**Abstract:**

This project aims to implement a Generative Adversarial Network (GAN) using Pytorch and Python for generating realistic images. Generative Adversarial Networks consist of two neural networks, a generator, and a discriminator, trained simultaneously in a competitive manner. The generator aims to create images that are indistinguishable from real ones, while the discriminator's task is to differentiate between real and generated images. The project explores the architecture and training procedures of GANs, focusing on generating high-quality images in a specific domain. Through this hands-on implementation, participants will gain insights into the capabilities of GANs and their applications in generating synthetic data, image-to-image translation, and creative content generation.

**Workflow**:

Setting up your environment

|

Building a data pipeline

|

Creating a generator & discriminator

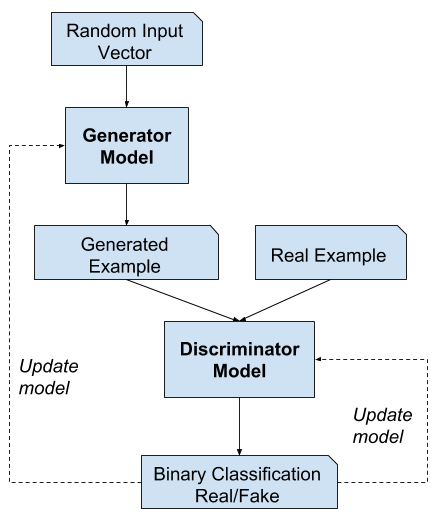
|

Building a custom training loop

|

Generating new image

Workflow for generator and discriminator



Methodology

1) Input Vector Specification: Week 1(13 January)

Initiate the process by creating random input vectors, also known as latent vectors, which act as seeds for the generator. These vectors are drawn from a specified distribution and are crucial in determining the uniqueness of the generated images.

2) Generator model architecture: Week 2(13 to 18 January)

Outline the architecture of the generator model responsible for transforming the random input vectors into meaningful images.

3) Generated Example Production: Week 2(19 January)

Employ the trained generator to transform the random input vectors into synthetic images. Illustrate the diversity and quality of generated examples, showcasing the model's ability to create realistic content.

4) Discriminator Model Architecture: Week 2(19 to 20 January)

Introduce the discriminator model designed to differentiate between real and generated images. Emphasize the integration of convolutional layers for feature extraction and binary classification capabilities.

5) Binary Classification Output: Week 3 (25 to 27 January)

Highlight the binary classification output of the discriminator, indicating whether an image is deemed real or fake. This output serves as a critical evaluation metric for the effectiveness of the generator in creating authentic-looking images.

6) Training and Optimization: Week4 (1 to 3 February)

Iterate through training loops where the generator aims to deceive the discriminator, while the discriminator strives to correctly classify both real and generated examples. The optimization process refines model weights to enhance performance.

7) Evaluation: Week5(8 to 10 February)

Assess the overall performance of the GAN by inspecting the generated images and analysing discriminator accuracy. Utilize evaluation results to fine-tune model hyperparameters and make architectural adjustments.

8) Documentation and Presentation: Week5

Compile comprehensive documentation illustrating each step, and present findings to stakeholders, showcasing the GAN's ability to generate realistic images and the discriminator's effectiveness in binary classification.