Online workshop on Generative AI Hands-on session: Generative Adversarial Networks & Its Variants

Praveen K Chandaliya, Phd

Sardar Vallabhbhai National Institute of Technology, Surat
Department of Artificial Intelligence
pkc@aid.svnit.ac.in

March 9, 2025

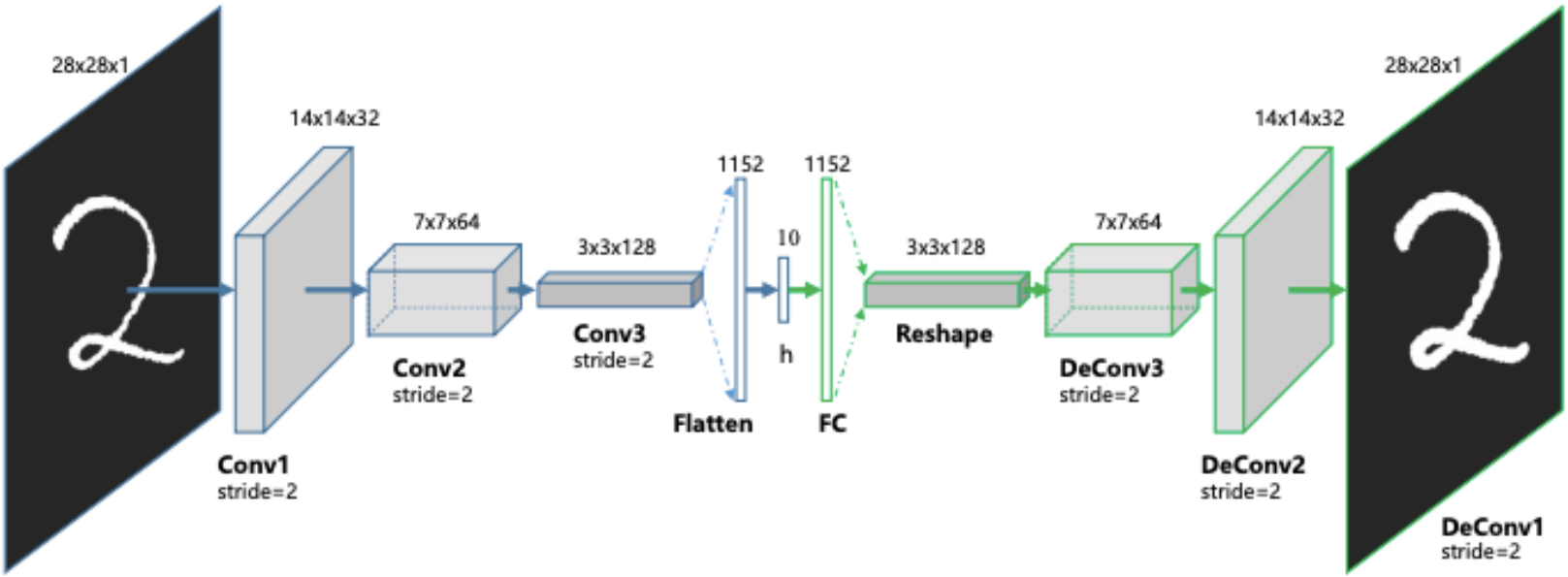
Outline

Prerequisites for GAN

- Dataloader in PyTorch
- Convolution and Deconvloution Operation in Pytorch
- Sequential Convolution block
- MSE and Binary Cross entropy loss functions in Pytorch

STTP_Prerequisites.ipynb

Convolution Autoencoder

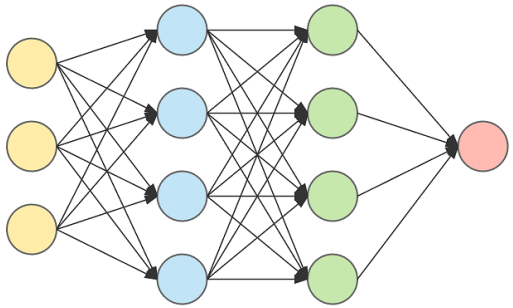


STTP_CNN_AE.ipynb

Generative Adversarial Network

Definition

- Generative
- Adversarial
- Networks



Generative Adversarial Network Architecture

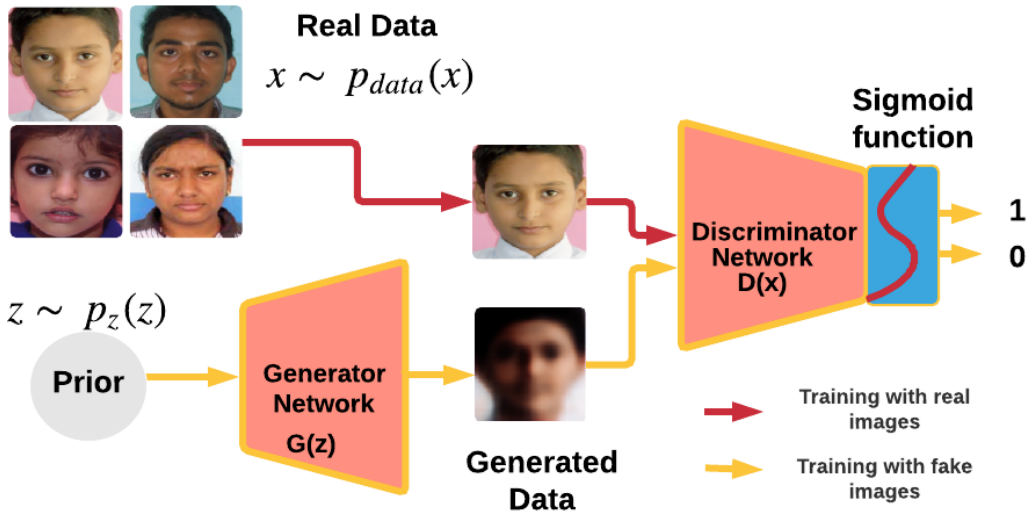


Figure 1: GAN Architecture

- Discriminator (D) distinguishing between real images and generated images and is responsible for classifying images as real(1) or fake (generated)(0).
- Generator (G) generating the images that fool the discriminator.

