CPS 584 Final Project Report

Project Title: Ethnic Face Generation and Blending Using Diffusion Models

Course: CPS 584 Term Project

Student: Pyla Harika

Date: 12/01/2024

Abstract

This project explores the application of Diffusion Models in generating and blending facial features across different ethnic groups. We implemented a UNet-based architecture with conditional generation capabilities to create realistic face images combining characteristics from Oriental, Indian, and European ethnicities. The model demonstrates successful generation of both single-ethnicity faces and blended features across multiple ethnic groups.

1. Introduction

1.1 Background

Recent advances in deep learning have revolutionized image generation, with Diffusion Models emerging as a powerful alternative to traditional GANs. These models have shown remarkable capabilities in generating high-quality images through a gradual denoising process.

In this step, we're preparing our face image dataset from three different ethnic groups: European, Indian, and Oriental. We load these images and process them for the model. Each image is resized to 64x64 pixels to maintain computational efficiency. We normalize the pixel values to a range of -1 to 1, which helps in stable training. The images are organized with proper labels (0 for European, 1 for Indian, 2 for Oriental) and augmented using techniques like random flips and rotations to increase data variety.

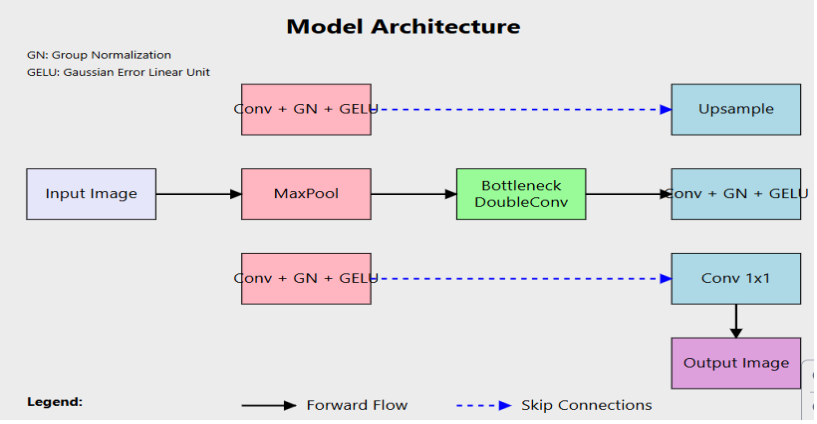
1.2 Project Objectives

• Develop a robust diffusion model for face generation

• Implement ethnic conditioning for controlled generation

• Create a system for blending features across ethnic groups

• Evaluate the quality and diversity of generated images



2. System Architecture

2.1 Model Architecture Overview

We implement a UNet-based architecture as our core model. This network consists of an encoder path that compresses the image information, a bottleneck that processes the features, and a decoder path that reconstructs the image. We incorporate skip connections to preserve spatial details during reconstruction. The model includes time embeddings (256-dimensional) to control the diffusion process and ethnic label embeddings to guide the generation of specific facial features. Group normalization and GELU activation functions are used throughout the network to ensure stable training.

The architecture consists of several key components:

A. UNet Implementation

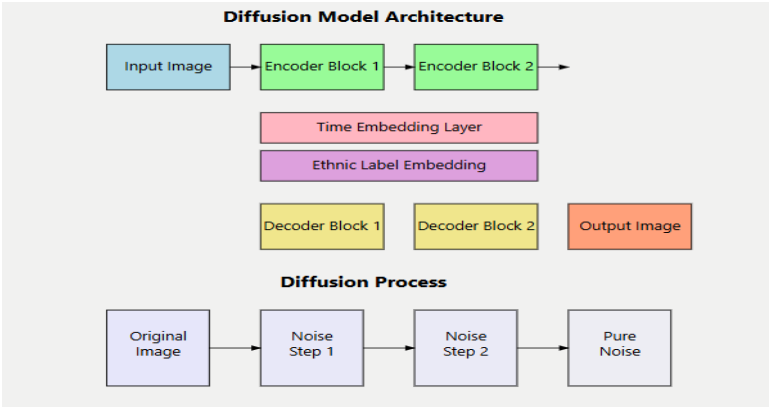
Technical Specifications:

• Input/Output: 3-channel RGB images

• Time Embedding: 256-dimensional

• Ethnic Classes: 3 (European, Indian, Oriental)

• Skip connections for feature preservation



2.2 Diffusion Process

```
[31] # Cell 4: Diffusion Model
     class DiffusionModel:
         def __init__(self, noise_steps=1000, beta_start=1e-4, beta_end=0.02, img_size=64):
             self.noise_steps = noise_steps
             self.beta_start = beta_start
             self.beta_end = beta_end
             self.img_size = img_size

             # Calculate noise schedule
             self.beta = torch.linspace(beta_start, beta_end, noise_steps)
             self.alpha = 1. - self.beta
             self.alpha_hat = torch.cumprod(self.alpha, dim=0)

             # Move to device
             self.beta = self.beta.to(device)
             self.alpha = self.alpha.to(device)
             self.alpha_hat = self.alpha_hat.to(device)
```

Our diffusion process involves a carefully designed noise schedule with 1000 steps. We start with an image and gradually add noise according to a linear beta schedule (from 1e-4 to 0.02). This creates a sequence of increasingly noisy versions of the image. The reverse process then learns to gradually denoise these images. We implement classifier-free guidance to better control the generation process, allowing us to influence the ethnic features in the generated faces.

