Date:

BITS Pilani
Pilani Campus

TITLE : Explaining VAE vs GAN: A Comparative Approach Using XAI

Sruti Darshini Neti - 2021B1A33212H
Devesh Dhyani - 2021B1A83131H
Shaheen Ali - 2021B4A33044H
Rajeshwari Gunda-2022A8PS0581H

Introduction

- Explainable AI (XAI) refers to techniques that make the decision-making processes of AI models more transparent and understandable to humans. Traditional AI models, particularly complex ones like deep neural networks, often function as "black boxes," where it is difficult to discern how they arrive at specific predictions.

- XAI aims to provide insights into model behavior by explaining which features or inputs contribute most to a decision, helping build trust and allowing for debugging and fairness checks.

- Common XAI methods include **GradCAM** for visualizing important regions in image data, **t-SNE, PCA and UMAP** for latent space representation, and feature importance methods like **SHAP** and **LIME**, which explain the impact of individual features on the model's output. These tools are essential for improving transparency in AI applications across domains.

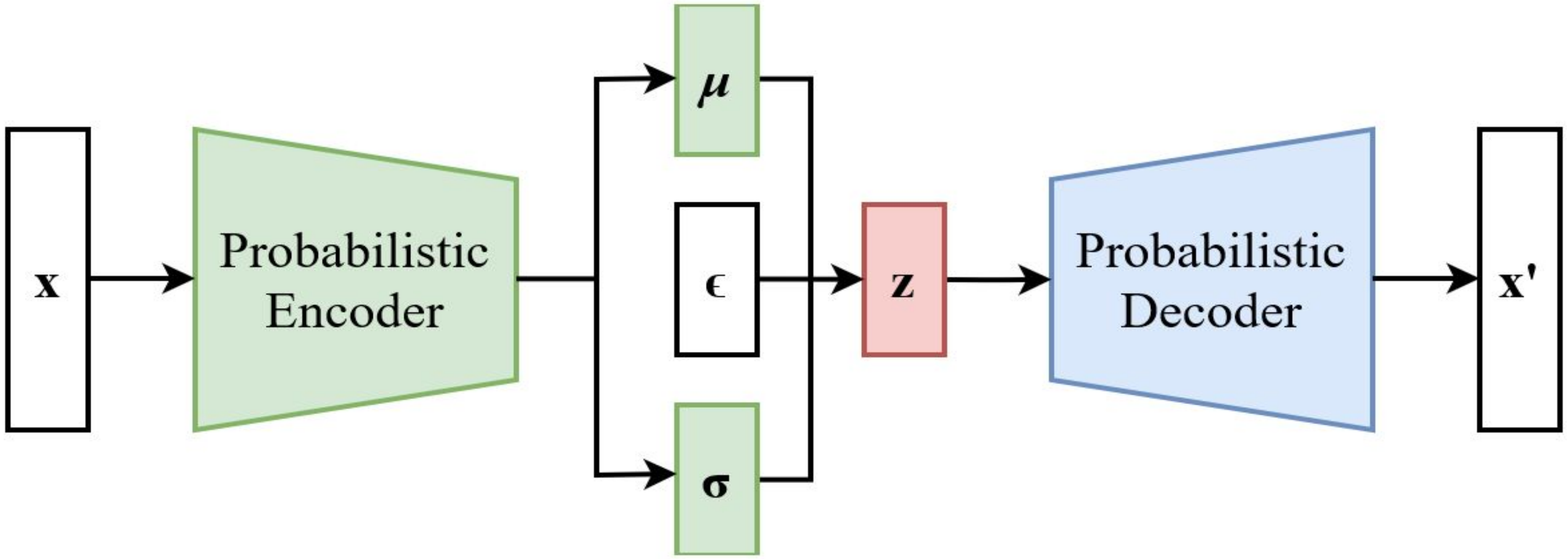
Motivation

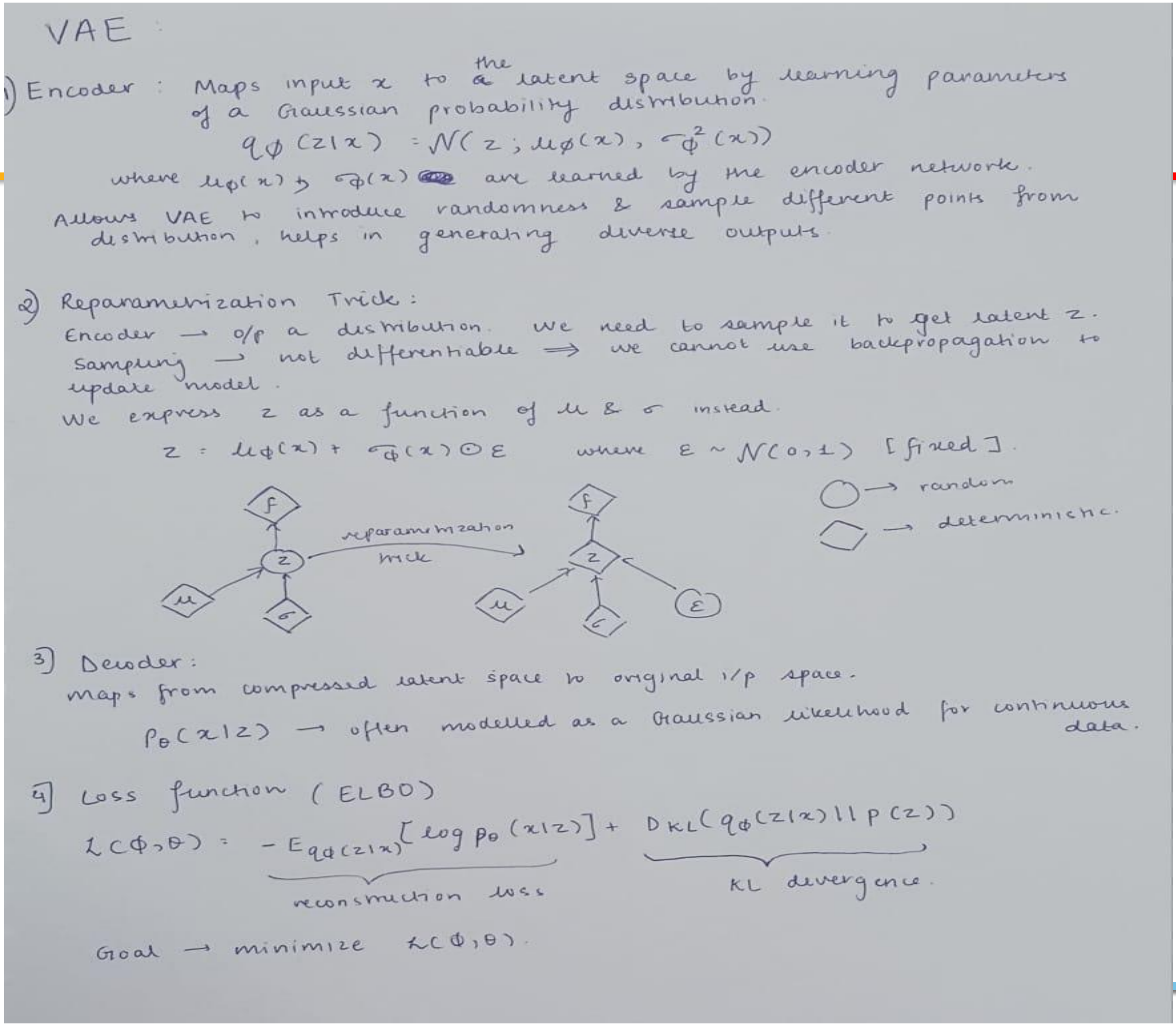
- Generative models like **CVAE** and **DCGAN** face key challenges such as **lack of interpretability**, making it difficult to understand how specific features are generated and how decisions are made.
- The structure of the **latent space** is often unclear, hindering the ability to map generated outputs to meaningful concepts. T
- The **significance of individual neurons and layers** in the generation process is not well understood, and these models struggle to **generalize** to more complex, diverse datasets, often facing issues like mode collapse (DCGAN) or poor image quality (CVAE).
- We address these gaps by using **XAI techniques** like **GradCAM**, **t-SNE**, **PCA, UMAP SHAP**, and **LIME** to interpret model behavior. This includes visualizing latent spaces, explaining how specific neurons or layers contribute to feature generation, and providing insights to improve model transparency, reliability, and control.

Objectives

- To understand the operation, architecture and formulation of VAE and GAN as generative models.
- To apply XAI techniques (PCA, UMAP, SHAP, LIME, t-SNE) to CVAE and DCGAN.
- To understand the significance of nodes and understand how different layers of the neural networks contribute to the generative process.
- To visualize and explain the latent space of the models and the output.
- Comparative analysis of the latent space and encoder-decoder outputs of both generative models side-by-side
- Define use cases for each model.