Loss Function

Binary Cross Entropy:

$$\mathcal{L}(\hat{y}, y) = y \log(\hat{y}) + (1 - y) \log(1 - \hat{y}) \tag{1}$$

Discriminator on real (D) :
$y =$ original, $\hat{y} =$ generated image; The label for the data coming from $P_{data}(x)$ is $y = 1$ and $\hat{y} = D(x)$

$$\mathcal{L}(D(x), 1) = \log(D(x)) \tag{2}$$

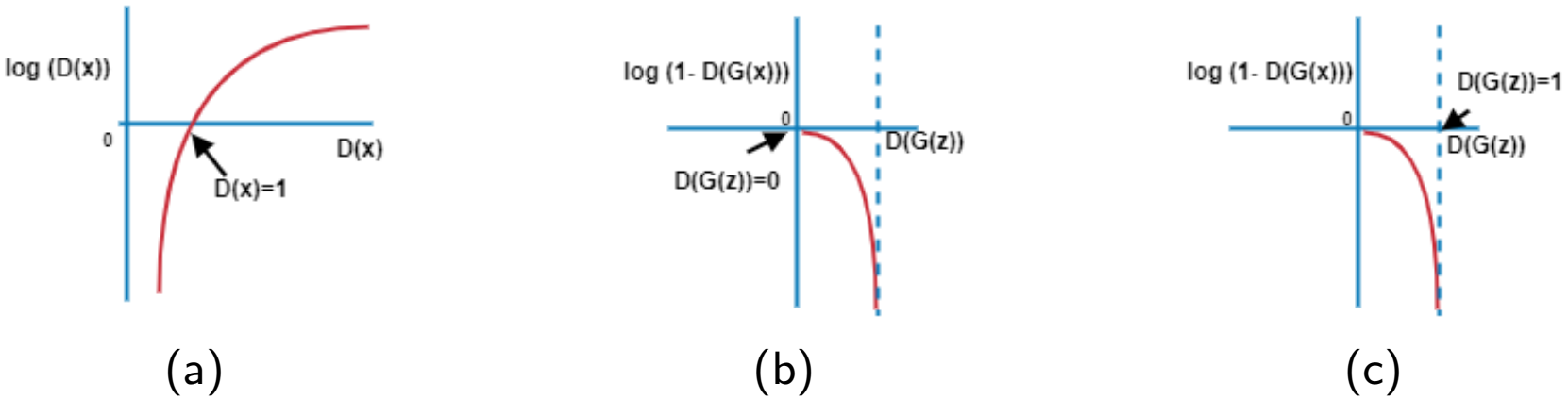
Discriminator on fake (D):
Data coming from generator the label is $y = 0$ and $\hat{y} = D(G(z))$

$$\mathcal{L}(D(G(z)), 0) = \log(1 - D(G(z))) \tag{3}$$

Continue

Objective of the Discriminator is to correctly classify fake vs the real dataset. For this Equation 2 and 3 should be maximized.

$$\max_{D} \quad \log(D(x)) + \log(1 - D(G(z))) \tag{4}$$



(a) (b) (c)

Figure 2: (a) and (b) Discriminator D Loss and (c) Generator G loss.

Objective of Generator to fool the discriminator. $D(G(z)) = 1$ then Generator fool the Discrimintor by producing the probability as 1.

$$\min_{G} \quad \log(1 - D(G(z))) \tag{5}$$

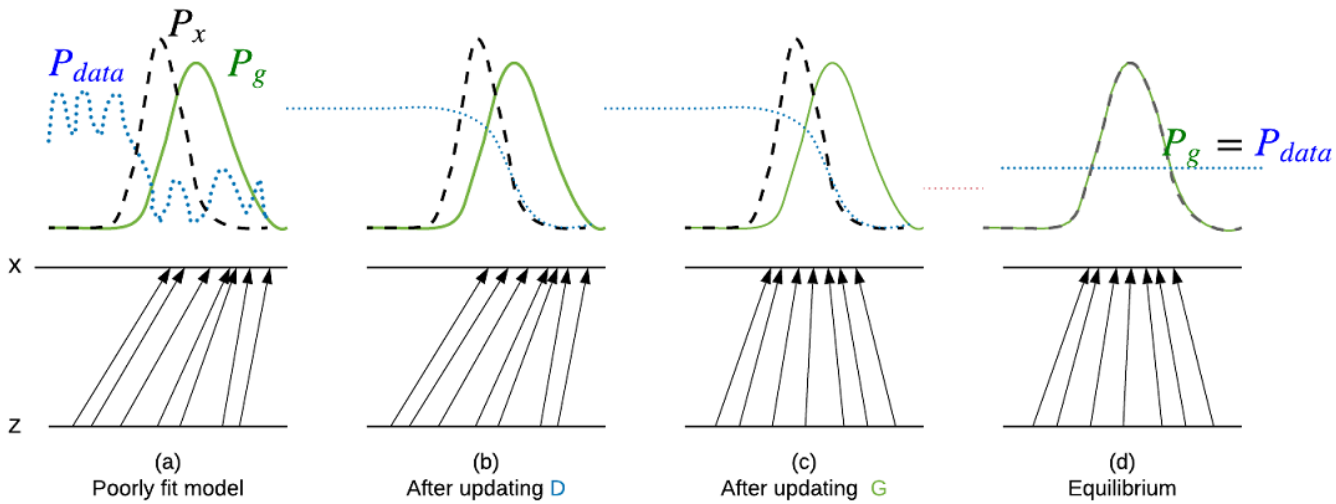
Mathematical Formulation

Training GAN is a two-player minimax game problem

- Discriminator D tries to maximize it classification accuracy.
- Generator G tries to minimize the discriminator classification accuracy.

$$\min_G \max_D V(D,G) = \log\left[D(x)\right] + \log\left[1 - D\left(G\left(z\right)\right)\right] \tag{6}$$

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim Pdata(x)} \log\left[D(x)\right] + \mathbb{E}_{z\sim Pz(z)} \log\left[1 - D\left(G\left(z\right)\right)\right] \tag{7}$$



