Generative Adversarial Networks (GANs)

Seminar Presentation

Guided by:

Dr. Deepa V.

Presented by:

Ajay T Shaju (SJC20AD004)

THE STORY

Young Photographer captured photos in a rainy day, and he got this:

OUTLINE

- **Introduction**
  - Discriminative and Generative models
- **Details of Generative Adversarial Networks**
  - **Components** – Generator & Discriminator
  - **Architecture**
  - **Loss Function**
  - **Training**
  - **Challenges of GANs** – Mode Collapse & Non-Convergence
- **Types of GANs**
- **Uses of GANs**
- **Recent Application of GAN**
  - FIGAN: CNN Learned Features

INTRODUCTION

Machine Learning Models

Discriminative Models   Generative Models

- **Discriminative Models** – Separates the **data points into different classes.**
  Mostly **Supervised in nature.**
  Eg – Logistic Regression.
- **Generative Models** – Models that can **generate new data points** within its training data.
  Mostly **Un-Supervised in nature**.
  Eg – GANs.

Generative Adversarial Networks

- **Introduced by Ian J Goodfellow in June 2014.** Generative Adversarial Networks (GANs) is a type of **deep learning model** that are used to **generate new data** that is similar to the data it was trained on.
- As per the creator, GAN is a framework for **estimating generative models via an adversarial process.** [1]
- GAN is a type of **unsupervised machine learning.**

Smooth transition from 1 to 5 (Image source: NIPS GAN Paper [2])

GAN COMPONENTS – GENERATOR & DISCRIMINATOR

GAN composed of two neural networks

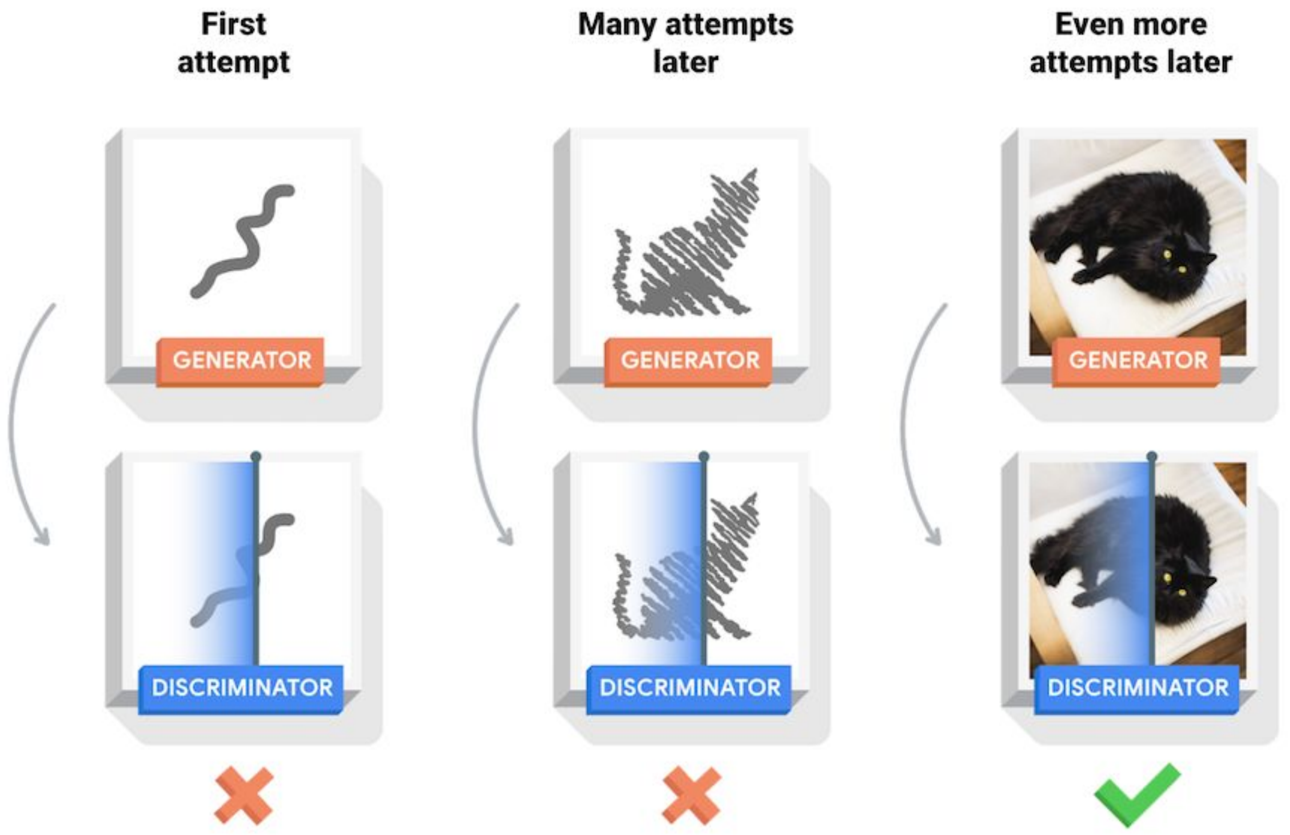
Generator

Responsible for **creating new data (i.e fake data)**

Discriminator

Responsible for **distinguishing between real and fake data.**

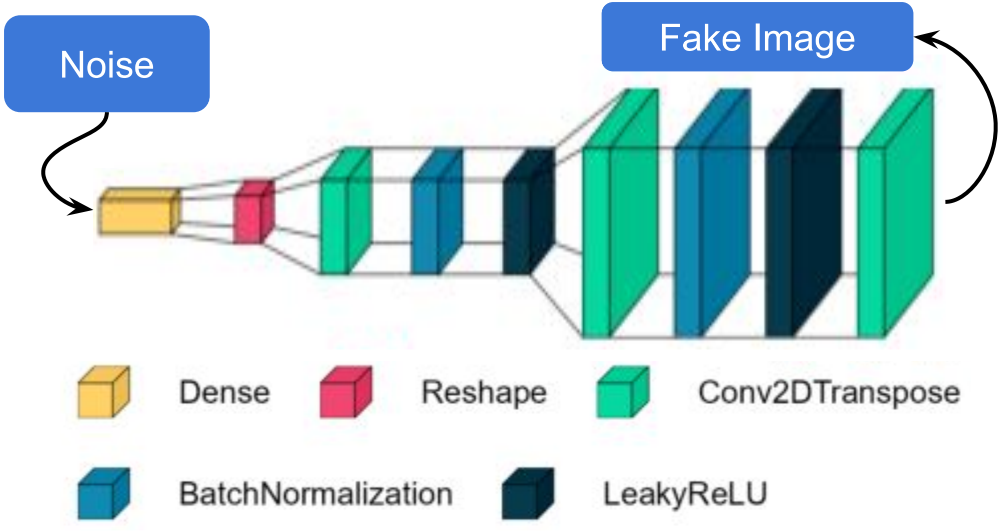
- GANs were inspired by the **game theory**, **the generator and discriminator will compete each other to achieve the** **Nash equilibrium**(i.e achieving the desired outcome not by deviating from initial strategy) during its training process. [2]



