Islamic University Of Technology

Project Report

Synthetic Brain MRI Image Generation Using Generative Adversarial Network (GAN)

EEE 4872

—

Tanveer Mahmood Mahin

190021108

Muniza Alam

190021111

Asma Rahman

190021128

Synthetic Brain MRI Image Generation Using Generative Adversarial Network (GAN)

1. Introduction

Generative Adversarial Networks (GANs) are deep learning models that learn to generate new data samples that are similar to a given dataset. In this project, we aim to generate synthetic brain MRI images using GANs. Brain MRI images are crucial for medical diagnosis and research, but collecting a large dataset of labeled brain MRI scans can be challenging. GANs offer a solution by learning to generate synthetic MRI images that closely resemble real images, which can augment existing datasets and facilitate medical image analysis.

2. Methodology

2.1 Dataset

We used the Alzheimer's MRI dataset, which contains brain MRI scans of patients with different stages of Alzheimer's disease.This can be found in kaggle,
link : https://www.kaggle.com/datasets/sachinkumar413/alzheimer-mri-dataset

The dataset consists of four classes: 'Mild Demented', 'Moderate Demented', 'Non Demented', and 'Very Mild Demented'. Each class contains MRI images in grayscale format. But in our project we don’t need to care about the classes as we will be trying to just generate synthetic images using the whole dataset for training.

2.2 Libraries

We utilized several Python libraries for this project:

- PyTorch: PyTorch is a deep learning framework that provides flexible and efficient tools for building and training neural networks.
- torchvision: torchvision is a PyTorch library that provides common image transformations and datasets.
- Pillow: Pillow is a Python Imaging Library (PIL) fork that adds image processing capabilities to Python.

- NumPy: NumPy is a library for numerical computing in Python, which is widely used for array manipulation and mathematical operations.

2.3 Model Architecture

We implemented a GAN architecture consisting of a generator and a discriminator:

- Generator: The generator takes random noise as input and generates synthetic brain MRI images. It consists of fully connected layers followed by batch normalization and leaky ReLU activation functions.
- Discriminator: The discriminator distinguishes between real and synthetic MRI images. It consists of fully connected layers with leaky ReLU activation functions and a sigmoid output layer.

2.4 Training

We trained the GAN model using the following steps:

1. Data Loading: We loaded the brain MRI images from the dataset, resized them to a uniform size, and applied grayscale normalization.
2. Model Initialization: We initialized the generator and discriminator models and defined the loss function and optimizers.
3. Training Loop: We iteratively trained the generator and discriminator networks in alternating steps.
4. Loss Calculation: We calculated the adversarial loss for both the generator and discriminator networks.
5. Gradient Descent: We performed backpropagation and gradient descent to update the model parameters.
6. Image Generation: At the end of each epoch, we generated synthetic MRI images using the trained generator and saved them for visualization.

2.5 Visualization

We visualized the synthetic images using matplotlib library. Here we can inspect the images and determine how closely they resemble the real images.

3. Explanation of the Code

The code can be found in the github repo :
https://github.com/tanveerTFF/SyntheticMRI_ImageGeneration_WithGAN.git

The code follows the methodology described above:

1. Setup:
    - The necessary libraries are imported.
    - The Kaggle API key is uploaded and configured for accessing datasets.
    - The Alzheimer MRI dataset is downloaded and extracted.
2. Model Definition:
    - The Generator and Discriminator neural networks are defined.
    - The Generator takes random noise as input and generates synthetic images.
    - The Discriminator takes images as input and predicts whether they are real or fake.
3. Dataset Preparation:
    - A custom dataset class is defined to load and preprocess images from the Alzheimer MRI dataset.
    - Images are resized, converted to grayscale, transformed into tensors, and normalized.
4. Training Setup:
    - The device (CPU or GPU) is set for training.
    - Hyperparameters such as the latent dimension, image shape, learning rate, batch size, and number of epochs are defined.
    - Generator, Discriminator, loss function (binary cross-entropy), and optimizers (Adam) are initialized.
5. Training Loop:
    - For each epoch:
        - Iterate over batches of real images from the dataset.
        - Train the Discriminator:
            - Compute loss separately for real and fake images.

        - Update Discriminator's parameters based on the computed loss.
    - Train the Generator:
        - Generate fake images from random noise.
        - Compute Generator's loss based on Discriminator's predictions.
        - Update Generator's parameters based on the computed loss.
    - Print progress (losses) at regular intervals.
    - Save generated images at the end of each epoch for visualization.

6. Visualization:

- Load and display a subset of generated images saved during training using matplotlib.

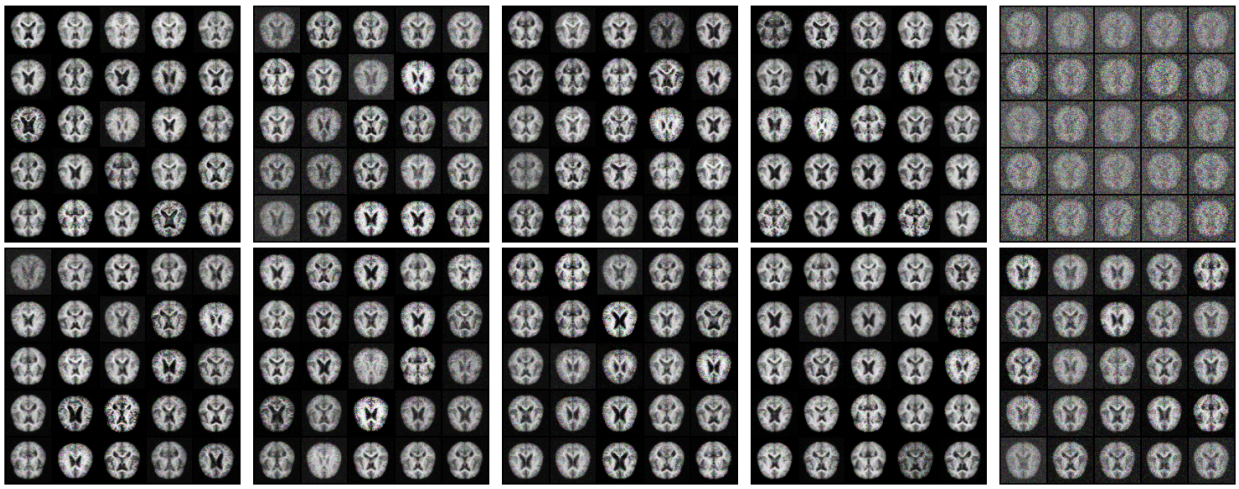

Figure : Generated synthetic MRI images

We generated images every epoch and compiled the images every 5 epoch into one. That gives us an idea how as the training went on gradually the images became more and more realistic.

4. Conclusion

In conclusion, this project demonstrates the use of Generative Adversarial Networks for generating synthetic brain MRI images. By training the GAN model on the Alzheimer's MRI dataset, we were able to generate realistic-looking MRI images that closely resemble real scans. GANs offer a promising approach for data augmentation and generation in medical imaging, which can aid in medical diagnosis and research.

5. Contributions

Each team member contributed fairly equally. Group discussions , theory overview and tutorial binging were shared among the three. So everyone deserves equal credit in this project.