Training GAN

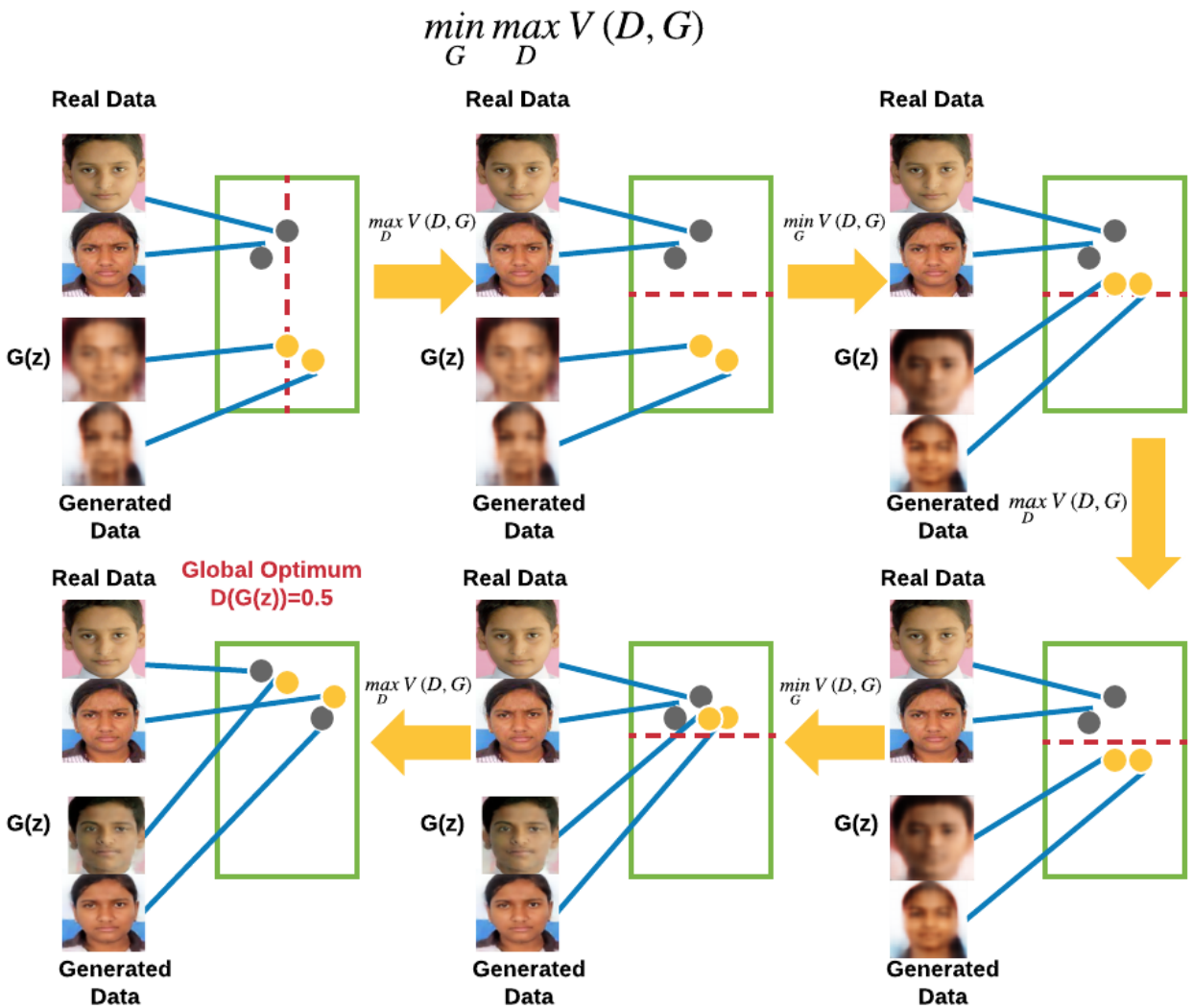


Figure 3: Training Steps in GAN

Algorithm

Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k, is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

for number of training iterations **do**

  for k steps **do**

  - Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
  - Sample minibatch of m examples $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\boldsymbol{x})$.
  - Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right) \right].$$

  end for

  - Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
  - Update the generator by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right).$$

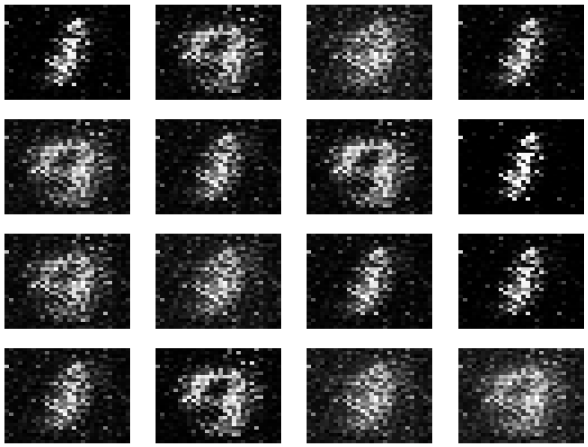
end for

The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

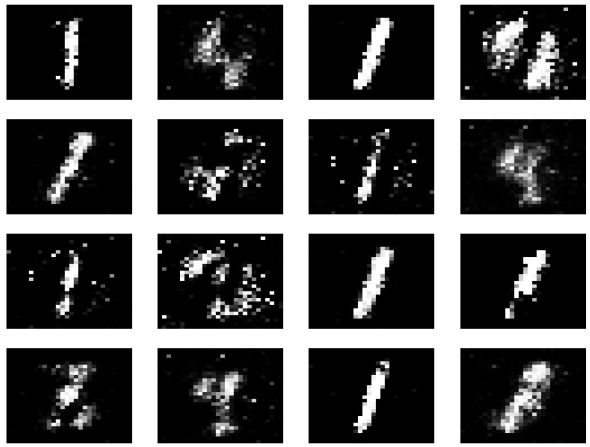
Figure 4: GAN Objective

Generative Adversarial Network implementation on colab (STTP_GAN.ipynb).