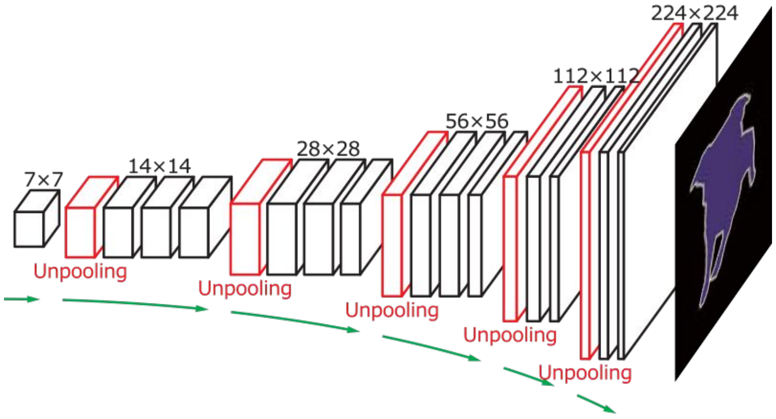
Working of Generator And Discriminator
Image source: TensorFlow

GENERATOR

- The **Generator** is a **Deconvolution and Unpooling Layered Network (Image Generation).**



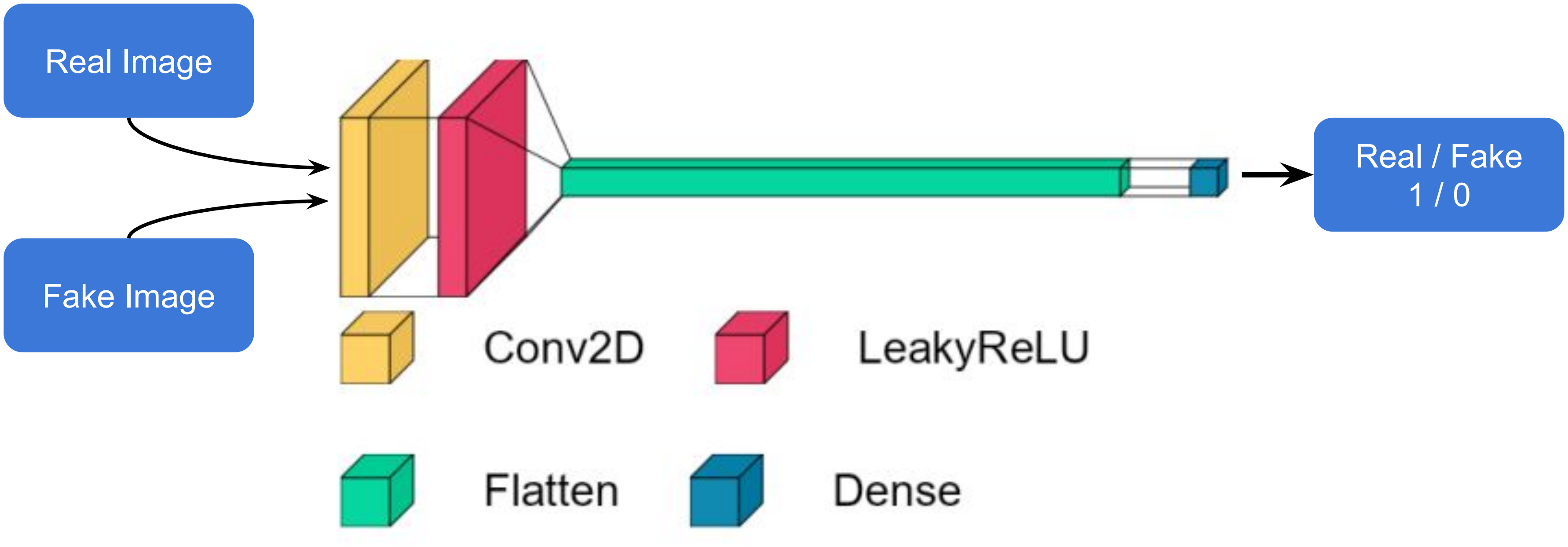
Simple Generator Architecture.
(Image source: Author)



Deconvolution Network
(Image source: Open Genus)

DISCRIMINATOR

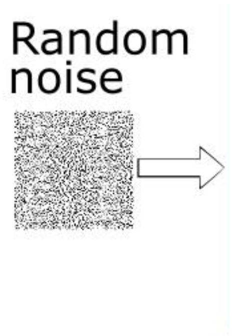
- The **discriminator** is just a **normal binary classifier(CNN).**



Simple Discriminator Architecture. (Image source: Author)

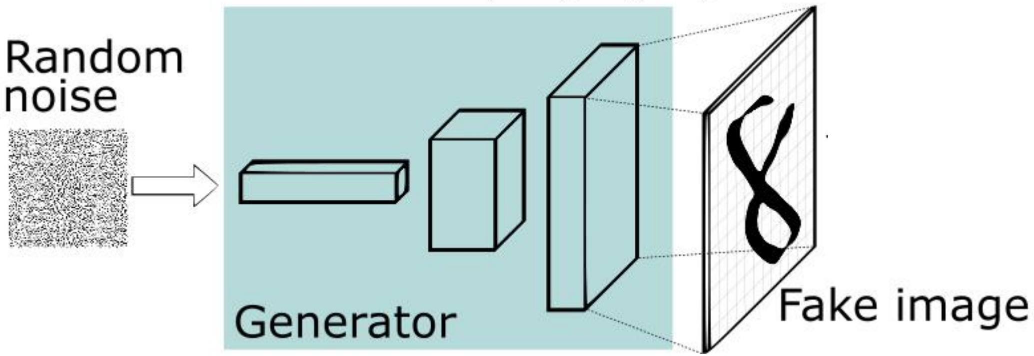
GAN ARCHITECTURE

GAN ARCHITECTURE

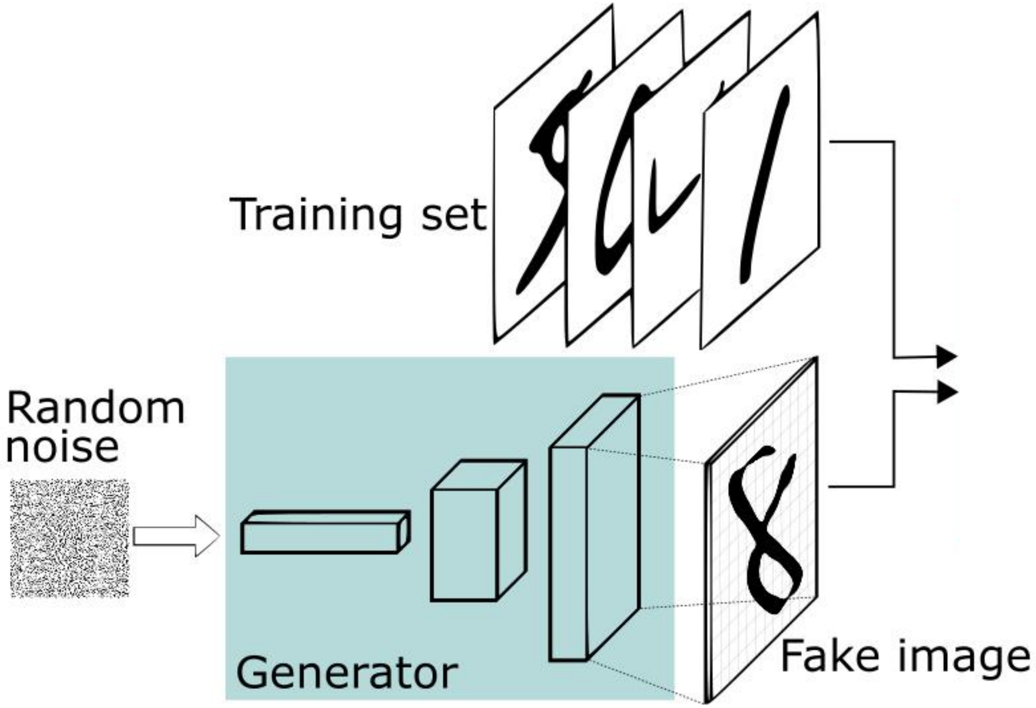


Architecture of GAN (Image source: Medium: Analytics Vidhya)

