Face generation

Kaggle Dataset Link: https://www.kaggle.com/jessicali9530/celeba-dataset

Problem Description:

Face generation is a field of computer vision and machine learning that deals with the generation of artificial or synthetic images of human faces. This involves training deep learning models on large datasets of human faces and then using these models to generate new faces that look realistic and plausible.

The CelebA dataset is a widely-used dataset in the field of face generation. It contains over 200,000 images of human faces, each annotated with attributes such as gender, age, and hair color. The dataset was created by collecting images of celebrities from the internet, and as such, it contains a diverse range of people from different races, ethnicities, and cultures.

The goal of this project is to train a deep learning model on the CelebA dataset and use it to generate new synthetic faces that are realistic and visually appealing. This will involve several steps, including preprocessing the data, training a generative adversarial network (GAN), and optimizing the model to generate high-quality images. The project will also involve evaluating the performance of the model using metrics such as inception score and FID score.

Overall, the project aims to demonstrate the power of deep learning for image generation and showcase the potential applications of this technology in fields such as entertainment, fashion, and art.

Key Objectives:

- Preprocess the CelebA dataset
- Train a Generative Adversarial Network (GAN) on the dataset
- Optimize the model to generate high-quality images

- Evaluate the performance of the model using metrics such as inception score and FID score
- Generate new synthetic faces using the trained model

Dataset Description: The CelebA dataset is a large-scale face attributes dataset with more than 200,000 celebrity images, each with 40 attribute annotations. The images cover large pose variations and background clutter. CelebA has large diversities, large quantities, and rich annotations, including:

- 10,177 number of identities
- 202,599 number of face images
- 5 landmark locations, 40 binary attributes annotations per image

The dataset is available in a zip file that contains a folder of images along with attribute information in a CSV file.

(Source: https://www.kaggle.com/jessicali9530/celeba-dataset)

Possible Framework:

Framework for a face generation project:

1. **Data Gathering and Preprocessing:**
- Gather a large dataset of face images (e.g. CelebA, FFHQ, etc.) and preprocess the data by resizing, normalizing, and augmenting it as necessary.
- Split the dataset into training, validation, and testing sets.
2. **Model Selection and Architecture:**
- Choose a suitable deep learning model for face generation (e.g. Generative Adversarial Networks (GANs), Variational Autoencoders (VAEs), etc.)
- Define the architecture of the model, including the number of layers, activation functions, and other hyperparameters.
3. **Model Training:**
- Train the model using the training dataset.
- Monitor the loss and other evaluation metrics on the validation dataset to ensure the model is not overfitting.
- Tune the hyperparameters as necessary.
4. **Fine-Tuning:**
- Fine-tune the model on a smaller dataset of faces from a specific domain (e.g. age, gender, ethnicity, etc.)
- Train the model for a shorter amount of time to prevent overfitting.
5. **Deployment:**
- Deploy the trained model to generate new faces.
- Set up a user interface to allow users to interact with the model and generate new faces.
- Monitor user feedback and make improvements to the model as necessary.
6. **Evaluation:**
- Evaluate the performance of the model using metrics such as FID, Inception Score, and visual inspection of generated faces.
- Compare the performance of the model to other state-of-the-art face generation models.
- Use the evaluation results to guide further improvements to the model.
7. **Future Work:**
- Explore the use of alternative models or architectures for face generation.
- Investigate the use of additional data sources or preprocessing techniques to improve the quality of generated faces.
- Extend the model to generate other types of images or visual media, such as landscapes or videos.

Code Explanation :

Here is the simple explanation for the code which is provided in the code.py file.

Problem Description:

The task of this project is to generate realistic images of human faces using Generative Adversarial Networks (GANs). The dataset used for this project is the CelebA dataset, which contains over 200,000 images of celebrity faces. The goal is to train a GAN to learn the distribution of human faces in the dataset and generate new, previously unseen, realistic images of faces.

Framework:

1. Data Collection
2. Data Preprocessing
3. Model Architecture
4. Model Training
5. Generate New Images

Code Explanation:

1. Data Collection: We use the CelebA dataset which contains over 200,000 images of celebrity faces. This dataset can be downloaded from Kaggle.
2. Data Preprocessing: We preprocess the images by resizing them to a uniform size, normalizing pixel values, and converting them to tensors. We also split the dataset into training and validation sets.
3. Model Architecture: We use a deep convolutional GAN (DCGAN) architecture, which is a type of GAN that uses convolutional neural networks (CNNs) for both the generator and discriminator networks. The generator network takes a noise vector as input and generates an image, while the discriminator network takes an image as input and predicts whether it is a real or generated image.
4. Model Training: We train the GAN on the preprocessed dataset by alternating between training the generator and discriminator networks. During each training iteration, we feed real images to the discriminator and generated images to the discriminator, and backpropagate the gradients to update the weights of the networks.
5. Generate New Images: After training the GAN, we can use the generator network to generate new, previously unseen, realistic images of faces by feeding noise vectors as input to the generator and obtaining the corresponding generated images as output.

Requirements to run the code:

- Python 3.6 or higher
- PyTorch
- TorchVision
- NumPy
- Matplotlib

Future Work :

Exercise :

Try to answers the following questions by yourself to check your understanding for this project. If stuck, detailed answers for the questions are also provided.

Concept Explanation :