Reflection Report – Image Generation with BigGAN

1. Introduction

GANs (Generative Adversarial Networks) are AI models that can create new data, like images. They work with two parts: a **generator** that tries to create realistic images, and a **discriminator** that tries to spot the fake ones. Over time, the generator gets better and better until it can produce images that look real.

2. What I Did

In this experiment, I used **BigGAN**, a pretrained model trained on the **ImageNet** dataset. I worked specifically with the version called `biggan-deep-128`.

To generate images, here's what I did:

- I created a **random noise vector** using a truncated normal distribution.
- I picked the class **"zebra"** using the function `one_hot_from_names()`.
- I passed the noise vector, the class vector, and a **truncation value** (0.4) into BigGAN.
- The model gave me a tensor, which I converted into a viewable image using Python libraries.

3. What I Saw

I only generated images of **zebras**, but even then, I noticed some interesting things:

- All the images looked like zebras with black and white stripes, but each one was a bit different.
- Sometimes the zebra looked clean and sharp, other times more blurry or abstract.
- Just changing the noise vector changed the image a lot – the zebra's pose, angle, and background could all shift.

4. What I Learned

This experiment helped me understand how GANs work in practice. It's cool to see how just a bit of random noise can turn into a full image.

What I liked:

- The images are realistic and the process is quick.
- You don't need a label beyond the class – the noise does the rest.

What could be improved:

- It's hard to control the exact details of the image.
- Sometimes the image quality isn't perfect, and you get strange results.

In the future, I'd like to try **StyleGAN**, which gives you more control over the details. But overall, this was a fun and powerful way to see how AI can create visual content from scratch.

Mamadou Amate FALL