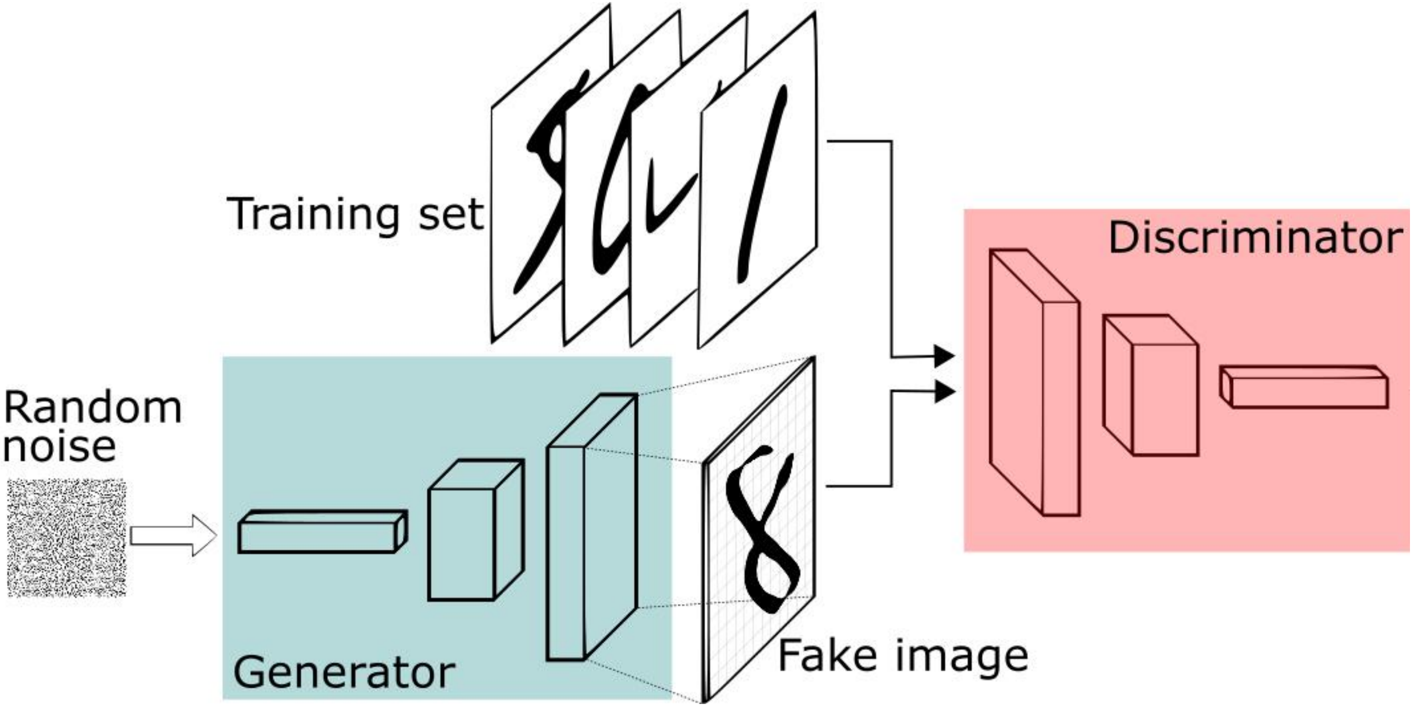
GAN ARCHITECTURE



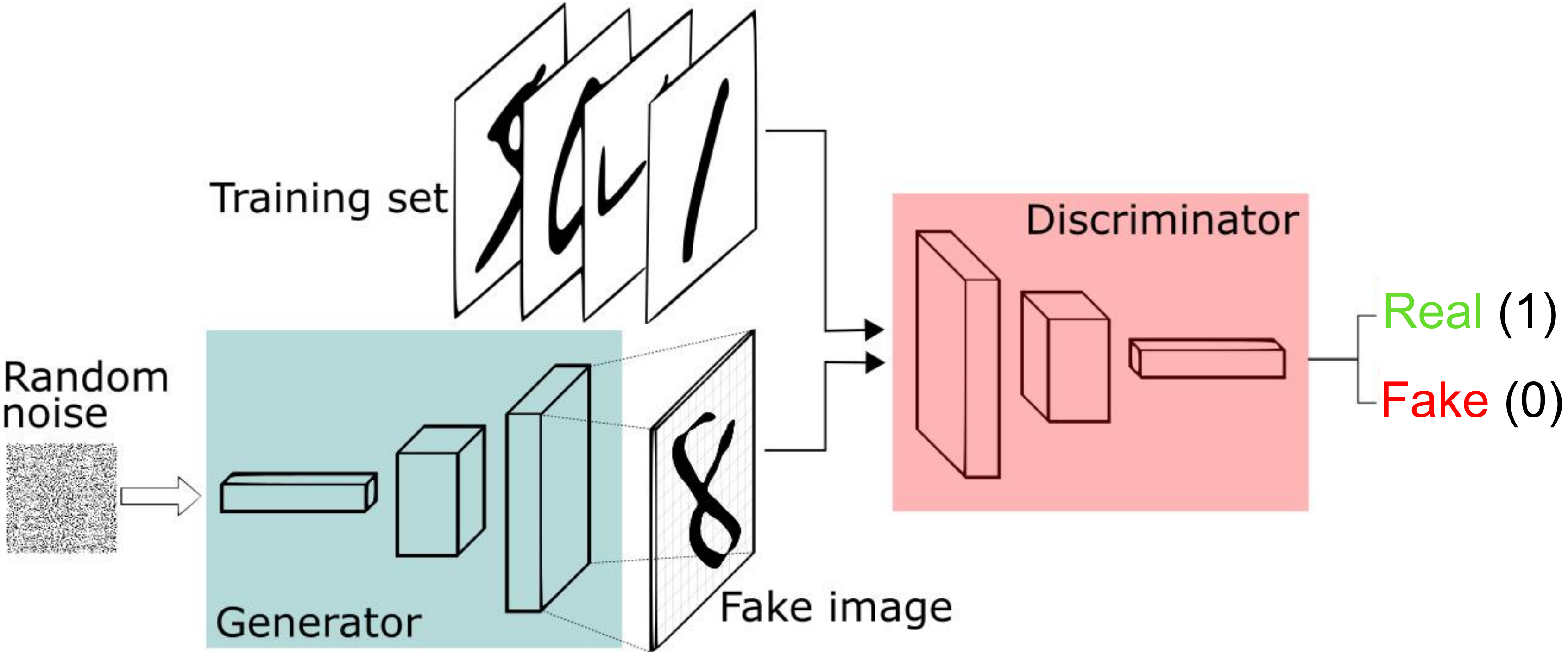
GAN ARCHITECTURE



GAN ARCHITECTURE



GAN ARCHITECTURE



Architecture of GAN (Image source: Medium: Analytics Vidhya)

GANs LOSS FUNCTION

- The **Standard GAN Loss Function**, also known as the **Min-Max Loss**[1].
- The Standard GAN loss function is divided into: **Discriminator loss and Generator loss.**

GANs LOSS FUNCTION

- The **Standard GAN Loss Function**, also known as the **Min-Max Loss**[1].
- The Standard GAN loss function is divided into: **Discriminator loss and Generator loss.**

$$\underset{G}{min}\underset{D}{max}\, V\,(D,G)\; =\; E_{x\sim Pdata(x)}[\log D(x)]\; +\; E_{z\sim Pz(z)}[\log\,(1-D(G(z)))]$$

Min-Max Loss Function

- **E** represents the expected value or the average.
- **x** represents real data samples drawn from the real data distribution
- **D(x)** is the output of the Discriminator when it tries to distinguish real data from fake data.