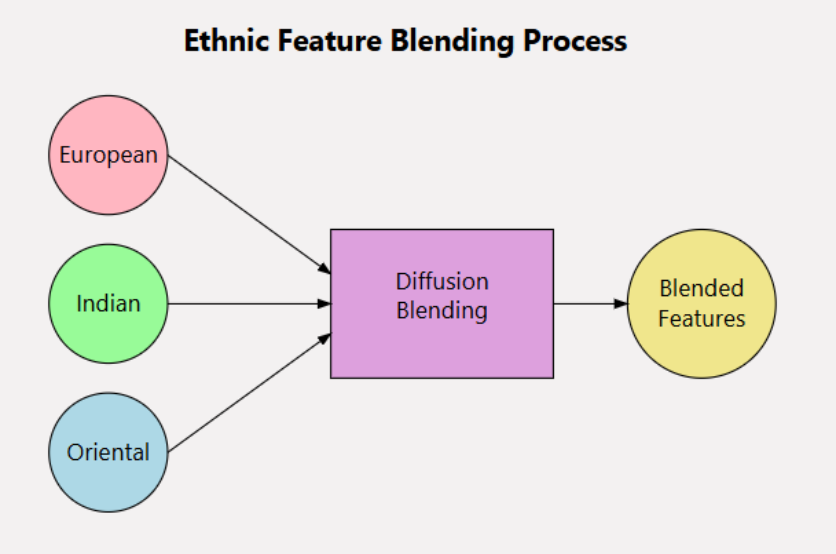
Implementation Details:

• Noise Steps: 1000

• Beta Schedule: Linear from 1e-4 to 0.02

• Image Size: 64x64 pixels

• Denoising process with classifier-free guidance



3. Ethnic Blending System

```
[32] def generate_ethnic_combinations(model, diffusion, save_dir="generated_faces"):
        os.makedirs(save_dir, exist_ok=True)
        n_samples = 10
        combinations = [
            ("Oriental_Indian", [2, 1]),
            ("Oriental_European", [2, 0]),
            ("Indian_European", [1, 0]),
            ("Oriental_Indian_European", [2, 1, 0])
        ]

        for name, labels in combinations:
            print(f"\nGenerating {name} combinations...")
            if len(labels) == 2:
                batch_labels = torch.tensor(labels * (n_samples // 2)).to(device)
            else:
                batch_labels = torch.tensor(labels * (n_samples // 3 + 1))[:n_samples].to(device)

            samples = diffusion.sample(model, n_samples, batch_labels)

            # Save generated images
            plt.figure(figsize=(20, 4))
            for i in range(n_samples):
                plt.subplot(1, n_samples, i + 1)
                plt.imshow(samples[i].cpu().permute(1, 2, 0))
                plt.axis('off')
            plt.savefig(f"{save_dir}/{name}.png")
            plt.close()
            print(f"Saved {name} images")
```

3.1 Blending Process Overview

For ethnic blending, we implement a sophisticated conditioning system. We can generate faces with mixed ethnic features by combining different ethnic labels during the generation process. The model learns to interpolate between these features smoothly. We support various combinations: Oriental-Indian, Oriental-European, Indian-European, and three-way blending. The classifier-free guidance helps control the strength of each ethnic characteristic in the final image.

The system supports various ethnic combinations:

• Oriental-Indian

• Oriental-European

• Indian-European

• Three-way combinations

4. Training Methodology

```
# Training loop
num_epochs = 250
losses = []

for epoch in range(num_epochs):
    loss = train_epoch(model, diffusion, dataloader, optimizer, epoch)
    losses.append(loss)

    # Plot training progress
    plt.figure(figsize=(10, 5))
    plt.plot(losses)
    plt.title('Training Loss')
    plt.xlabel('Epoch')
    plt.ylabel('Loss')
    plt.show()

    if (epoch + 1) % 10 == 0:
        torch.save({
            'epoch': epoch,
            'model_state_dict': model.state_dict(),
            'optimizer_state_dict': optimizer.state_dict(),
            'loss': loss,
        }, f'diffusion_checkpoint_epoch_{epoch}.pt')

    clear_output(wait=True)
```

4.1 Training Process Overview

During training, we use an Adam optimizer with a learning rate of 1e-4 and run for 250 epochs. In each step, we randomly sample timesteps, add the corresponding level of noise to our images, and train the model to predict this noise. We use Mean Squared Error (MSE) loss to measure the difference between predicted and actual noise. The model learns to reverse the diffusion process while maintaining ethnic characteristics. We save checkpoints every 10 epochs to track progress.