Variational Autoencoder





VAE :

1) Encoder : Maps input x to ~~a~~ the latent space by learning parameters of a Gaussian probability distribution.

$$q_\phi(z|x) = \mathcal{N}(z; \mu_\phi(x), \sigma^2_\phi(x))$$

where $\mu_\phi(x)$ & $\sigma_\phi(x)$ are learned by the encoder network.

Allows VAE to introduce randomness & sample different points from distribution, helps in generating diverse outputs.

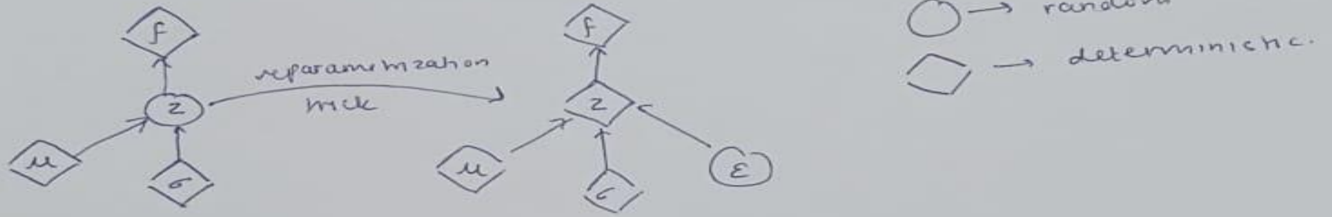
2) Reparametrization Trick :

Encoder → o/p a distribution. We need to sample it to get latent z.

Sampling → not differentiable ⇒ we cannot use backpropagation to update model.

We express z as a function of μ & σ instead.

$$z = \mu_\phi(x) + \sigma_\phi(x) \odot \varepsilon \quad \text{where } \varepsilon \sim \mathcal{N}(0,1) \text{ [fixed]}.$$



3) Decoder :

maps from compressed latent space to original i/p space.

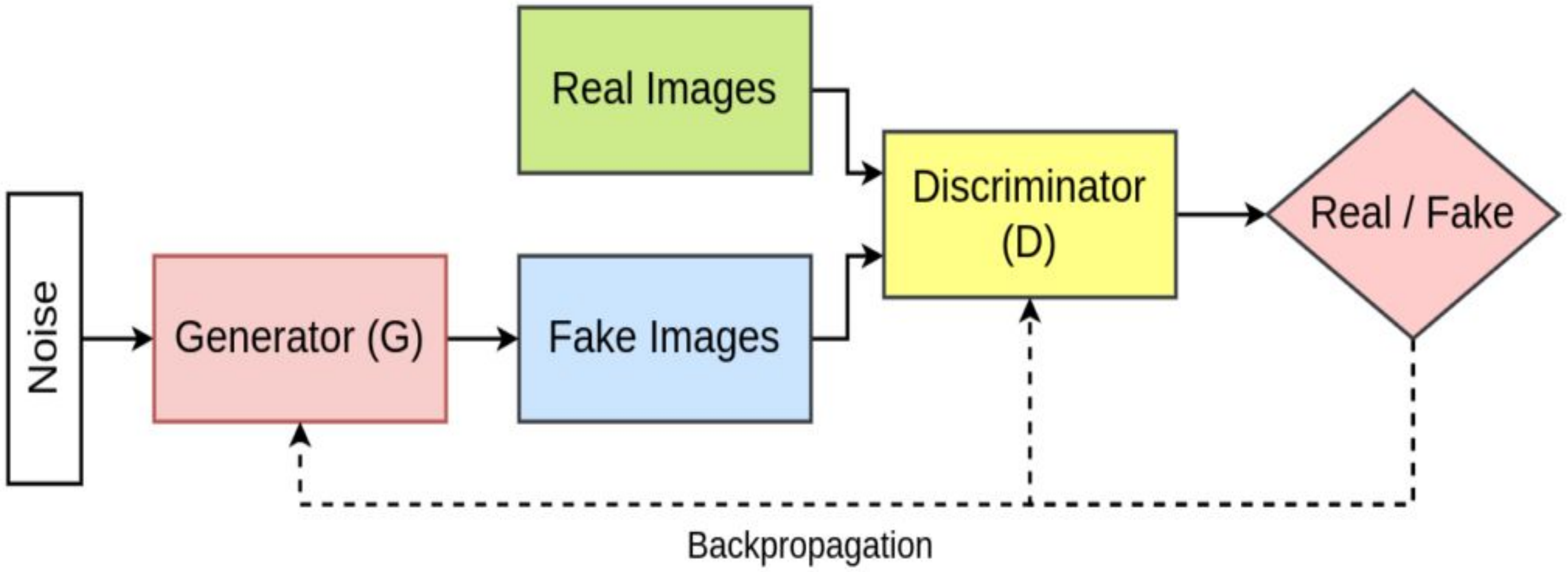
$p_\theta(x|z)$ → often modelled as a Gaussian likelihood for continuous data.

4) Loss function (ELBO)

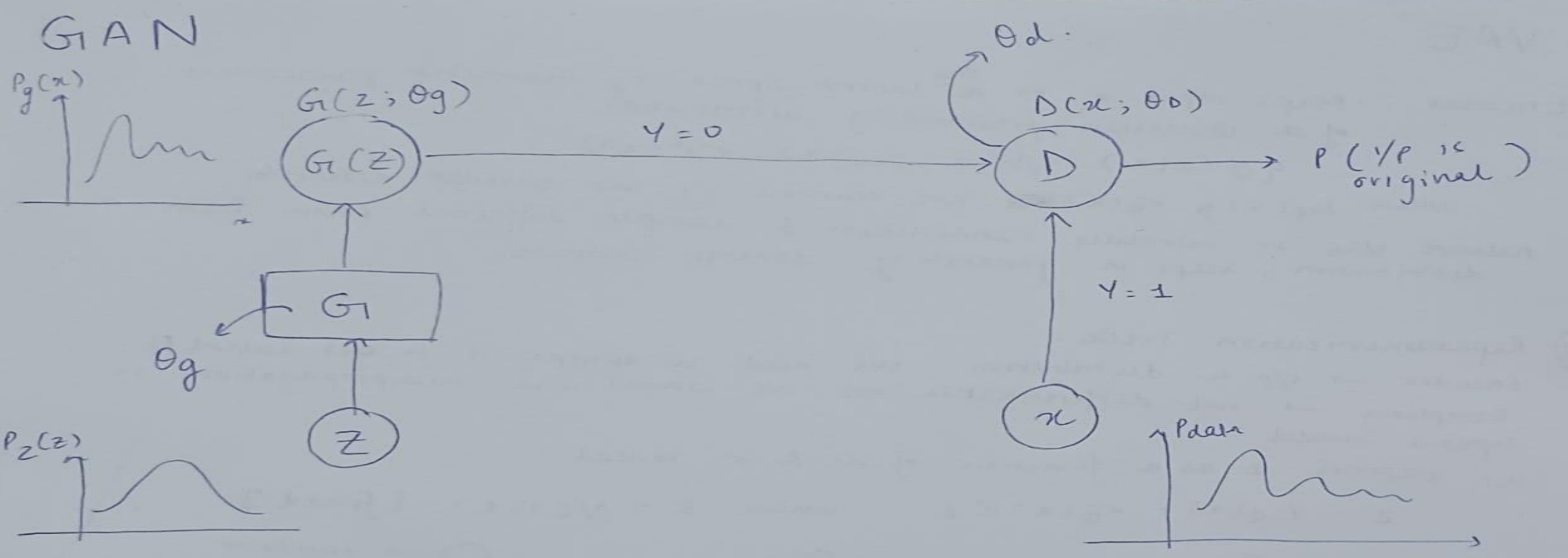
$$\mathcal{L}(\phi,\theta) = \underbrace{-\mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)]}_{\text{reconstruction loss}} + \underbrace{D_{KL}(q_\phi(z|x)\,||\,p(z))}_{\text{KL divergence}}$$

Goal → minimize $\mathcal{L}(\phi,\theta)$.