- **z** represents random noise.
- **G(z)** is the output of the Generator when given random noise as input.
- **1−D(G(z))** represents the probability that the Discriminator assigns to the generated data being fake (1 minus the probability it assigns to it being real).

FOR DISCRIMINATOR

We are familiar with the equation of Binary Cross-Entropy:

$$L(y,\, \hat{y}) \;=\; -\left(y \cdot \log \hat{y} \;+\; (1-y) \cdot \log\left(1-\hat{y}\right)\right)$$

[**L(x, x')** is the loss.]

FOR DISCRIMINATOR

We are familiar with the equation of Binary Cross-Entropy:

$$L(y,\, \hat{y}) \;=\; -\left(y \cdot \log \hat{y} \;+\; (1 - y) \cdot \log\left(1 - \hat{y}\right)\right)$$

[**L(x, x’)** is the loss.]

We substitute y = 1 for real images and y = 0 for fake images in the BCE equation, to get:

$$L(\text{Discriminator}) \;=\; \min[-[\log\left(D(x)\right) \;+\; \log\left(1 - D(G(z))\right)]]$$

FOR DISCRIMINATOR

We are familiar with the equation of Binary Cross-Entropy:

$$L(y,\, \hat{y}) \;=\; -\left(y \cdot \log \hat{y} \;+\; (1-y) \cdot \log\left(1-\hat{y}\right)\right)$$

[**L(x, x')** is the loss.]

We substitute y = 1 for real images and y = 0 for fake images in the BCE equation, to get:

$$L(\text{Discriminator}) \;=\; \min[-[\log\left(D(x)\right) \;+\; \log\left(1-D(G(z))\right)]]$$

Removing the negative sign will change min to max:

$$L(\text{Discriminator}) \;=\; max[\log\left(D(x)\right) \;+\; \log\left(1-D(G(z))\right)]$$ **— (1)**

Maximizing the loss function means to put D(x) = 1, and D(G(z))) = 0 [Note: **log(1) = 0**]

So, loss function will result in **0 (Fake image).**

FOR GENERATOR

- Generator loss is calculated from the discriminator loss.
- Generator will try to fool the discriminator into classifying the fake data as real data. This implies that the **generator tries to minimise the second term in the discriminator loss equation**.

$$L(\text{Generator}) = \min[\log(1 - D(G(z)))]$$ **— (2)**

Minimizing the loss function means to put **D(G(z))) = close to 1**

So, loss function will result in **1** (Real image).

FOR GENERATOR

- Generator loss is calculated from the discriminator loss.
- Generator will try to fool the discriminator into classifying the fake data as real data. This implies that the **generator tries to minimise the second term in the discriminator loss equation**.

$$L(\text{Generator}) = \min[\log(1 - D(G(z)))] \text{ — } \textbf{(2)}$$

Minimizing the loss function means to put **D(G(z))) = close to 1**

So, loss function will result in **1** (Real image).

By summing both the equations (1) & (2), we get:

$$\underset{G}{min}\,\underset{D}{max}\, V(D, G) = E_{x \sim Pdata(x)}[\log D(x)] + E_{z \sim Pz(z)}[\log(1 - D(G(z)))]$$

Min-Max Loss Function

GAN TRAINING

- Start GAN training in **alternating periods**.
- The discriminator trains for one or more epochs then the generator's turn and this step repeats.
- Careful training is required, as **most challenges of GANs are in training phase.**

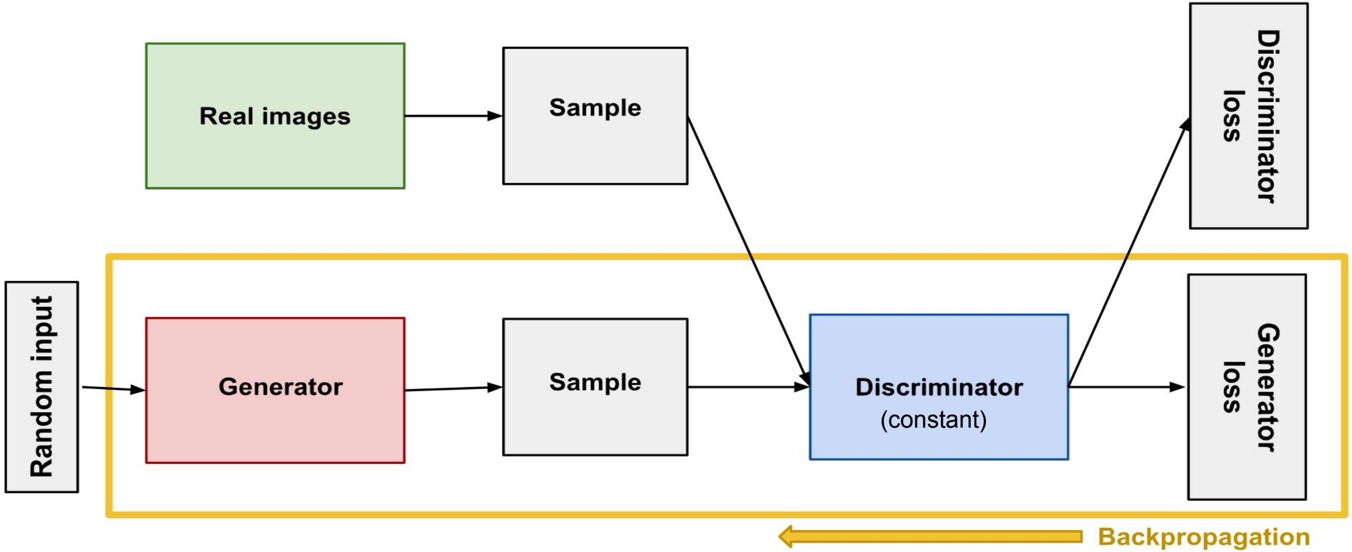
- Optimal training duration will depend on:
  - **Complexity of the data being generated**
  - **Size of the training dataset, and**
  - **Desired quality of the generated data.**

Stopping the training:

- **Generator is able to produce high-quality samples** that are identical to real data.
- **Discriminator is no longer able to classify real and fake data** with high accuracy.