GAN Implementation Result: MNIST



(a) Epoch 5



(b) Epoch 20

Figure 5: GAN implementation result on MNIST

Challenges Faced by GANs

Training Instability

1. Non-convergence: GANs often fail to converge due to the adversarial nature of their training, as the generator and discriminator compete with each other.
2. Vanishing Gradients: When the discriminator becomes too good, the generator receives almost no gradient to improve itself.
3. Mode Collapse: The generator produces limited diversity, repeatedly generating similar outputs instead of a wide variety of data.
4. Sensitive Hyperparameters: Training requires careful tuning of hyperparameters like learning rates, batch sizes, and network architectures

DCGAN: MNIST

Generator

1. Hidden layers: Four 4x4 strided convolutional layers (1024, 512, 256, and 128 kernels, respectively) with ReLU.
2. Output layer: 4x4 strided convolutional layer (4096 nodes = 64x64 size image) with Tanh.
3. Batch normalization is used except for output layer.

Discriminator

1. Hidden layers: Four 4x4 convolutional layers (128, 256, 512, and 1024 kernels, respectively) with Leaky ReLU.
2. Output layer: 4x4 convolutional layer (1 node) with Sigmoid.
3. Batch normalization is used except for 1st hidden layer and output layer.

Deep Convolution Generative Adversarial Network

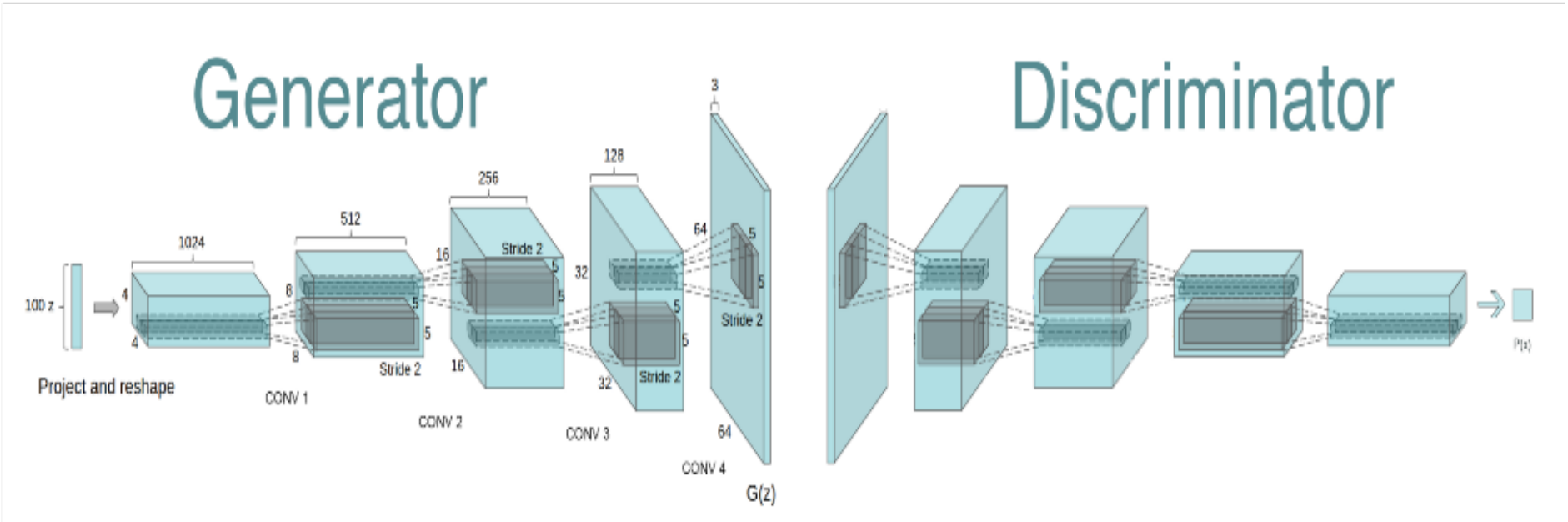
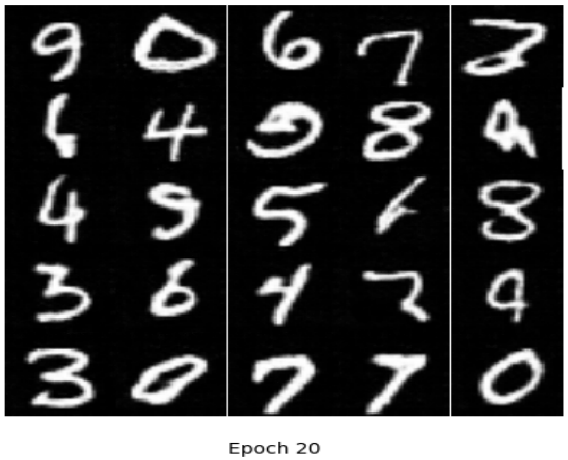


Figure 6: Deep Convolution Generative Adversarial Network [1]

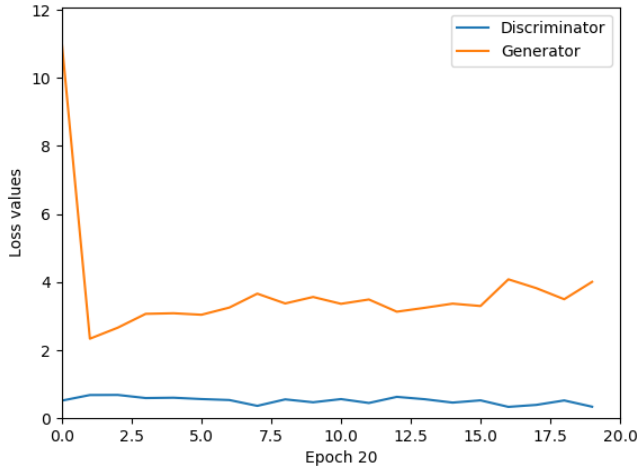
1. Replace any pooling layers with strided convolutions (discriminator) and fractional-strided convolutions (generator).
2. Batchnorm in both the generator and the discriminator.
3. ReLU activation in generator for all layers except for the output, which uses Tanh.
4. LeakyReLU activation in the discriminator for all layers.