Generative Adversarial Network



GAN



Value function:

$$\min_{G}\max_{D} V(D,G) = E_{x\sim p_{data}(x)}[\underbrace{\log D(x)}_{A}] + E_{x\sim p(x)}[\underbrace{\log(1-D(G(z)))}_{B}]$$

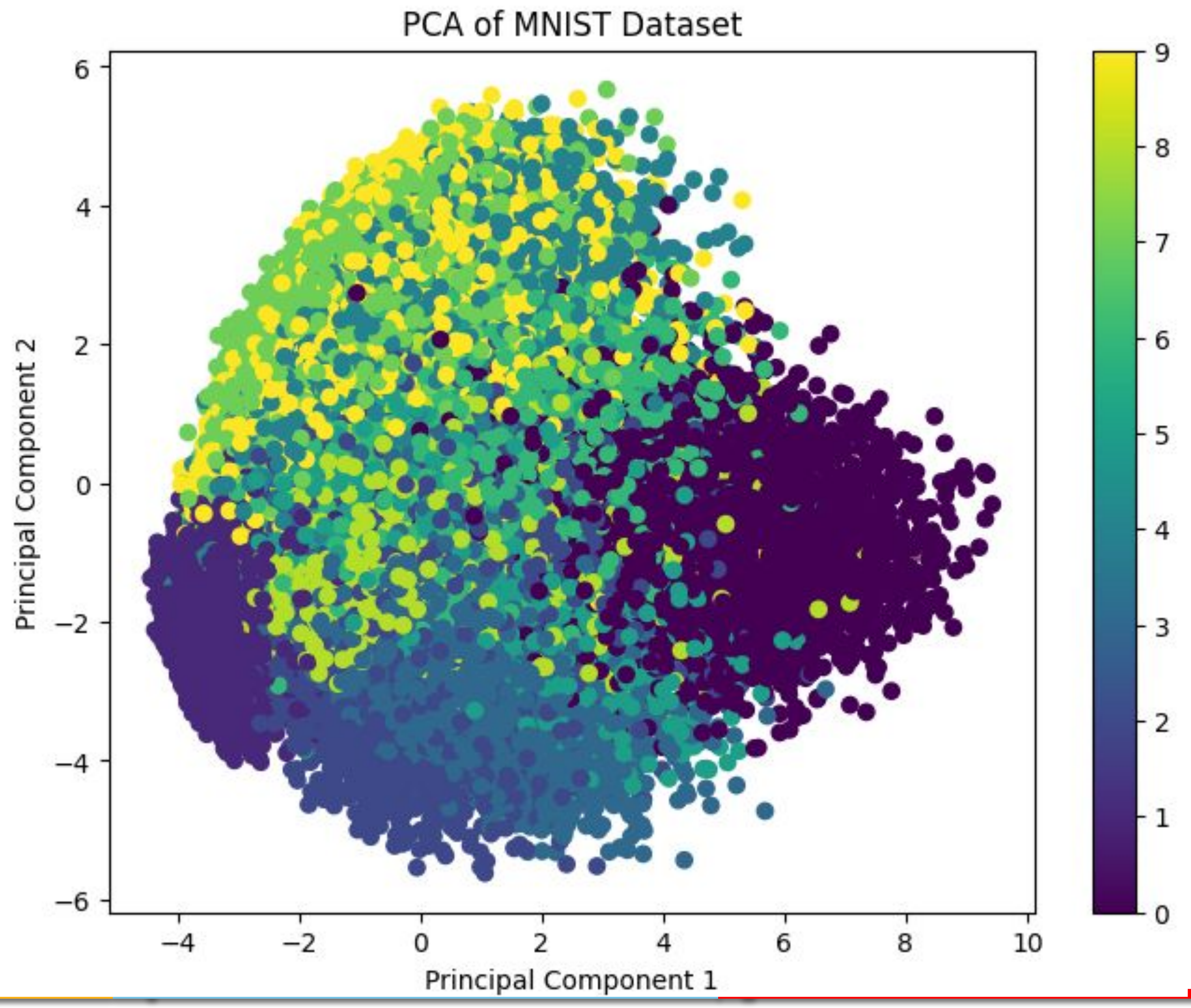
G, D participate in a 2 player min-max game.

Discriminator goal → Maximise A (for real data) & B (for fake data)

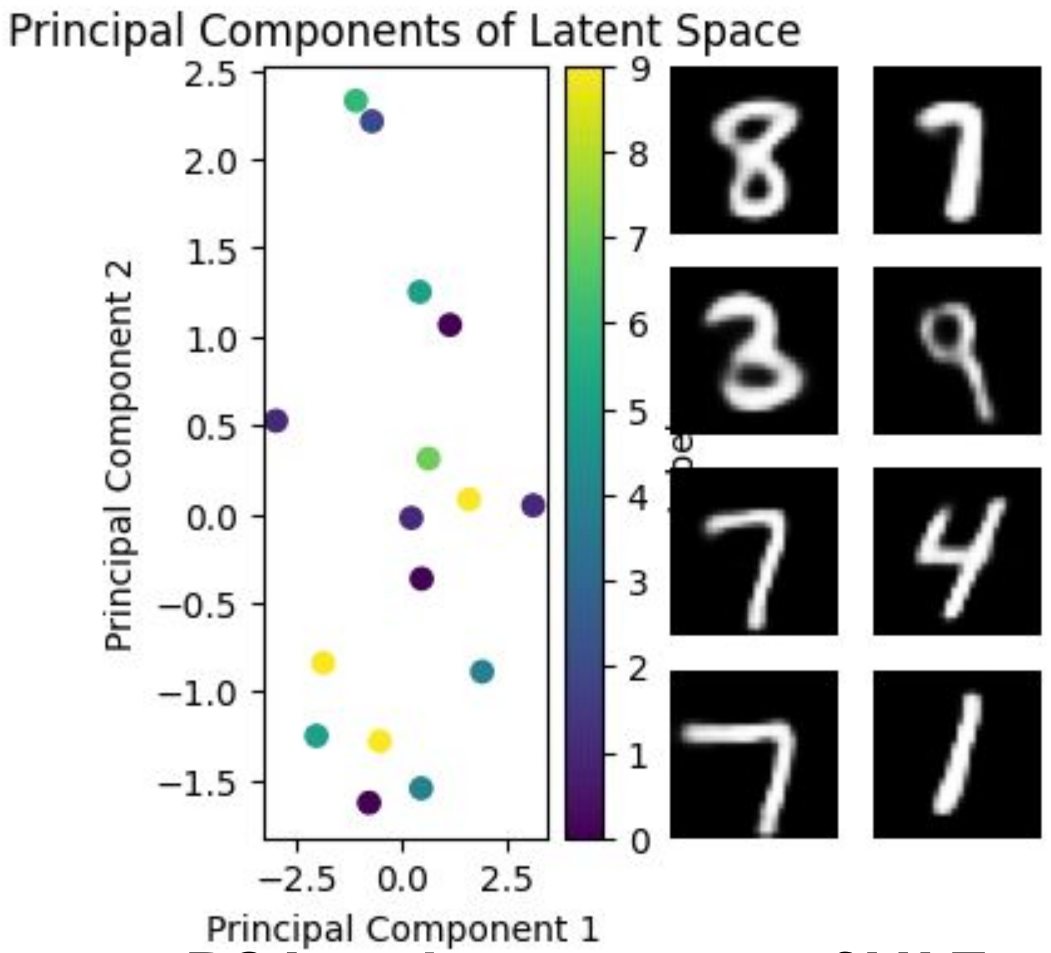
Generator goal → Minimise B ~~i.e generate~~

Principal Component Analysis (PCA)

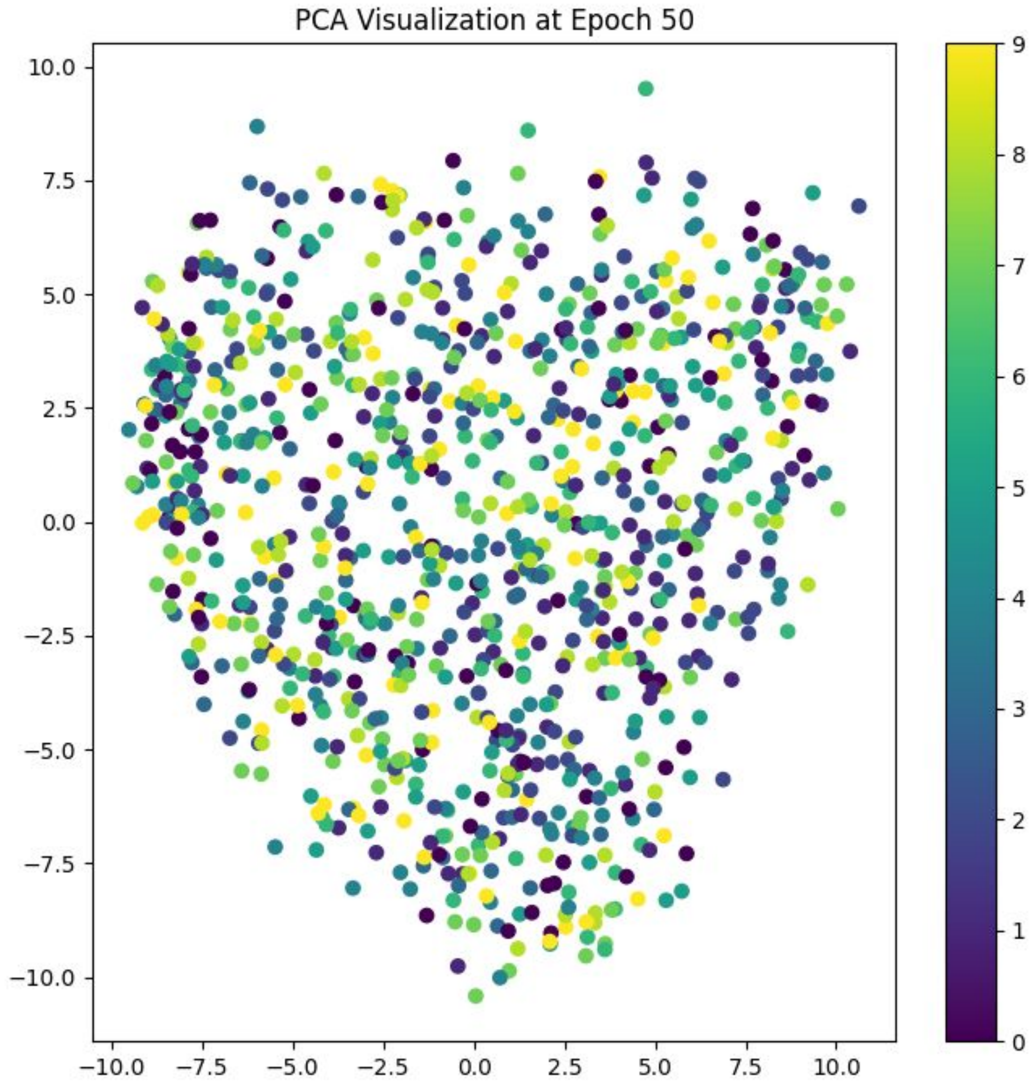
- Linear dimensionality reduction technique.
- Finds new axes (principal components) that maximize variance.
- Transforms high-dimensional data into fewer dimensions.
- Preserves global structure but misses non-linear relationships.
- Computationally efficient and widely used for visualization and noise reduction.
- Commonly applied to large datasets in areas like image and data compression.