GENERATOR TRAINING

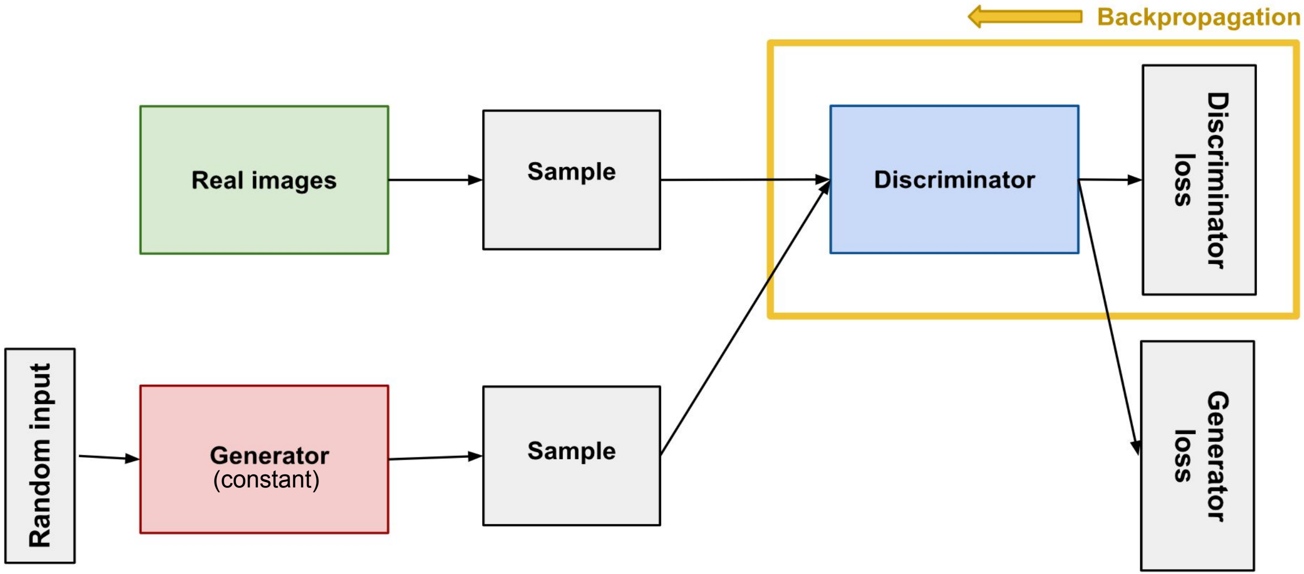
- The discriminator is kept constant during the generator training phase. Else the generator would try to hit a target (output of discriminator) and never converge.
- Random noise → Generator output → Discriminator classification → Loss calculation from discriminator classification → **Backpropagates through both the discriminator and generator to obtain gradients → Use gradients to change only the generator weights.**



Backpropagation in generator training. (Image source: Google Developers)

DISCRIMINATOR TRAINING

- During discriminator training the generator is paused. Its weights remain constant while it produces examples for the discriminator to train on.
- Fake Image → Discriminator classification → Loss calculation → Backpropagation → **Use gradients to change discriminator weights**.

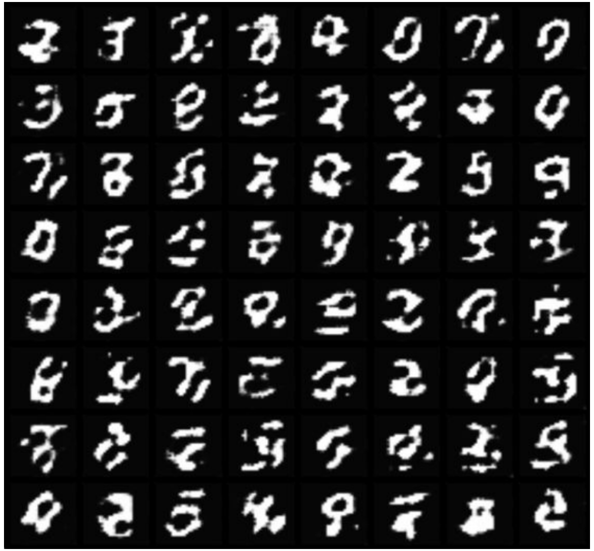


Backpropagation in discriminator training. (Image source: Google Developers)

GAN WORKING ON MNIST DATASET

Ground Truth

Epoch 10



Epoch 100

Progress during training the GAN (Image source: Neptune AI)

GAN CHALLENGES

→ Mode Collapse

- Issue: GAN **fails to capture and generate diverse samples from the entire data distribution**. Instead, it may **focus on generating samples from only a few modes or patterns** in the data. This results in a **lack of diversity** in the generated outputs.

- Why it's a challenge: It **makes the generated data less useful** for tasks like data augmentation or creative content generation.

→ Non-Convergence

- Issue: **The discriminator performance gets worse when generator outperforms it**, thus **lowering the accuracy of discriminator**.

- Why it's a challenge: The **discriminator feedback gets less meaningful over time** and gives **random feedback to generator**, by which the **generator starts to train on junk feedback**, reducing its own quality.

TYPES OF GANs

A **normal GAN may be unstable with different challenges** like Mode Collapse, Non-Convergence, Uninformative Loss, and Sensitive Hyperparameter. **To solve these problem to an extend, new types of GANs like DCGAN** [3]**, SAGAN** [5]**, VAE-GAN** [6], **TransGAN** [7] **and [500+ other GANs] were introduced.**

Conditional GAN (CGAN)

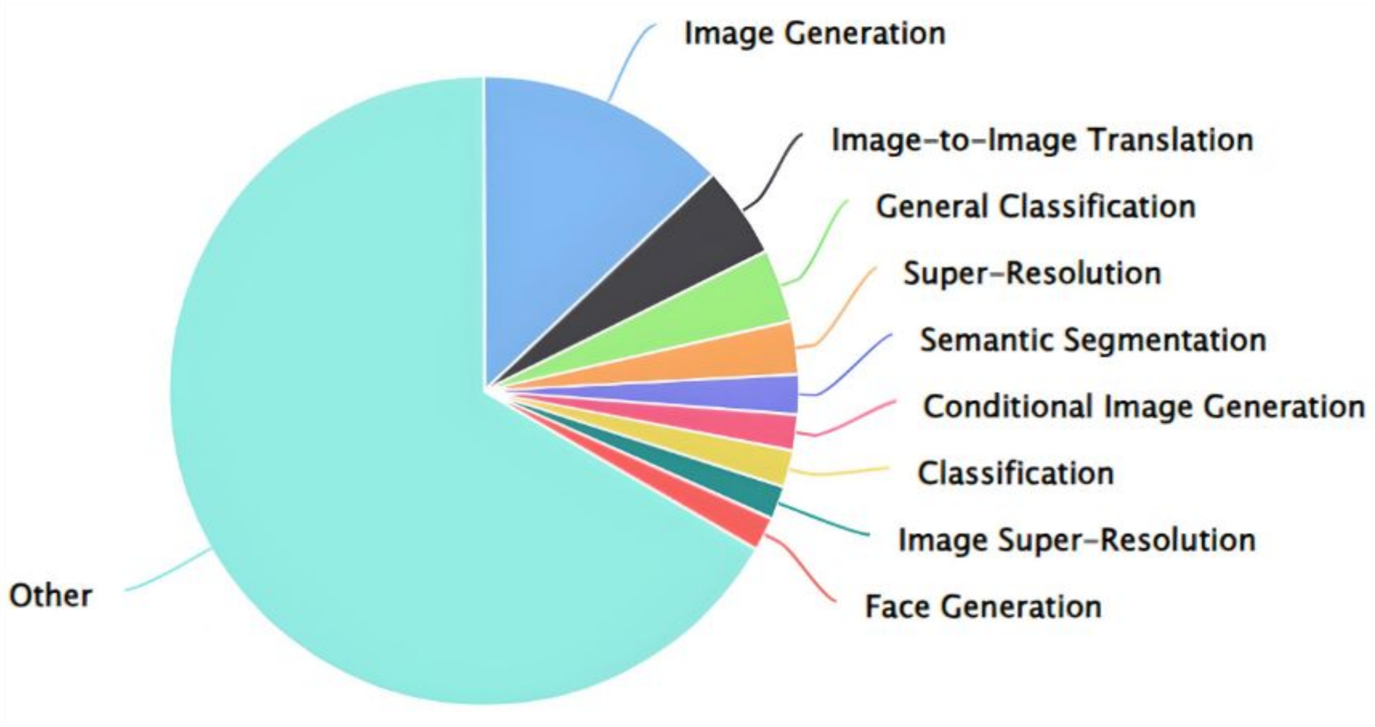
- *Generative adversarial nets can be extended to a conditional model if both the generator and discriminator are conditioned on some extra information y.*
  – Conditional Generative Adversarial Nets Paper [10]