[1] Soumith et al. "Unsupervised representation learning with deep convolutional generative adversarial networks.

DCGAN Implementation Result: MNIST



(a) MNIST Data



(b) Training Loss

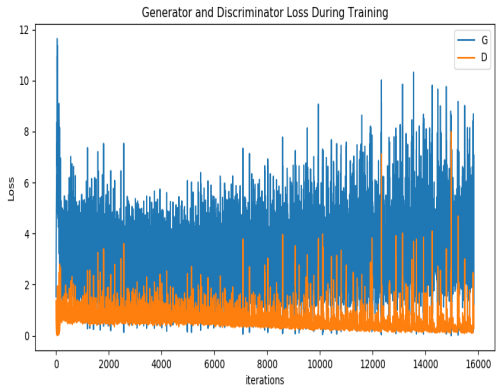
Figure 7: DCGAN results on MNIST dataset (STTP_DCGAN_MNIST.ipynb)

DCGAN Implementation Result: Celeb

(a) Celeb Data

(b) epoch-1

(c) epoch-15



(d) Training Loss

Figure 8: DCGAN results on Celeb dataset

Variants of GAN structure

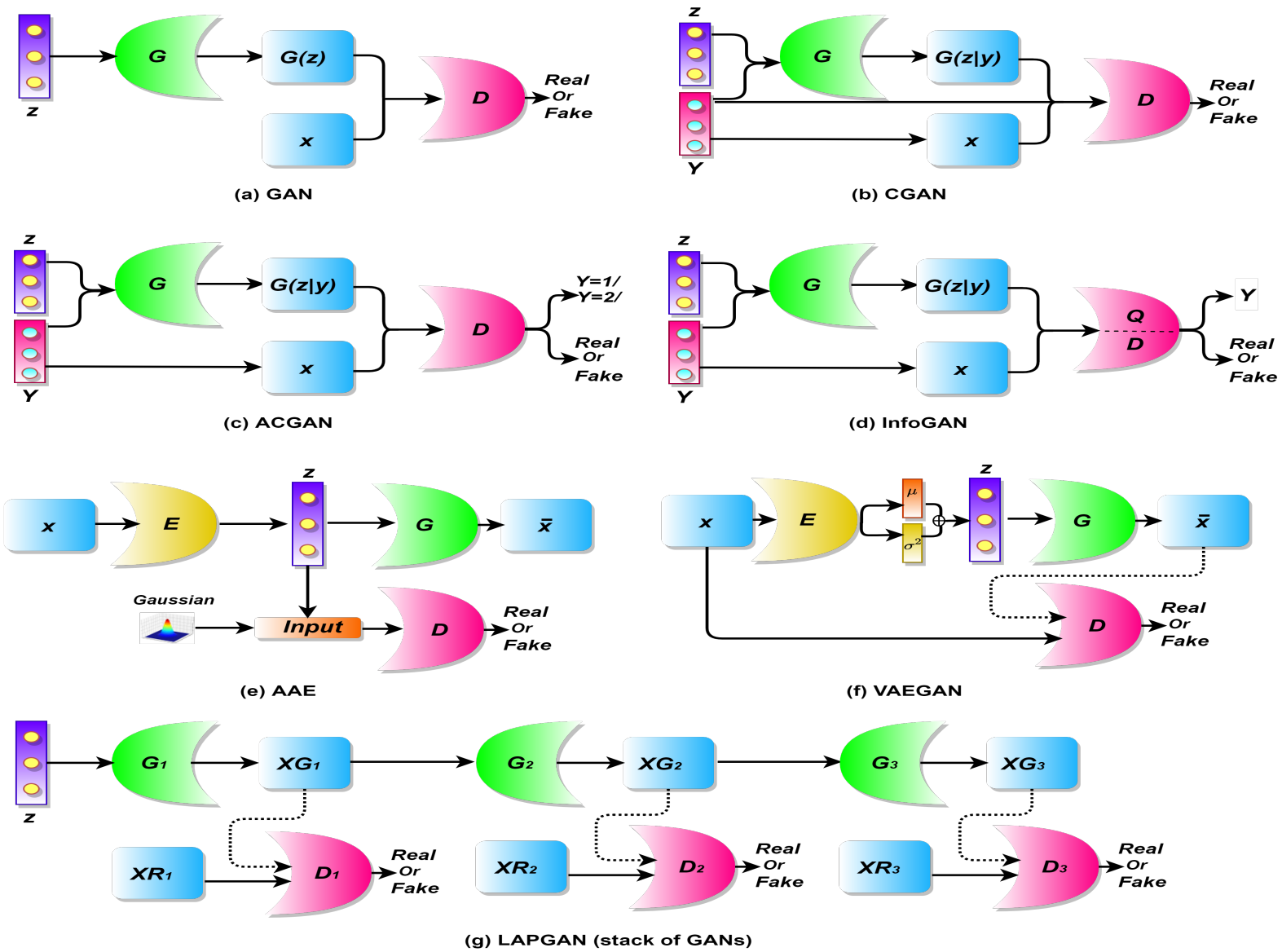


Figure 9: Variants of GAN structure

GAN Variants

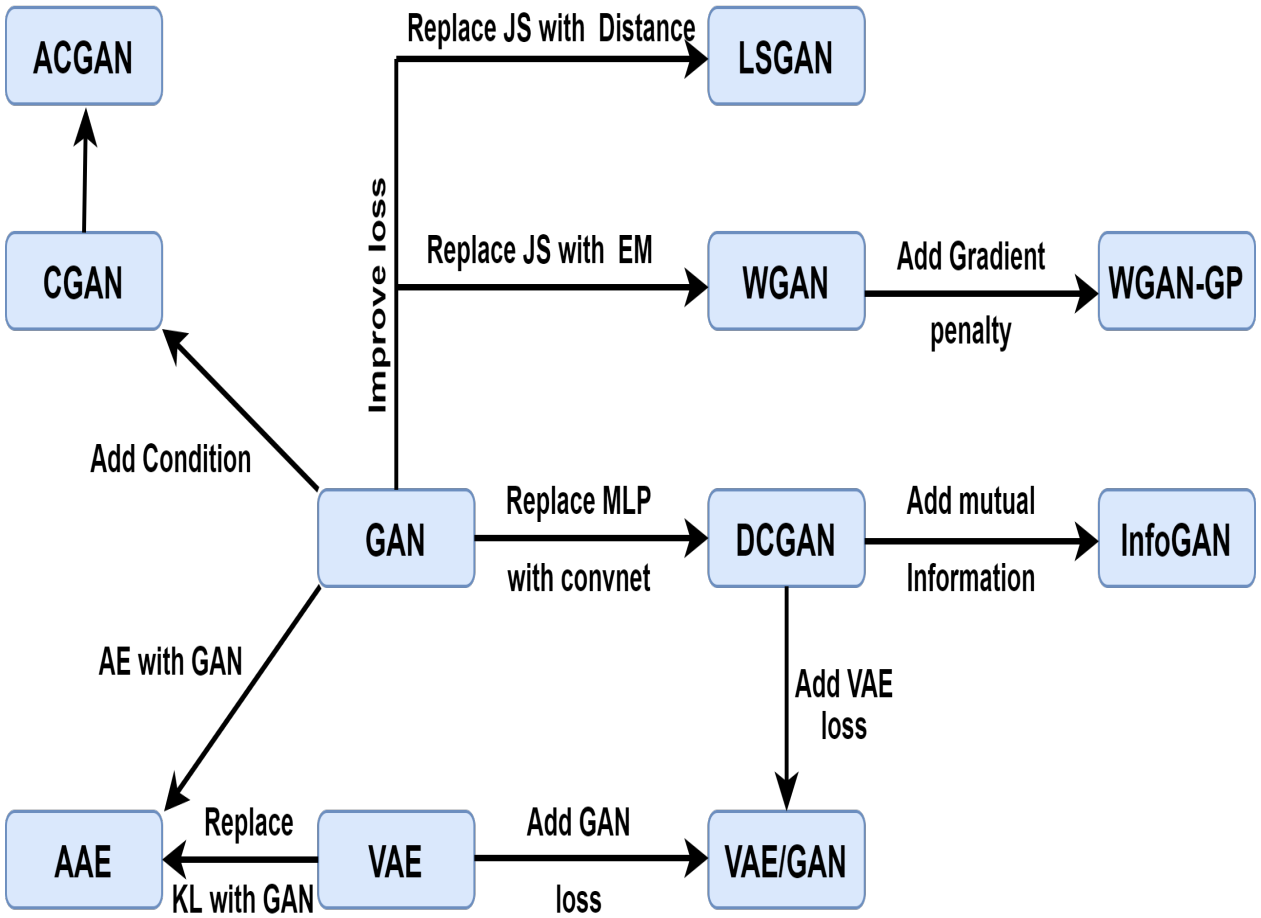


Figure 10: Variants of GAN structure

Conditional Generative Adversarial Network (cGAN)

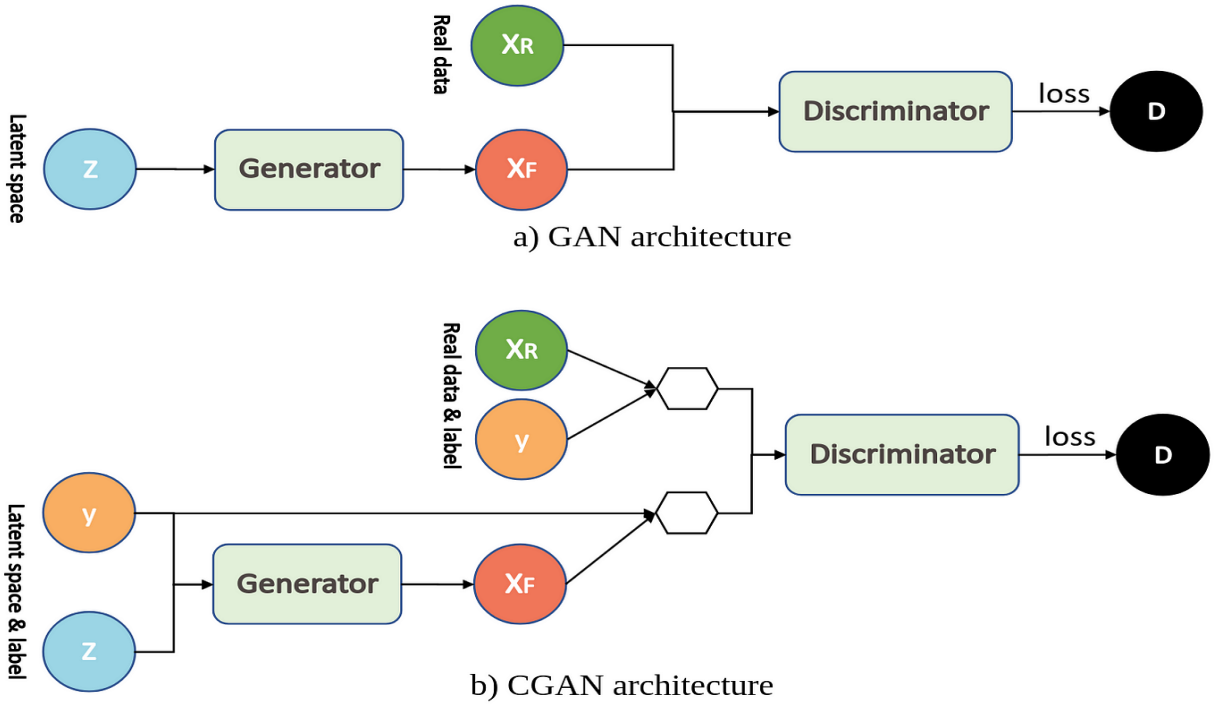


Figure 11: Conditional generative adversarial network

Conditional Generative Adversarial Network implementation on colab.