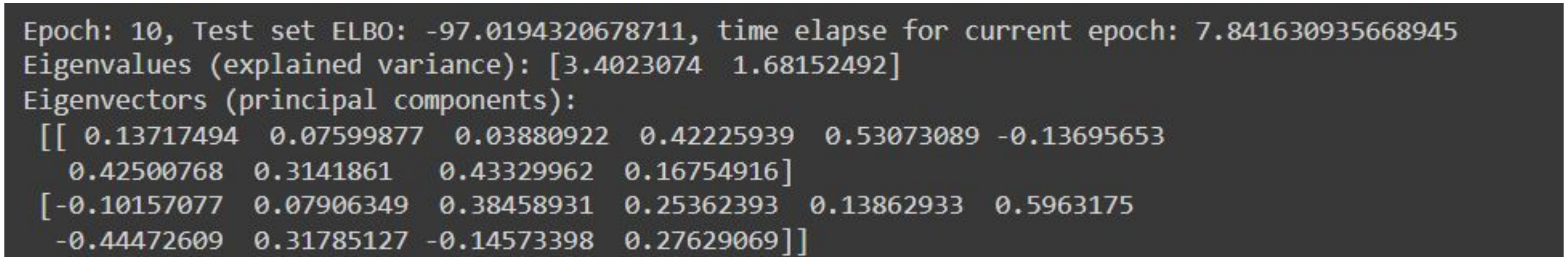
PCA on generator latent space of GAN



PCA on latent space of VAE



PCA inferences (VAE example)

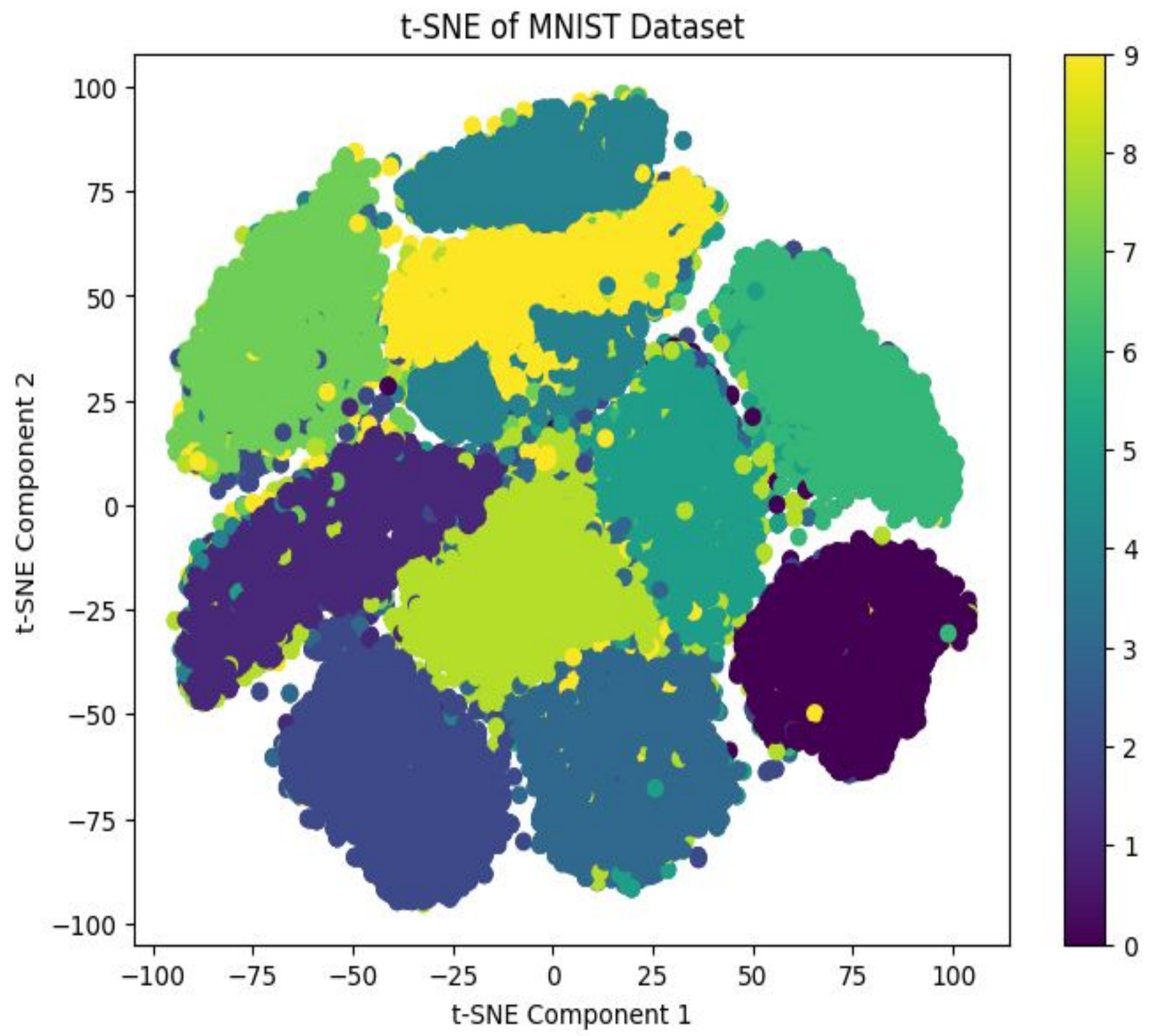


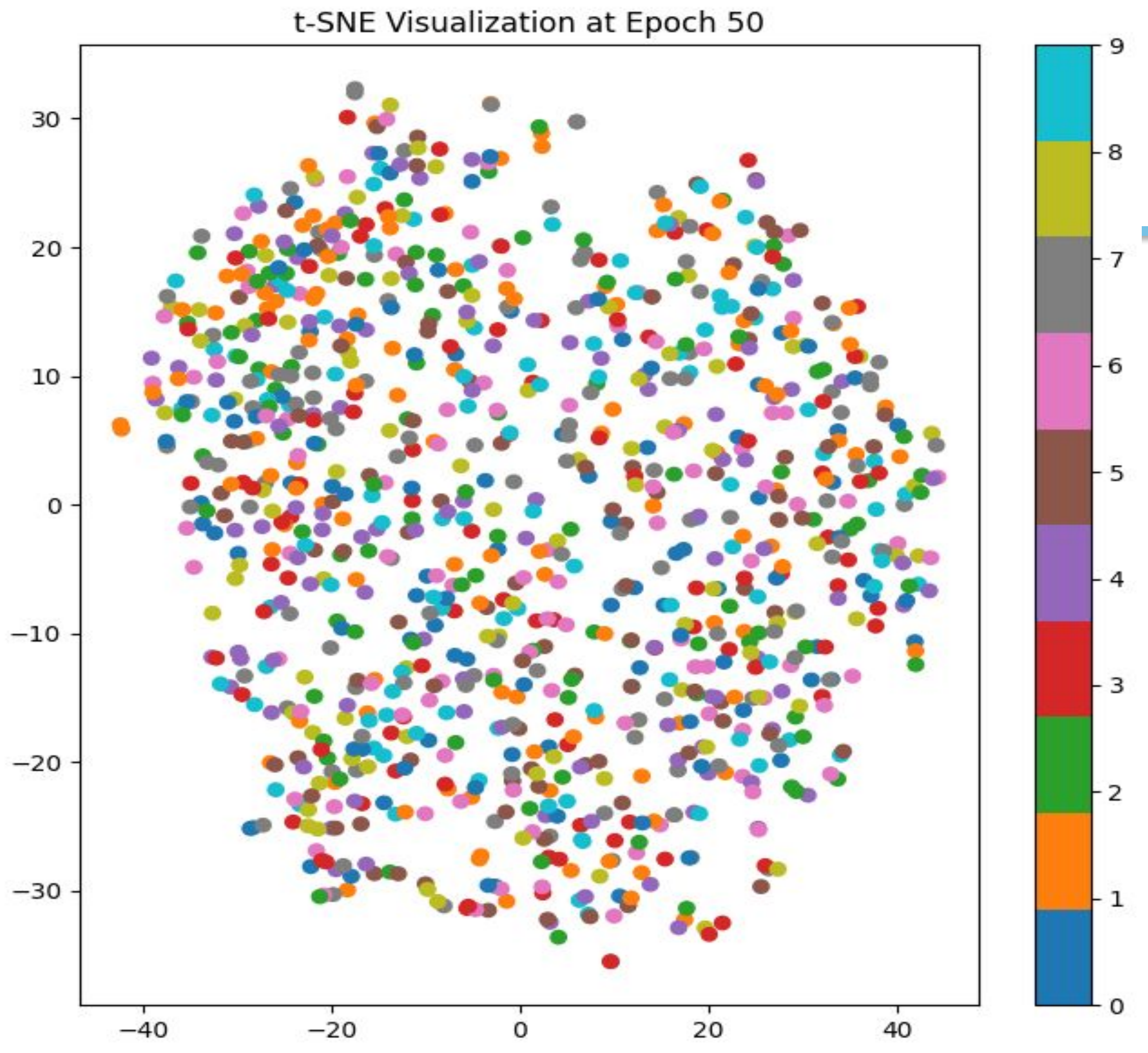
```
Epoch: 10, Test set ELBO: -97.0194320678711, time elapse for current epoch: 7.841630935668945
Eigenvalues (explained variance): [3.4023074  1.68152492]
Eigenvectors (principal components):
 [[ 0.13717494  0.07599877  0.03880922  0.42225939  0.53073089 -0.13695653
   0.42500768  0.3141861   0.43329962  0.16754916]
 [-0.10157077  0.07906349  0.38458931  0.25362393  0.13862933  0.5963175
  -0.44472609  0.31785127 -0.14573398  0.27629069]]
```