$$\underset{G}{min}\,\underset{D}{max}\,V(D,G) = E_{x\sim Pdata(x)}[\log D(x\,|\,y)] + E_{z\sim Pz(z)}[\log(1 - D(G(z\,|\,y)))]$$

New **y** term is introduced.

Adding **y** term gives more control over the output.

USE CASES OF GANs



Source: [Papers with code]

GAN RECENT APPLICATION

Visualizing CNN’s Learned Features using FIGAN

Research Paper Title: Feature Interpretation Using Generative Adversarial Networks (FIGAN): A Framework for Visualizing a CNN’s Learned Features (January 2023).

The brief of the paper:

- Discusses the limitations of current methods for **explainability in medical imaging classification tasks**.
- Proposed **FIGAN framework** that uses a **Conditional Generative Adversarial Network(CGAN)** for visualizing CNN learned features.
- How **FIGAN differs from other methods** in its approach to visualizing a CNN's learned features.
- Provides an example of how FIGAN has been used in a **real-world medical imaging application**.

Problem that the research paper addresses:

- **Limited explainability** in medical imaging classification tasks using CNN.
- Lack of **transparency limits the translation** of these methods into clinical radiology practice.
- Most **CNN’s architectural complexity** makes it difficult to explain the imaging features.

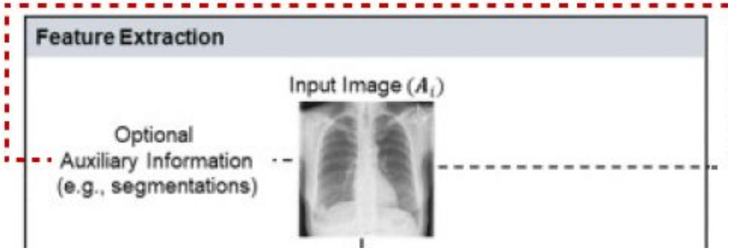
How GANs are used to solve the problem:

- Researchers used **FIGAN Framework**(Type of GAN) that uses **CGAN** to create images representing **CNN's key features**.

- Solution involves a **two-stage training** process:
    - First: train a **CNN on the medical imaging dataset** to be analyzed.
    - Second: train the **FIGAN model to generate images of main features** from CNN

- The authors also perform a series of experiments to study the **effect of auxiliary segmentations, training sample size, and image resolution** on FIGAN's ability to provide consistent and interpretable synthetic images.

ARCHITECTURE

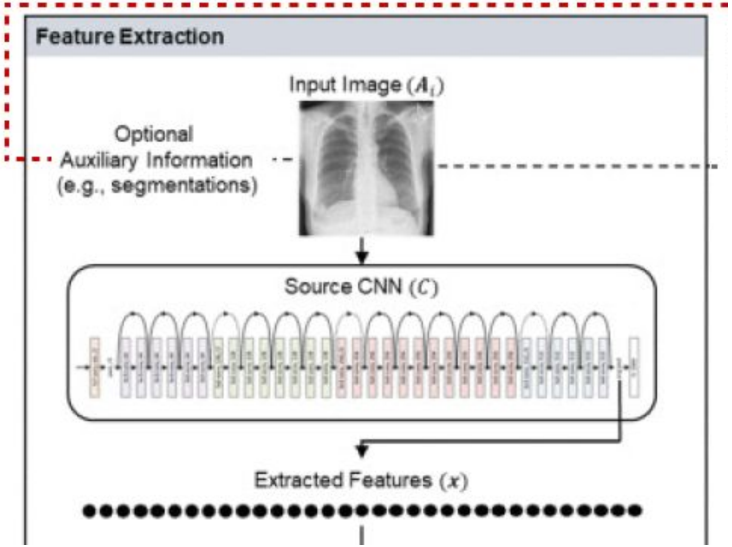


Three main steps:

- Feature Extraction (CNN),
- Generative Model Training (FIGAN), and
- Generated Image Analysis (to gain insights into the CNN's decision-making process and identify which features are most important for classification).

Image Source: [9]

ARCHITECTURE

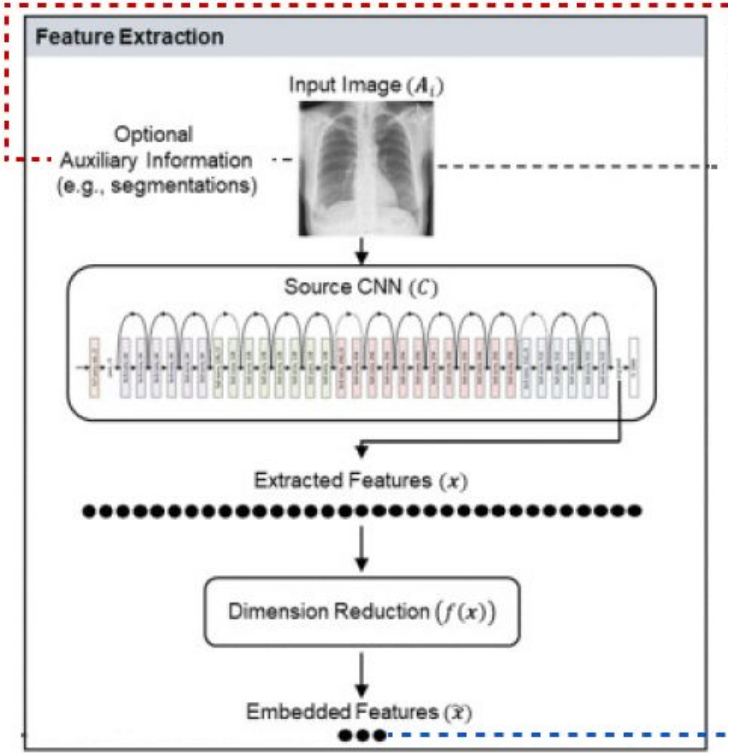


Three main steps:

- Feature Extraction (CNN),
- Generative Model Training (FIGAN), and
- Generated Image Analysis (to gain insights into the CNN's decision-making process and identify which features are most important for classification).