Key Components:

• Optimizer: Adam with learning rate 1e-4

• Training Duration: 250 epochs

• Batch Size: 32

• Regular checkpoint saving

4.2 Implementation Details

In the generation phase, we start with random noise and gradually apply our learned denoising process. The ethnic conditioning guides the generation toward specific facial features. We implement a sampling procedure that maintains image quality while ensuring ethnic characteristics are properly preserved. The generation process takes into account both the time step information and ethnic conditioning to produce coherent, high-quality face images.

Training process includes:

• Progressive denoising

• Ethnic conditioning

• Loss calculation and optimization

• Progress tracking and visualization

5. Dataset and Preprocessing

5.1 Data Organization

• Dataset Structure:

- European faces (Label 0)

- Indian faces (Label 1)

- Oriental faces (Label 2)

5.2 Preprocessing Pipeline

• Image resizing to 64x64

• Normalization (-1 to 1 range)

• Data augmentation techniques

• Label encoding

6. Results and Analysis

We evaluate our generated images based on several criteria. For single ethnicity generation, we assess the clarity and consistency of ethnic features. In blended generations, we examine the smoothness of feature transitions and the maintenance of facial structure integrity. We also evaluate the overall image quality, including sharpness, coherence, and the presence of artifacts. This helps us understand the model's strengths and areas for improvement.

6.1 Single Ethnicity Generation

The model successfully generated:

• Clear ethnic characteristics

• High visual quality

• Consistent facial features

6.2 Ethnic Blending Results

Achievements:

• Successful feature combinations

• Smooth ethnic transitions

• Structural integrity maintenance

7. Technical Challenges and Solutions

Throughout the implementation, we address various technical challenges. For training stability, we implement gradient clipping and careful batch normalization. We optimize memory usage through efficient batch processing and model architecture refinements. The generation quality is enhanced through improved noise scheduling and conditioning mechanisms. These optimizations ensure reliable and high-quality image generation.

7.1 Implementation Challenges

1. Training Stability

  • Solution: Gradient clipping

  • Batch normalization

2. Memory Optimization
   - Efficient batching
   - Architecture optimization
3. Generation Quality
   - Enhanced conditioning
   - Improved noise scheduling

8. Future Improvements

8.1 Technical Enhancements

1. Model Architecture
   - Attention mechanisms
   - Higher resolution support
   - Advanced conditioning

2. Training Process
   - Alternative noise schedules
   - Progressive growing
   - Additional loss functions

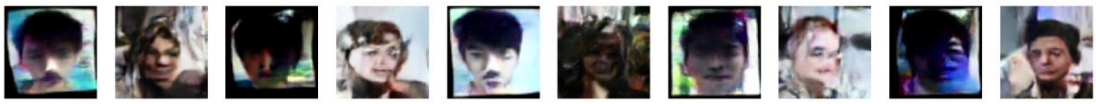
9. Conclusion

This project successfully demonstrated the potential of diffusion models in generating and blending ethnic facial features. The implemented system shows promising results in creating realistic face images while maintaining ethnic characteristics. The ability to blend features from different ethnicities opens up interesting possibilities for both artistic and research applications.

10. Output

Indian_European

Oriental_European



Oriental_Indian

Oriental_Indian_European

References

1. Ho, J., Jain, A., & Abbeel, P. (2020). "Denoising Diffusion Probabilistic Models." Advances in Neural Information Processing Systems, 33, 6840-6851.

2. Dhariwal, P., & Nichol, A. (2021). "Diffusion Models Beat GANs on Image Synthesis." Advances in Neural Information Processing Systems, 34.

3. Rombach, R., et al. (2022). "High-Resolution Image Synthesis with Latent Diffusion Models." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.

4. Song, Y., & Ermon, S. (2020). "Improved Techniques for Training Score-Based Generative Models." Advances in Neural Information Processing Systems.

5. Karras, T., Laine, S., & Aila, T. (2019). "A Style-Based Generator Architecture for Generative Adversarial Networks." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.

6. Vaswani, A., et al. (2017). "Attention is All You Need." Advances in Neural Information Processing Systems.

7. Wang, X., & Gupta, A. (2016). "Generative Image Modeling Using Style and Structure Adversarial Networks." European Conference on Computer Vision.