- The first principal component explains about twice as much variance as the second (3.40 vs 1.68). This suggests that there's a dominant direction of variation in the latent space.
- Dimensions 4, 5, 7, and 9 have the largest absolute values (0.42225939, 0.53073089, 0.42500768, 0.43329962).
- This suggests these latent dimensions are most important for the primary variation in the MNIST digits.
- Dimensions 3 and 6 have large positive values (0.38458931, 0.5963175), while dimension 7 has a large negative value (-0.44472609).
- This indicates a secondary pattern of variation, possibly capturing different aspects of the digit shapes. Ex. Variation in digit orientation and size.

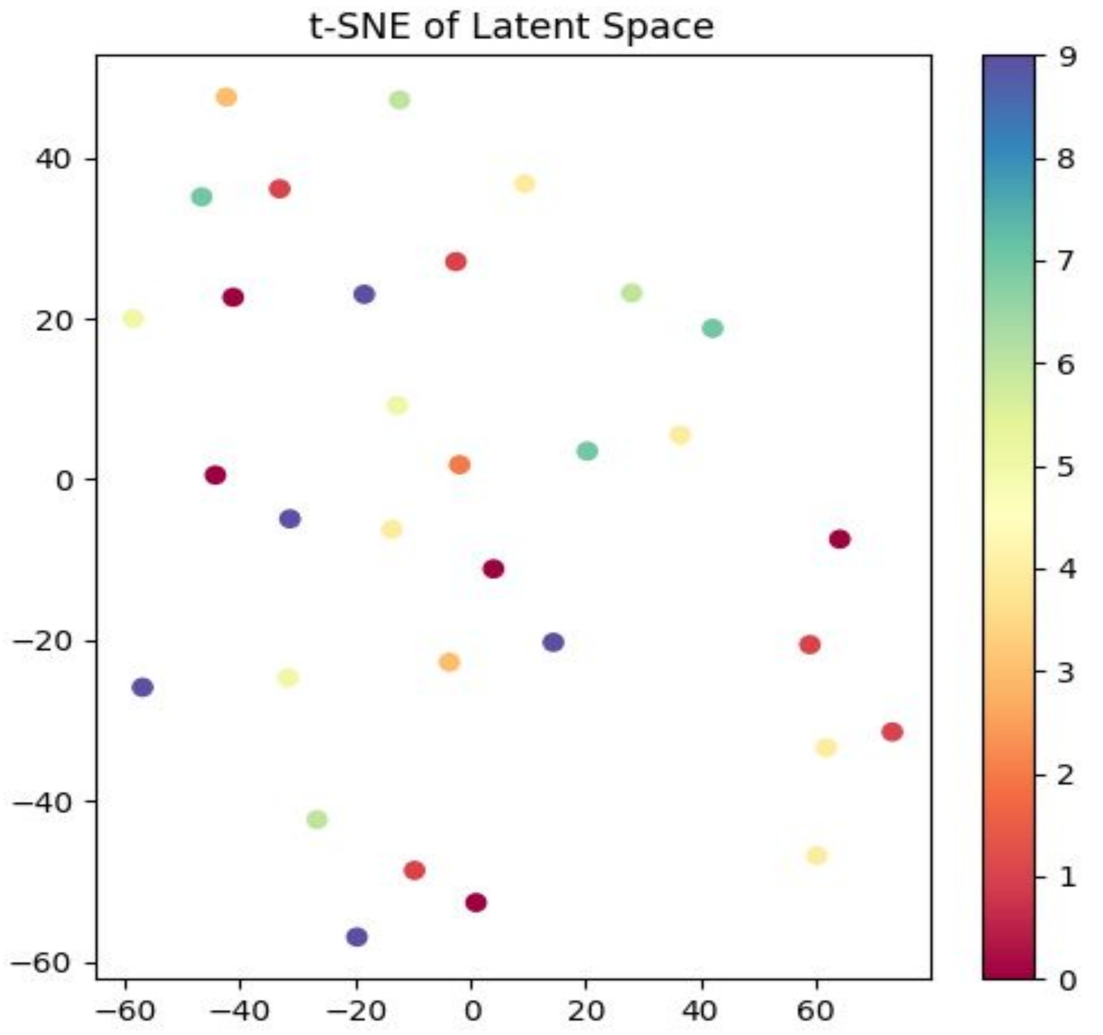
t-SNE
(t-Distributed Stochastic Neighbour Embedding)

- Non-linear dimensionality reduction technique for visualization.
- Preserves local structure by modeling pairwise similarities.
- Minimizes divergence between high-dimensional and low-dimensional distributions.
- Effectively reveals clusters and patterns in complex data.
- Computationally expensive and slow for large datasets.
- Does not preserve global structure, meaning distances between far-apart points can be inaccurate.
- Popular for visualizing high-dimensional data like images and gene expression data.