Image Source: [9]

ARCHITECTURE

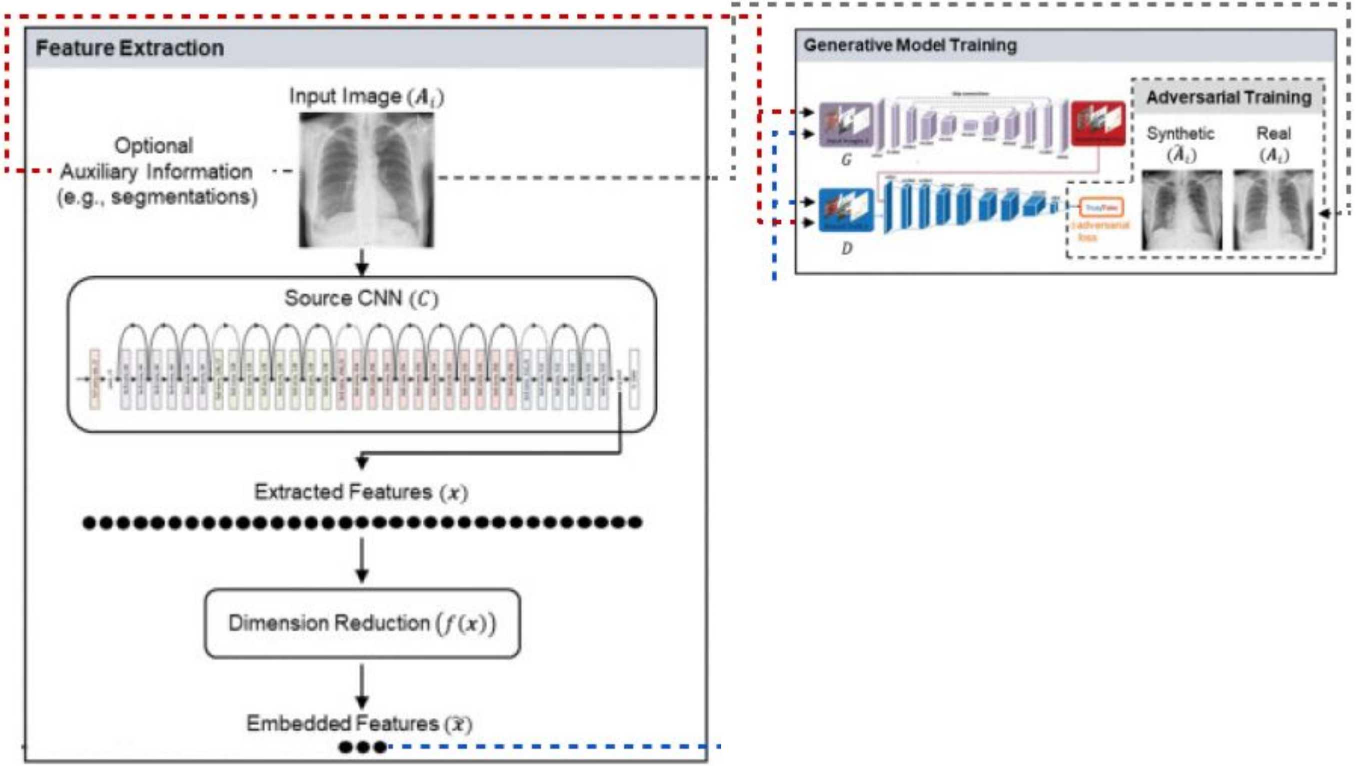


Three main steps:

- Feature Extraction (CNN),
- Generative Model Training (FIGAN), and
- Generated Image Analysis (to gain insights into the CNN's decision-making process and identify which features are most important for classification).

Image Source: [9]

ARCHITECTURE

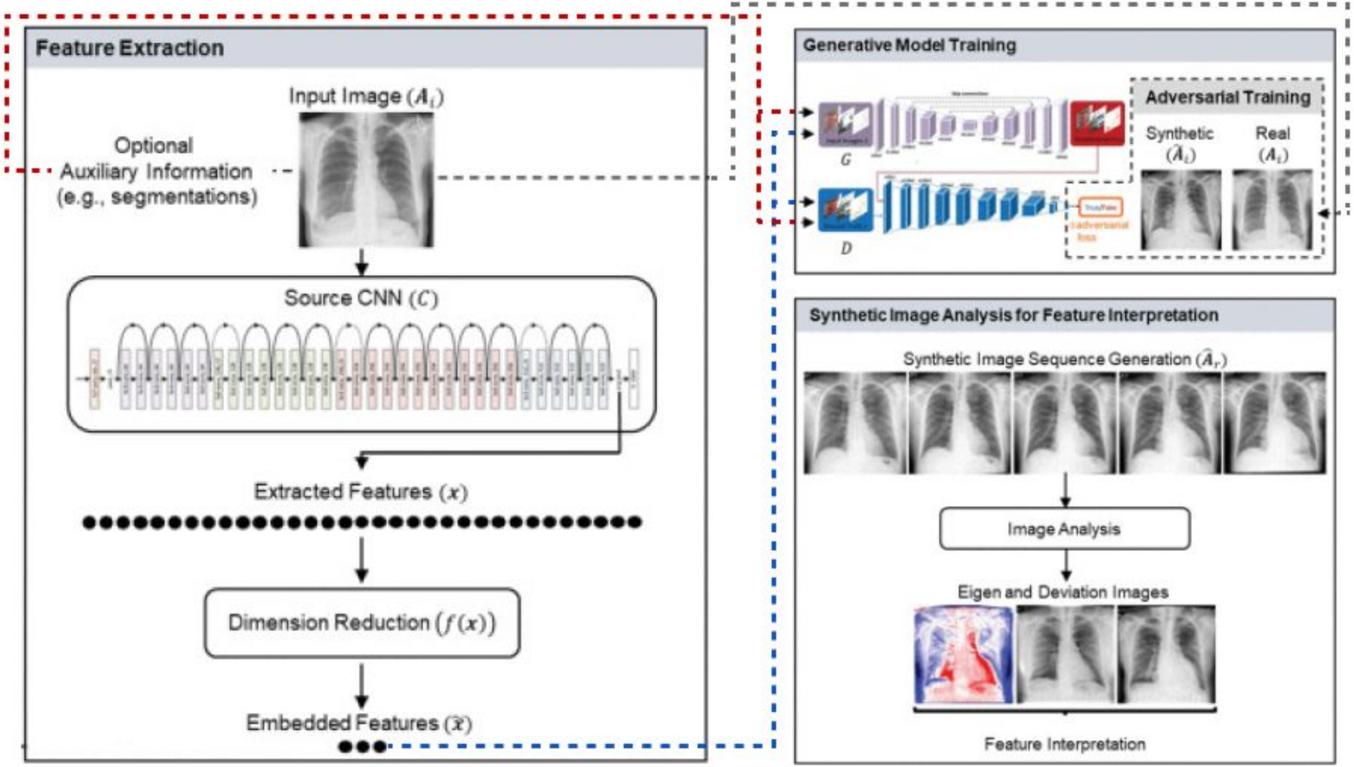


Three main steps:

- Feature Extraction (CNN),
- Generative Model Training (FIGAN), and
- Generated Image Analysis (to gain insights into the CNN's decision-making process and identify which features are most important for classification).

Image Source: [9]

ARCHITECTURE



Three main steps:

- Feature Extraction (CNN),
- Generative Model Training (FIGAN), and
- Generated Image Analysis (to gain insights into the CNN's decision-making process and identify which features are most important for classification).

Image Source: [9]

Results of the research paper:

- Proposed FIGAN framework provides a **new level of understanding and explainability** in medical imaging classification tasks.