cGAN Result

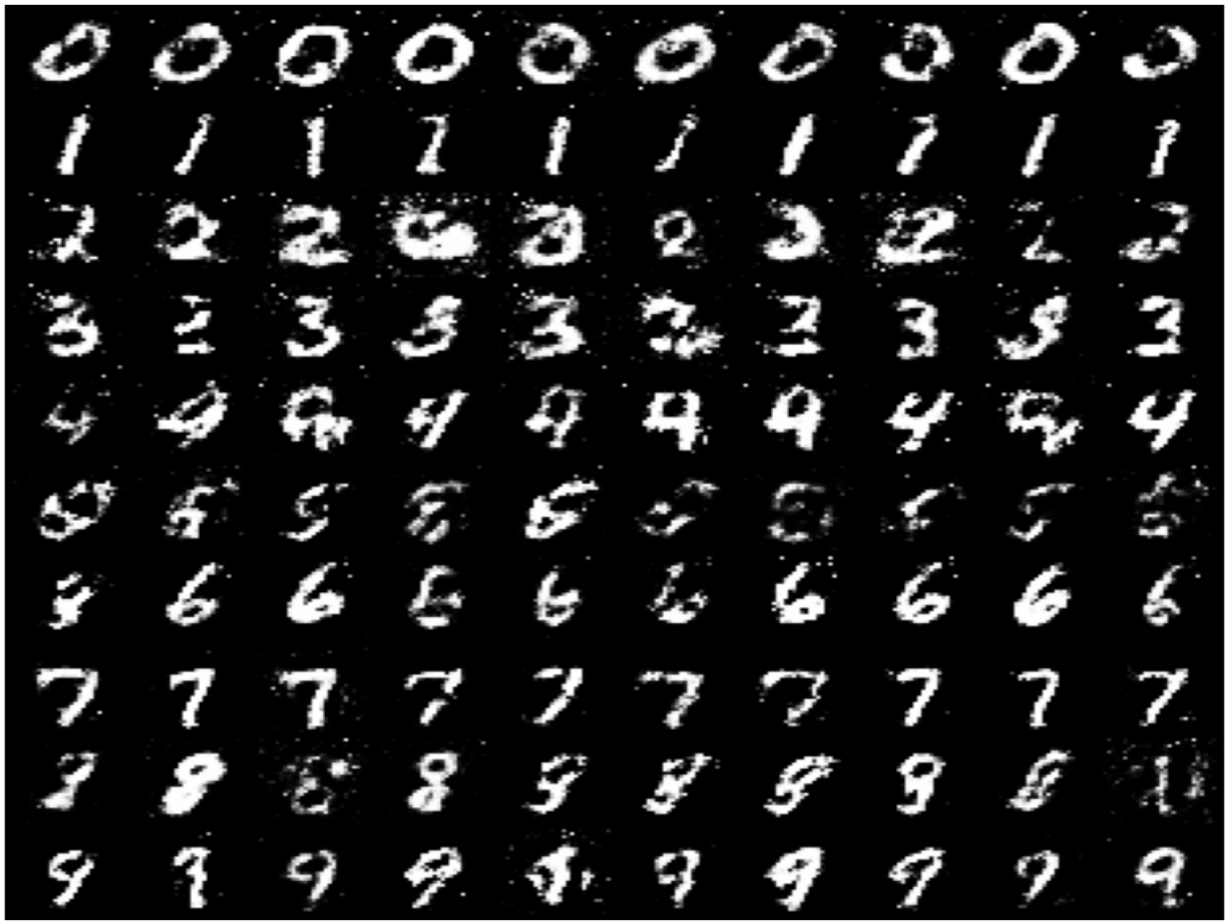


Figure 12: cGAN

Least squares generative adversarial networks (LSGAN)

Least-square GAN

- For discriminator

D has linear output

$$\min_D \frac{1}{2} E_{x \sim P_{data}}[(D(x) - b)^2] + \frac{1}{2} E_{x \sim P_G}[(D(x) - a)^2]$$

1 0

- For Generator

$$\min_D \frac{1}{2} E_{z \sim P_{data}}\left[\left(D\left(G(z)\right) - c\right)^2\right]$$

1

LSGAN:STTP_LSGAN.ipynb

Face Datasets

Table 1: Face datasets.

Dataset	Subjects	Images	Images Subject	Range	Avg.Age	Public
FGNET	82	1,002	6-18	0-69	16	Yes
Adience	2,958	26,580	NA	0-60	NA	Yes
Morph	13,000	55,134	2-53 (avg. 4.2)	16-77	42	Yes
CACD	2,000	163,446	22-139 (avg. 81.7)	16-62	31	Yes
IMDBWiki	20,284	523,051	avg. 25.79	0-100	38	Yes
UTKFace	NA	23,709	NA	0-116	33	Yes
AgeDB	568	16,488	avg. 29	1-101	50.3	Yes
AGFW-v2	36,325	27,688	-	10-64	–	Yes
ITWCC	754	7,990	3-37 (avg. 10.7)	0-32	13	Yes
CLF	919	3,682	2-6 (avg. 4.0)	2-18	8	No