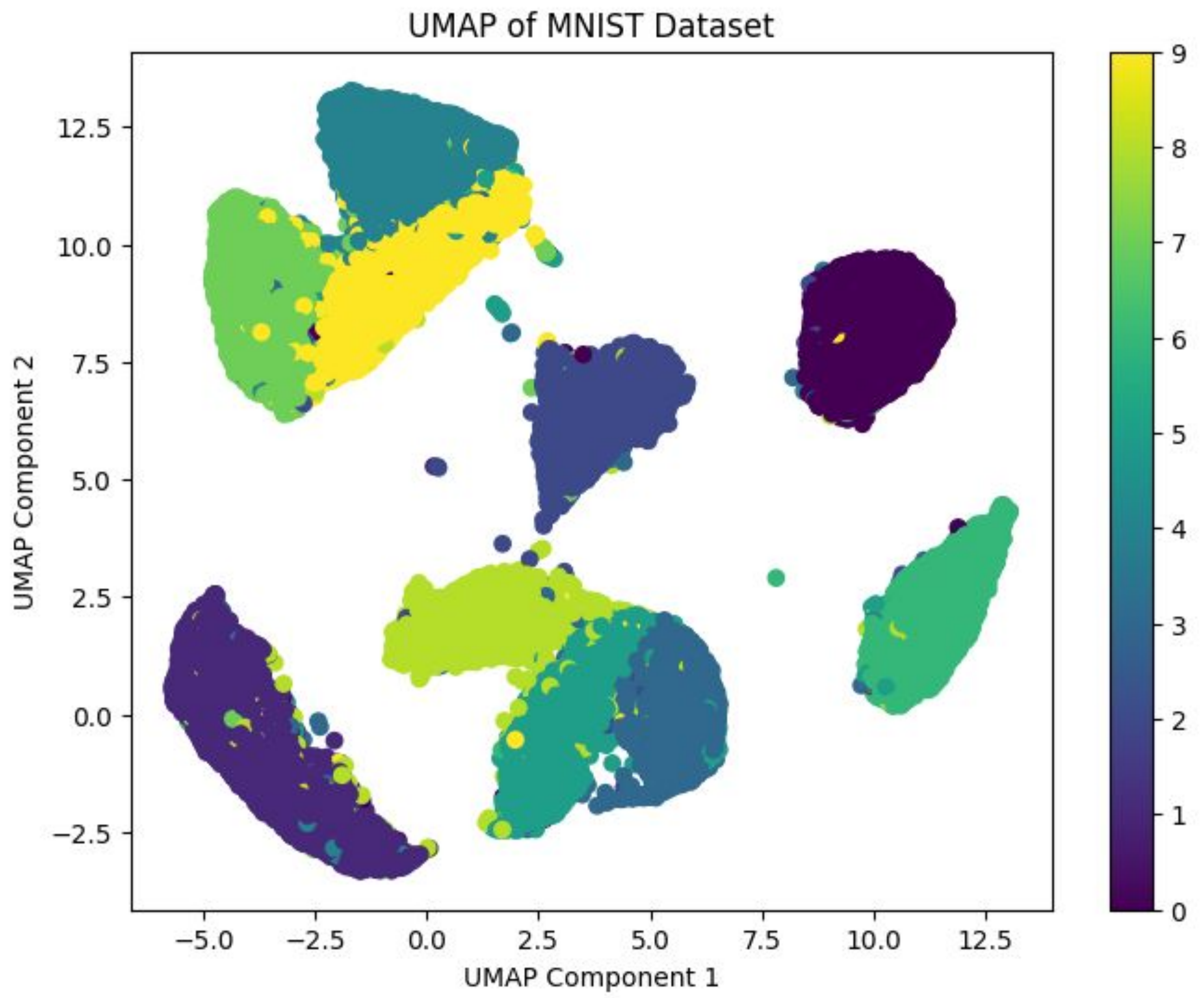
t-SNE on generator latent space of GAN

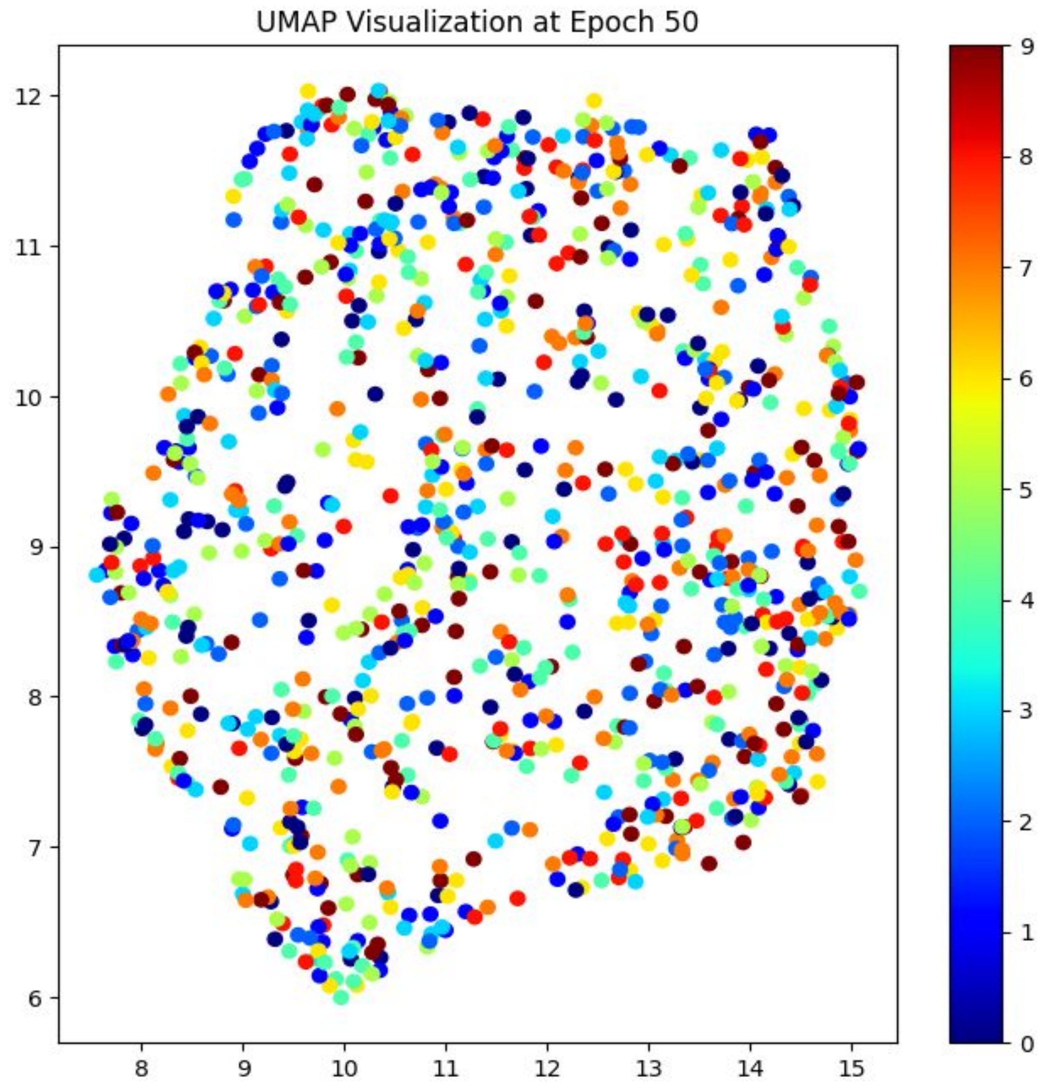


t-SNE on latent space of VAE

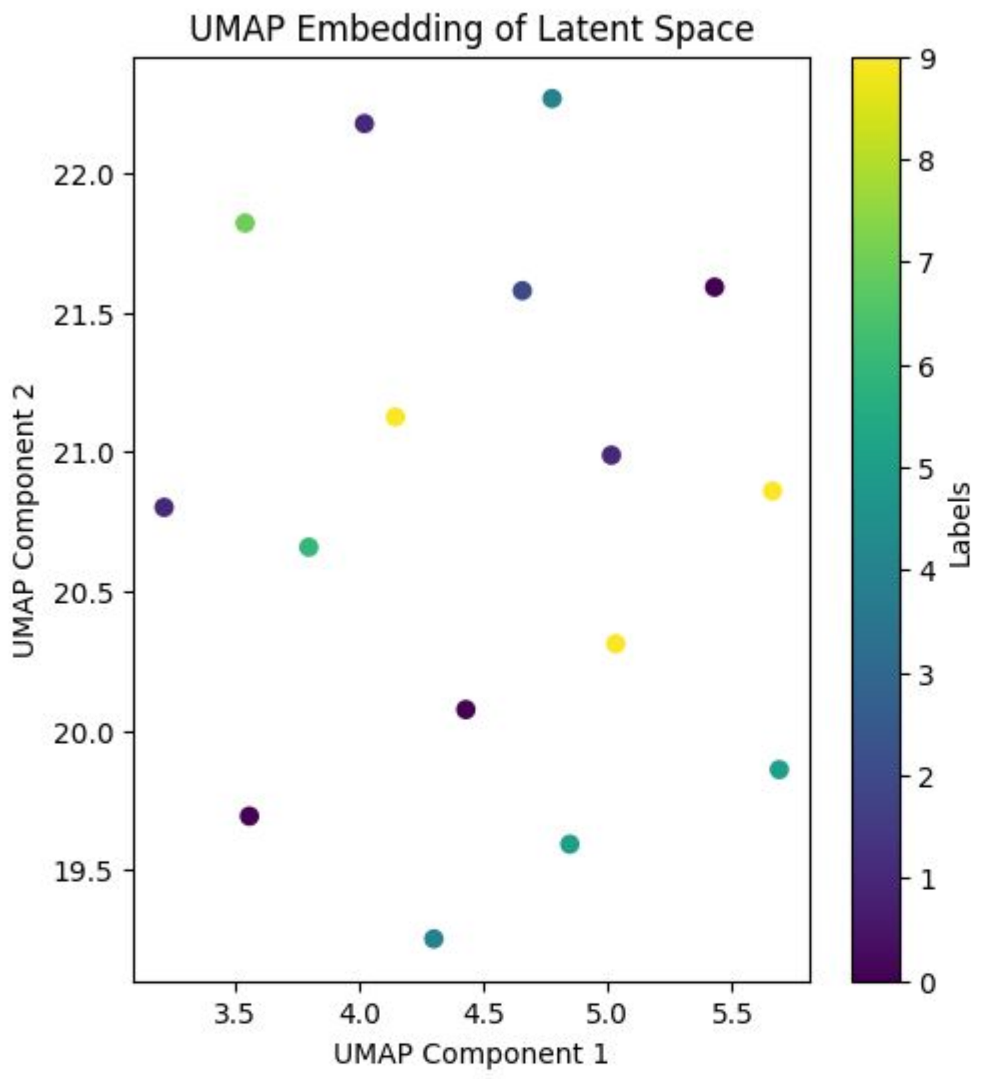
UMAP
(Uniform Manifold Approximation and Projection)

- Non-linear dimensionality reduction algorithm.
- Balances preserving both local and global structure.
- Models data as a high-dimensional graph and optimizes its lower-dimensional representation.
- Faster and more scalable than t-SNE, suitable for large datasets.
- Preserves more global structure compared to t-SNE.
- Highly interpretable and used for clustering, visualization, and manifold learning.
- Effective in visualizing complex datasets like single-cell data and word embeddings.





