creative and practical applications in image generation.