Convolution Autoencoder with Face Dataset

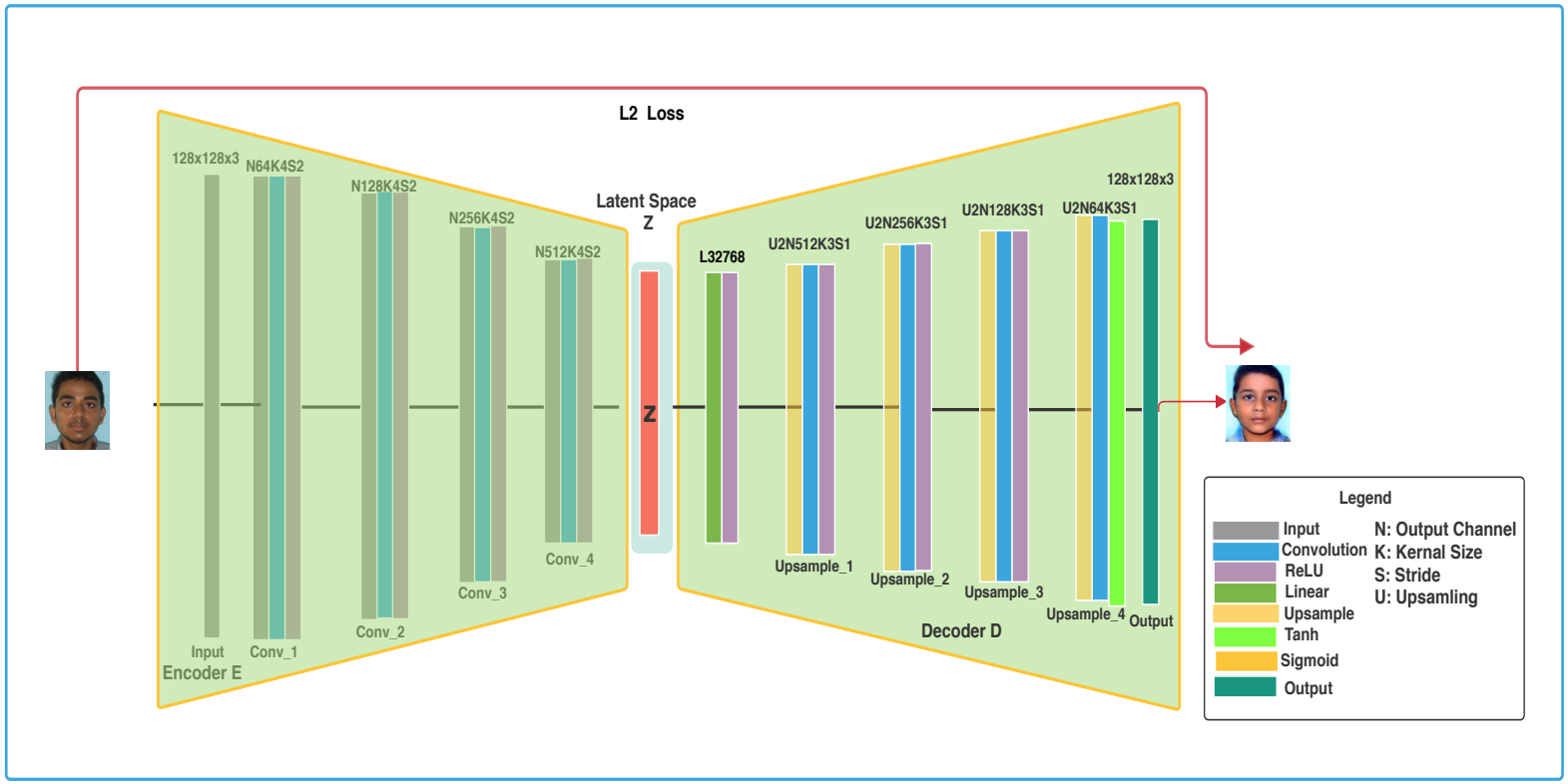
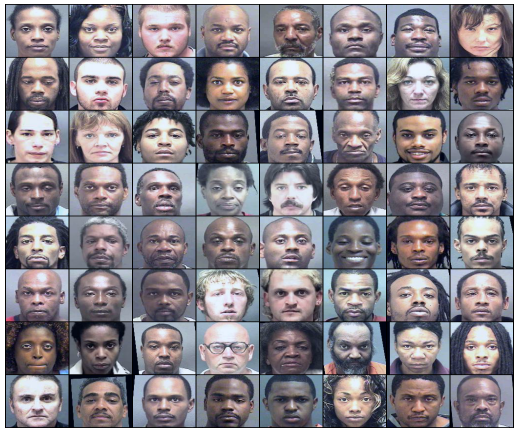


Figure 13: Convolution Autoencoder implementation on colab **STTP_CAE_Face.ipyb** [2]

[2] Praveen Kumar Chandaliya, Neeta Nain, Conditional Perceptual Adversarial Variational Autoencoder,ICB2019,IEEE.

Convolution Autoencoder Implementation Result



(a) Morph Data

(b) epoch-1

(c) epoch-15

(d) epoch-30

Figure 14: Convolution Autoencoder Implementation Results

Publication I

- Praveen Kumar Chandaliya, Preyas Garg and Neeta Nain, "Retrieval of Facial Images Re-rendered with Natural Aging Effect using Child Facial Image and Age," (SITIS), Las Palmas de Gran Canaria, Spain, 2018, pp. 457-463
- Praveen Kumar Chandaliya, Neeta Nain "Conditional Perceptual Adversarial Variational Autoencoder for Age Progression and Regression on Children Face" ICB 2019.**B Level**,
- Praveen Kumar Chadnaliya, Vardhman, Mayank Harjani, Neeta Nain, "SCDAE: Ethnicity and Gender Alteration on CLF and UTKFace Dataset" CVIP 2019
- Praveen Kumar Chandaliya, Aditya Sinha, Neeta Nain "ChildFace: Gender Aware Child Face Aging" BIOSIG 2020 **B Level**.

References I


Neeta Nain Praveen Kumar Chandaliya.
Childgan: Face aging and rejuvenation to find missing children.
In *Pattern Recognition*, volume 129, page 108761, 2022.


P. Gallinari, Yann Lecun, S. Thiria, and F. Fogelman Soulie.
Memoires associatives distribuees: Une comparaison (distributed associative memories: A comparison).
In *Proceedings of COGNITIVA 87, Paris, La Villette, May 1987*.
Cesta-Afcet, 1987.


Diederik P. Kingma and Max Welling.
Auto-encoding variational bayes.
CoRR, abs/1312.6114, 2013.

References II


Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio.
Generative adversarial nets.
In *NIPS*, 2014.


Mehdi Mirza and Simon Osindero.
Conditional generative adversarial nets.
CoRR, abs/1411.1784, 2014.


Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, and Ian J. Goodfellow.
Adversarial autoencoders.
CoRR, abs/1511.05644, 2015.

References III


Alec Radford, Luke Metz, and Soumith Chintala.
Unsupervised representation learning with deep convolutional generative adversarial networks.
CoRR, abs/1511.06434, 2015.


Debayan Deb, Neeta Nain, and Anil K. Jain.
Longitudinal study of child face recognition.
CoRR, abs/1711.03990, 2017.