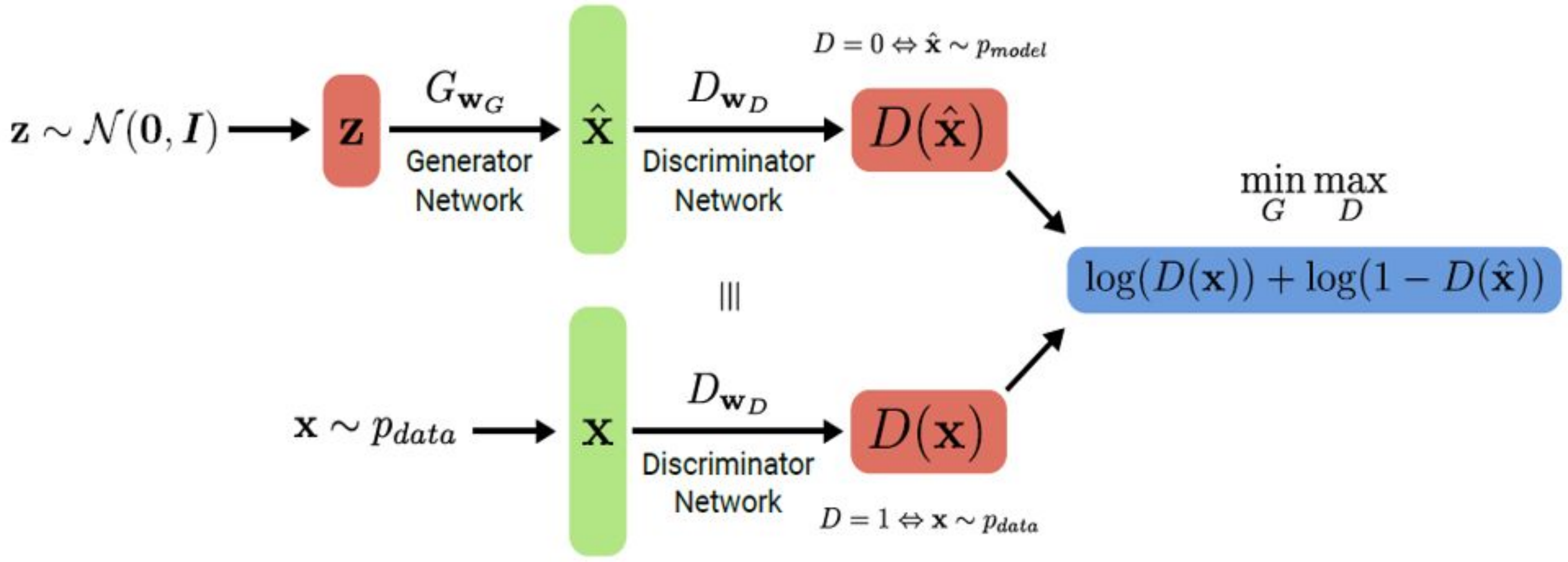
UMAP on generator latent space of GAN



UMAP on latent space of VAE

Exploring Neuron Activations

Generative Artificial Intelligence :


$\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$
$\mathbf{z}$
$G_{\mathbf{w}_G}$
Generator Network
$\hat{\mathbf{x}}$
$D_{\mathbf{w}_D}$
Discriminator Network
$D = 0 \Leftrightarrow \hat{\mathbf{x}} \sim p_{model}$
$D(\hat{\mathbf{x}})$
$\min_G \max_D$
$\log(D(\mathbf{x})) + \log(1 - D(\hat{\mathbf{x}}))$
$\parallel$
$\mathbf{x} \sim p_{data}$
$\mathbf{x}$
$D_{\mathbf{w}_D}$
Discriminator Network
$D(\mathbf{x})$
$D = 1 \Leftrightarrow \mathbf{x} \sim p_{data}$

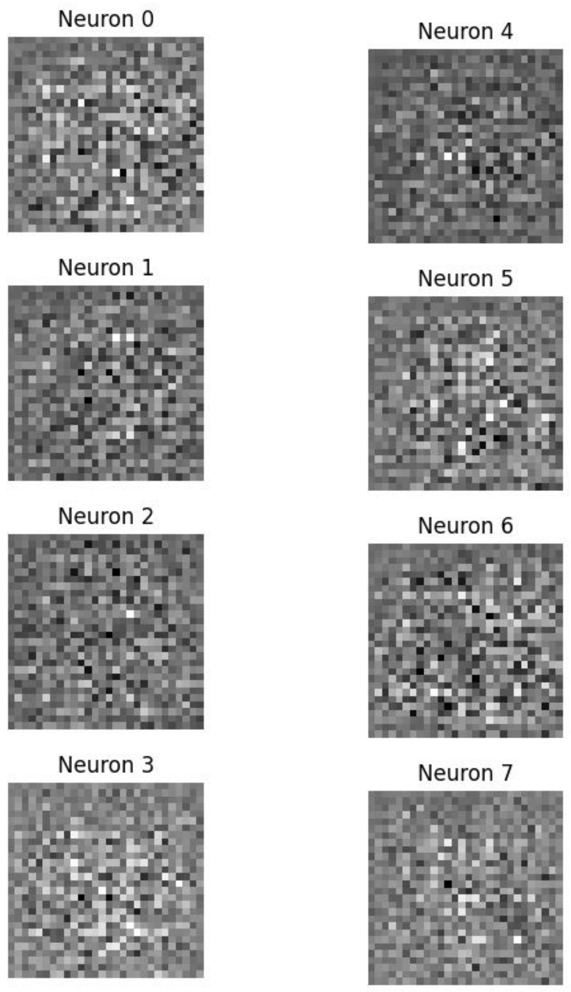
Exploring Neuron Activations

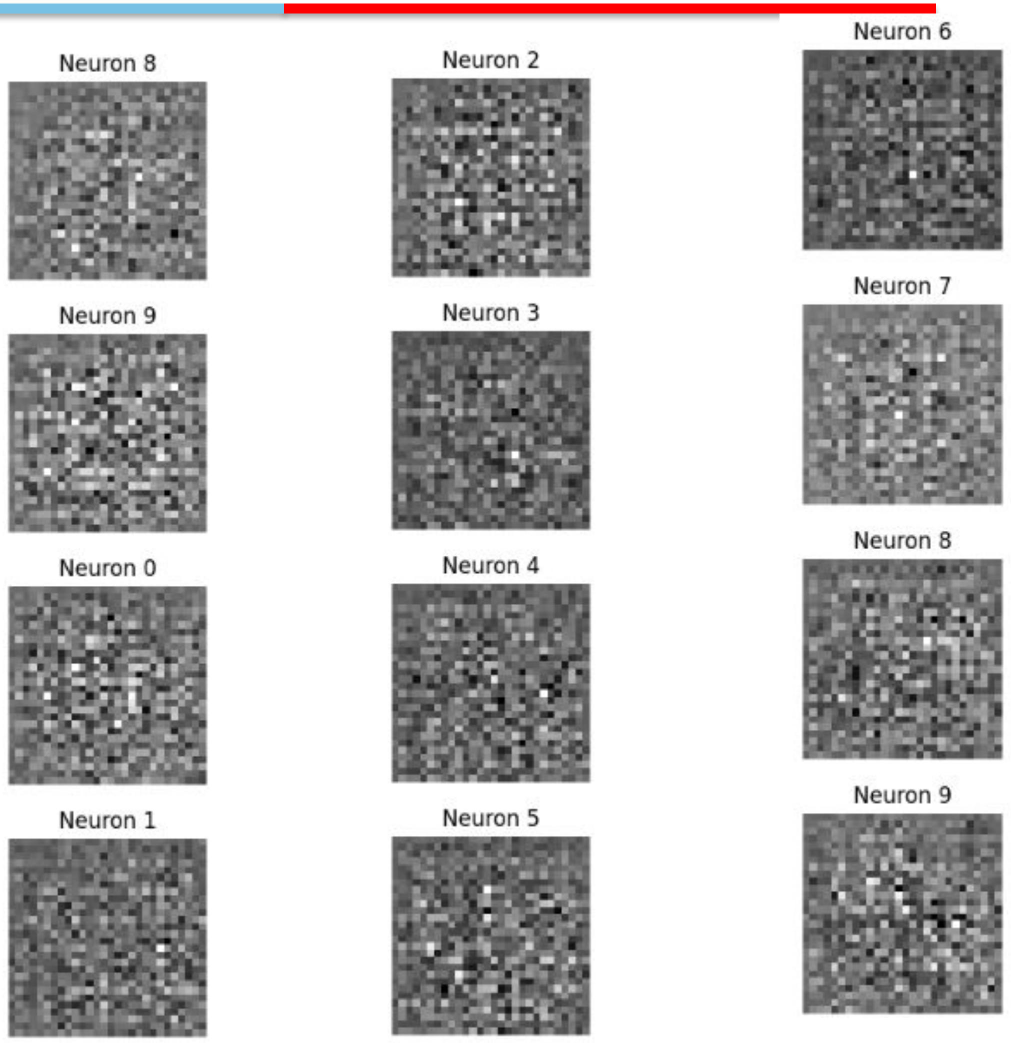
The function performs **gradient ascent** on the latent vector to **maximize the activation** of a specific neuron in the generator. It does this by:

1. Tracking the neuron's activation.
2. Computing gradients with respect to the latent vector.
3. Updating the latent vector iteratively to increase the neuron's activation.
4. Clipping the latent vector to stay within a valid range for the generator.