- Using FIGAN on previously developed CNNs (Gradient-Based Saliency Maps, Attention Networks) show that the resulting feature interpretations can **clarify ambiguities within attention areas** highlighted.

- The authors also demonstrate that **FIGAN can provide consistent and interpretable synthetic images across a range of medical imaging datasets.**

Limitations of the research paper:

- Proposed **FIGAN framework requires a pre-trained CNN**, which may not always be available or may not be applicable to all medical imaging classification tasks.

- Synthetic images generated by FIGAN may **not always be clinically relevant or useful for diagnosis.**

- Further experiments by authors show that **FIGAN's ability is affected by auxiliary segmentations, training sample size, and image resolution**.

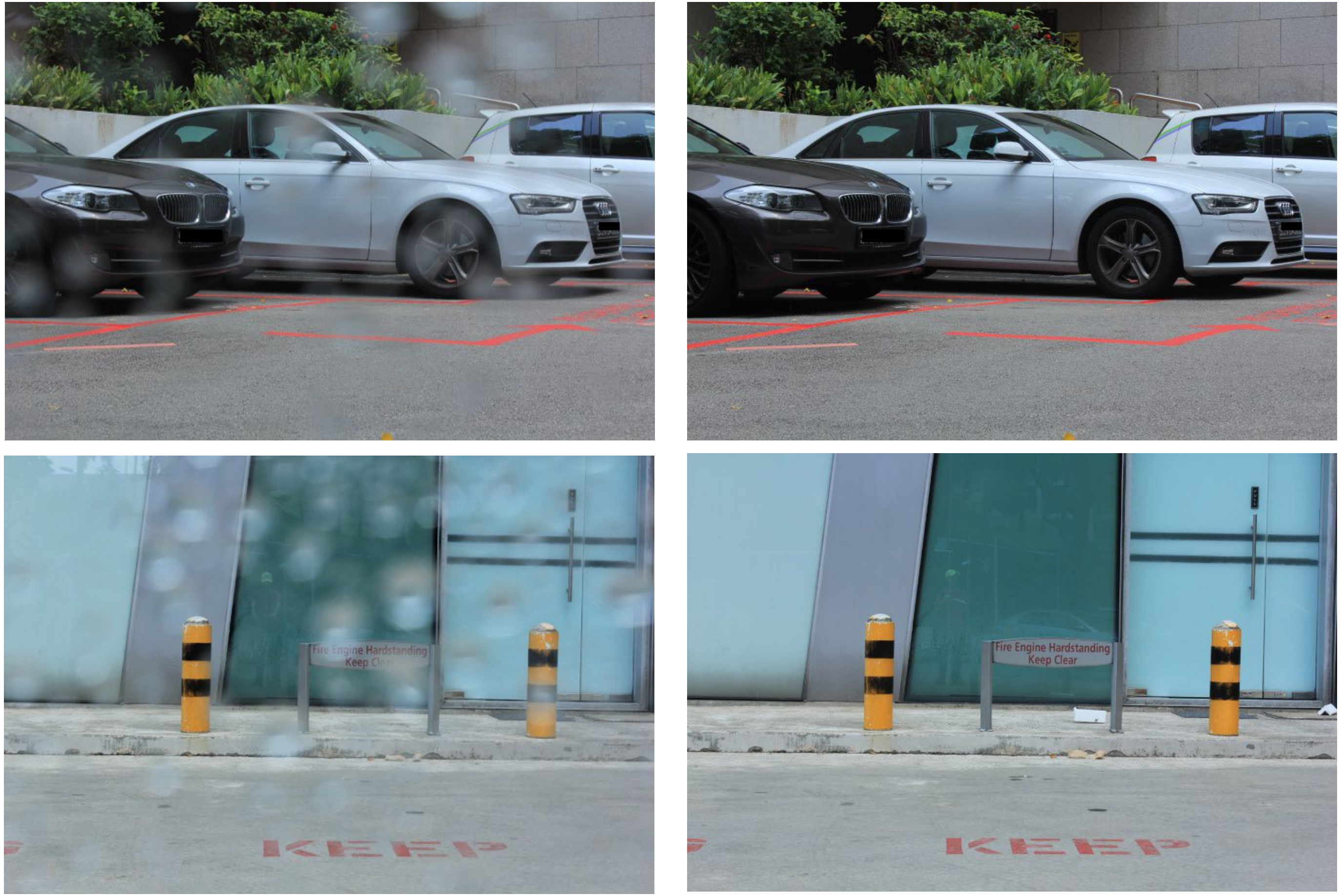
- Additionally, the authors notes the **interpretability of CNNs as a complex and ongoing research area**.
- **FIGAN is just one approach** to improving the **explainability of CNNs in medical imaging** classification tasks.

THE STORY ENDING

Young Photographer got this result after using **GAN** for his problem

BEFORE

AFTER



BEFORE **AFTER** (Source: [8] See this)

From a simple idea, GANs has created a storm.

Artbreeder

DeepFace

FacePlay

MidJourney

Prisma

DALLE

Backend of these apps are powered by various architectures of GANs.

References

[1] Ian Goodfellow, “***NIPS 2016 Tutorial: Generative Adversarial Networks***”, DOI: arXiv.1701.00160, Dec 2016.

[2] Ian J. Goodfellow, Jean Pouget Abadie, Mehdi Mirza, Bing Xu, David Warde Farley, Sherjil Ozair, Aaron Courville,Yoshua Bengio, “***Generative Adversarial Nets***”, DOI: arXiv:1406.2661, June 2014.

[3] Zhaoqing Pan, Weijie Yu, Xiaokai Yi, Asifullah Khan, Feng Yuan, Yuhui Zheng, “***Recent Progress on Generative Adversarial Networks (GANs): A Survey***”, DOI: 10.1109/ACCESS.2019.2905015, March 2019.

[4] Radford, Alec, Metz, Luke, Soumith Chintala, "***Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks***", DOI: ICLR.S2CID 11758569, November 2015

[5] Han Zhang, Ian Goodfellow, Dimitris Metaxas, Augustus Odena, "***Self-Attention Generative Adversarial Networks***", DOI: PMLR: 7354–7363, June 2019.

[6] Anders Boesen Lindbo Larsen, Søren Kaae Sønderby, Hugo Larochelle, Ole Winther, "***Autoencoding beyond pixels using a learned similarity metric***". DOI: arXiv:1512.09300, February 2016.

[7] Yifan Jiang, Shiyu Chang, Zhangyang Wang, "***TransGAN: Two Pure Transformers Can Make One Strong GAN, and That Can Scale Up***", DOI: arXiv:2102.07074, December 2021.

[8] Egor Zakharov, Aliaksandra Shysheya, Egor Burkov, Victor Lempitsky, "***Few-Shot Adversarial Learning of Realistic Neural Talking Head Models***", DOI: arXiv:1905.08233, May 2019.

[9] Kyle A. Hasenstab, Justin Huynh, Samira Masoudi, Guilherme M. Cunha, Michael Pazzani, Albert Hsiao, "***Feature Interpretation Using Generative Adversarial Networks (FIGAN): A Framework for Visualizing a CNN's Learned Features",*** DOI: 10.1109/ACCESS.2023.3236575, January 2023.

[10] Mehdi Mirza, Simon Osindero, "***Conditional Generative Adversarial Nets (CGAN)***", DOI: arXiv:1411.1784v1, November 2014.

- For **more resources related to GANs**, see here

Thank You

Questions?