In the context of GANs, this allows you to understand what kind of input (latent vector) will strongly activate specific neurons in the generator, and potentially gain insights into what features those neurons are responsible for.

Exploring Neuron Activations





GradCam:

What is GradCAM?

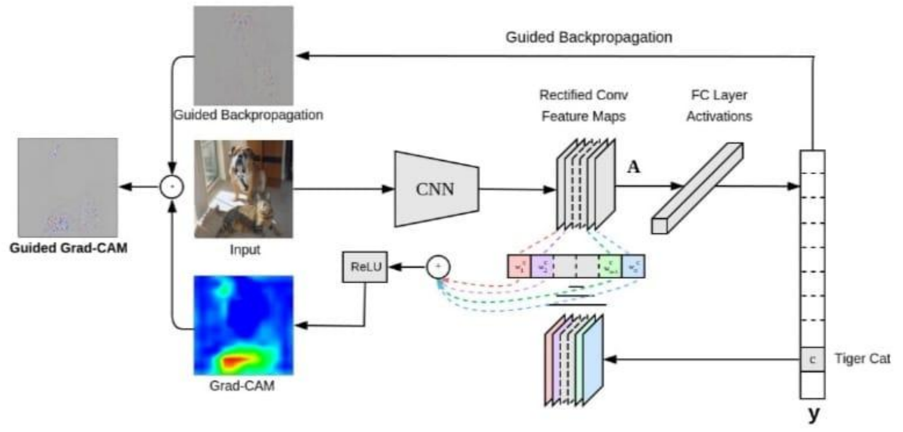
- GradCAM is a **visual explanation technique** for CNN-based models.
- It provides **class-discriminative localization**, highlighting the important regions of an input image contributing to a model's prediction.

Basic Functionality:

- Works with **any CNN architecture** without modifying the model.
- Generates heatmaps that show **where the model is looking** when making predictions.

Working Principle:

- **Uses gradients** of the target class flowing into the last convolutional layer.
- **Highlights feature maps** that have the most influence on the model's output.
- Produces a **coarse localization map** showing areas in the input image important for predicting the target class

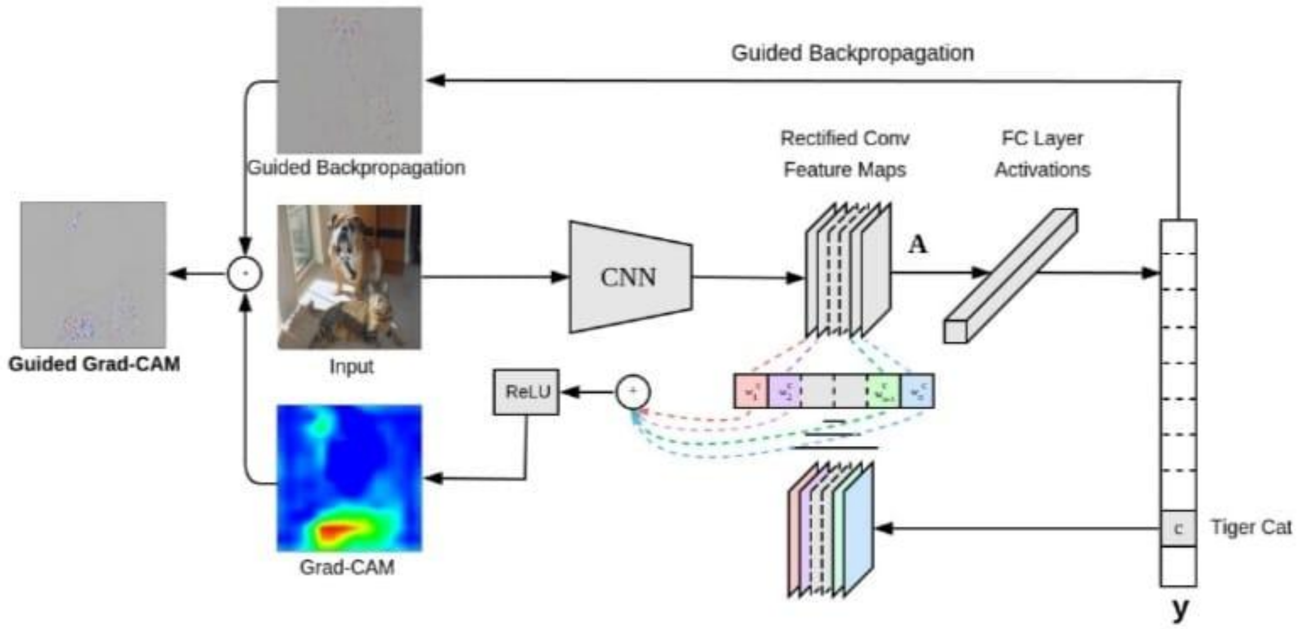


1.

$$\alpha_k^c = \overbrace{\frac{1}{Z}\sum_i\sum_j}^{\text{global average pooling}} \underbrace{\frac{\partial y^c}{\partial A_{ij}^k}}_{\text{gradients via backprop}}$$

Where:

- y^c is the score for the target class c.
- A^k is the k-th feature map in the convolutional lay



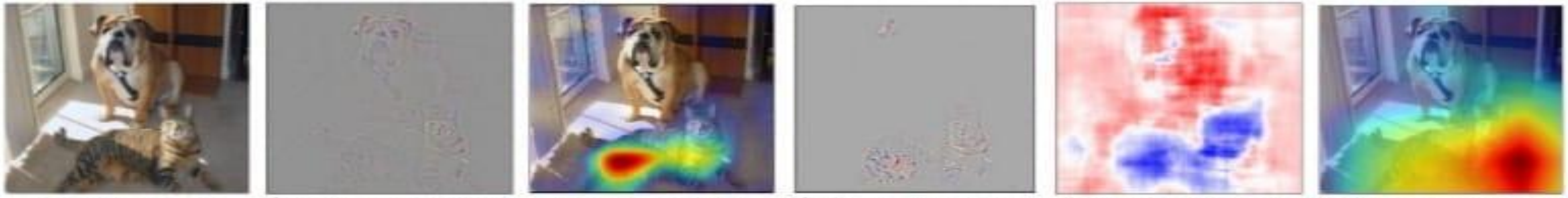
2. F^k to be the global average pooled output

$$F^k = \frac{1}{Z}\sum_i \sum_j A^k_{ij}$$

3. CAM computes the final scores by,

$$Y^c = \sum_k w^c_k \cdot F^k$$

Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization 3



(a) Original Image (b) Guided Backprop 'Cat' (c) Grad-CAM 'Cat' (d)Guided Grad-CAM 'Cat' (e) Occlusion map 'Cat' (f) ResNet Grad-CAM 'Cat'

Resources and References

Original models for DCGAN and CVAE:
https://www.tensorflow.org/tutorials/generative/dcgan, https://www.tensorflow.org/tutorials/generative/cvae

Modified notebooks (Please use BITS email ID):
https://drive.google.com/drive/folders/1qOwP3t2dRgOfUOjRPB3dz-7O32to-QZI?usp=sharing

References:
GAN: https://youtu.be/Gib_kiXgnvA?si=O8WaLmVfSsDyXdp7
https://www.geeksforgeeks.org/generative-adversarial-network-gan/,
https://aws.amazon.com/what-is/gan/

VAE: https://www.youtube.com/watch?v=9zKuYvjFFS8&t=5s&pp=ygUNZ2FuIGV4cGxhaW5lZA%3D%3D
https://www.geeksforgeeks.org/variational-autoencoders/

PCA,UMAP,t-SNE: https://www.youtube.com/watch?v=o_cAOa5fMhE,
https://www.bioinformatics.babraham.ac.uk/training/10XRNASeq/Dimension%20Reduction.pdf
https://medium.com/@aastha.code/dimensionality-reduction-pca-t-sne-and-umap-41d499da2df2
https://www.kaggle.com/code/samuelcortinhas/intro-to-pca-t-sne-umap

GradCam:Selvaraju et al., 2017. Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization innovate achieve lead

Thank You