Generative Adversarial Networks (GAN)

Dr. Sunil Kumar Vuppala

Senior Member, IEEE and Fellow, IETE

www.linkedin.com/in/sunilvuppala/

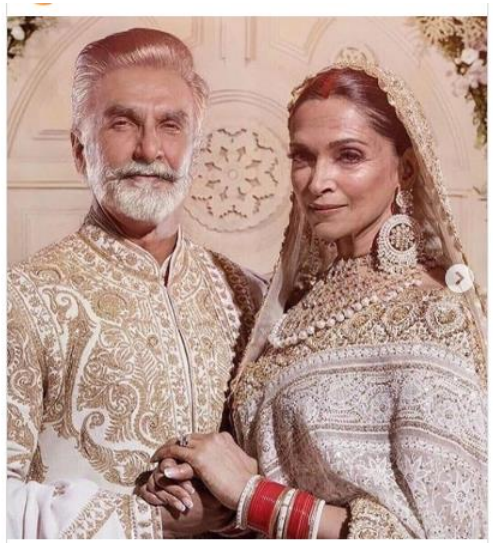
Agenda

- Introduction to Generative Adversarial Networks (GAN)
- Generative models, Discriminative models
- How GANs work? GAN Lab
- Different types of GANs
  - DCGAN
  - CGAN
  - ACGAN
- Detailed use case: Using GANs to generate synthetic data
  - Mode collapse problem
  - Steps to stabilize the training
  - Results
- Other GAN applications
- Summary
- References

Pre-requisites

- Basics of neural networks
- Forward pass
- Back propagation
- Basic Python programming and Keras+TF / TF
- Loss functions
- Optimization techniques
- Basics of probability distributions
- Convolution operation

Face App of celebrities



Smile, older, younger, style changing, Gender swapping kind of features
What's new? GAN

Introduction

- **Generative** Model
  - Aim to model how data is generated
  - Generation of synthetic data points
  - Model class conditional PDFs and prior probabilities
  - Knowledge about the data **distribution** – range of possibilities
  - **Probability of x and y (**Baye's theorem)
  - Very expensive step
  - Needs lot of data
  - Popular models – Gaussian, Naïve Bayes, HMM, Multinomial

- **Discriminative** model
  - Aim **to learn P(c|x)** by using probabilistic approaches
  - Estimate posterior probabilities
  - Easy to model
  - To classify but not to generate data
  - Popular models – Logistic regression, SVM, Conditional Random Fields and Traditional neural network

Generative Models

So Far we are trying to solve these problems?

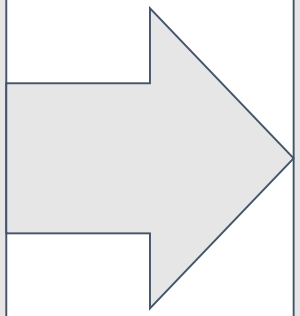
1. Predict the Price of a Car
2. Classify Dogs and Cats
3. Predict whether it will Rain or Not?
4. Whose Face is it?

These all are Discriminative Models.

Generative Models

1. Predict the Price of a Car
2. Classify Dogs and Cats
3. Predict whether it will Rain or Not?
4. Whose Face is it?

Discriminative Model

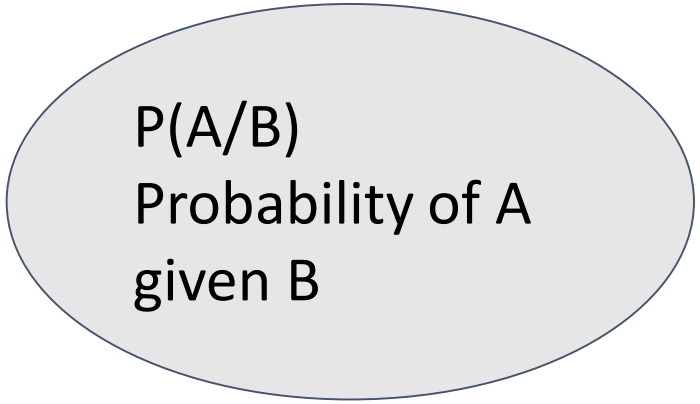


- What is a Car?
- What is a Dog or a Cat?
- What is Rain?
- What is Face?

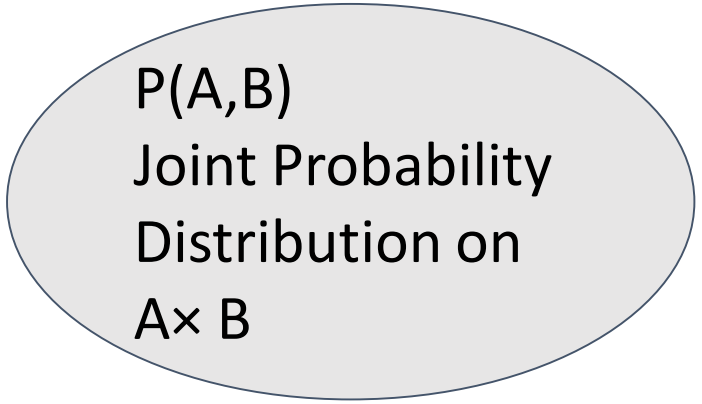
Generative Model

Discriminative Models vs Generative Model in Math

What are we trying to solve?



Discriminative Models



Generative Model

KL and JS Divergence

- Kullback-Leibler Divergence: It is **relative entropy** and is a measure of how one probability distribution is different from a second, reference probability distribution.
- It is a distribution-wise ***asymmetric*** measure

- KL Divergence from Q to P

$$D_{KL}(P||Q) = \sum_{x=1}^{N} P(x) \log \frac{P(x)}{Q(x)}$$

- Jensen-Shannon divergence - **symmetric**

$$D_{JS}(p||q) = \frac{1}{2} D_{KL}(p||\frac{p+q}{2}) + \frac{1}{2} D_{KL}(q||\frac{p+q}{2})$$

https://en.wikipedia.org/wiki/Kullback%E2%80%93Leibler_divergence

Who proposed GANs?

IAN Goodfellow in 2014

From Stanford and University of Montreal

PhD Advisor: Yoshua Bengio, Turing Award winner

Co-authored Deep Learning Text book

Worked in Google

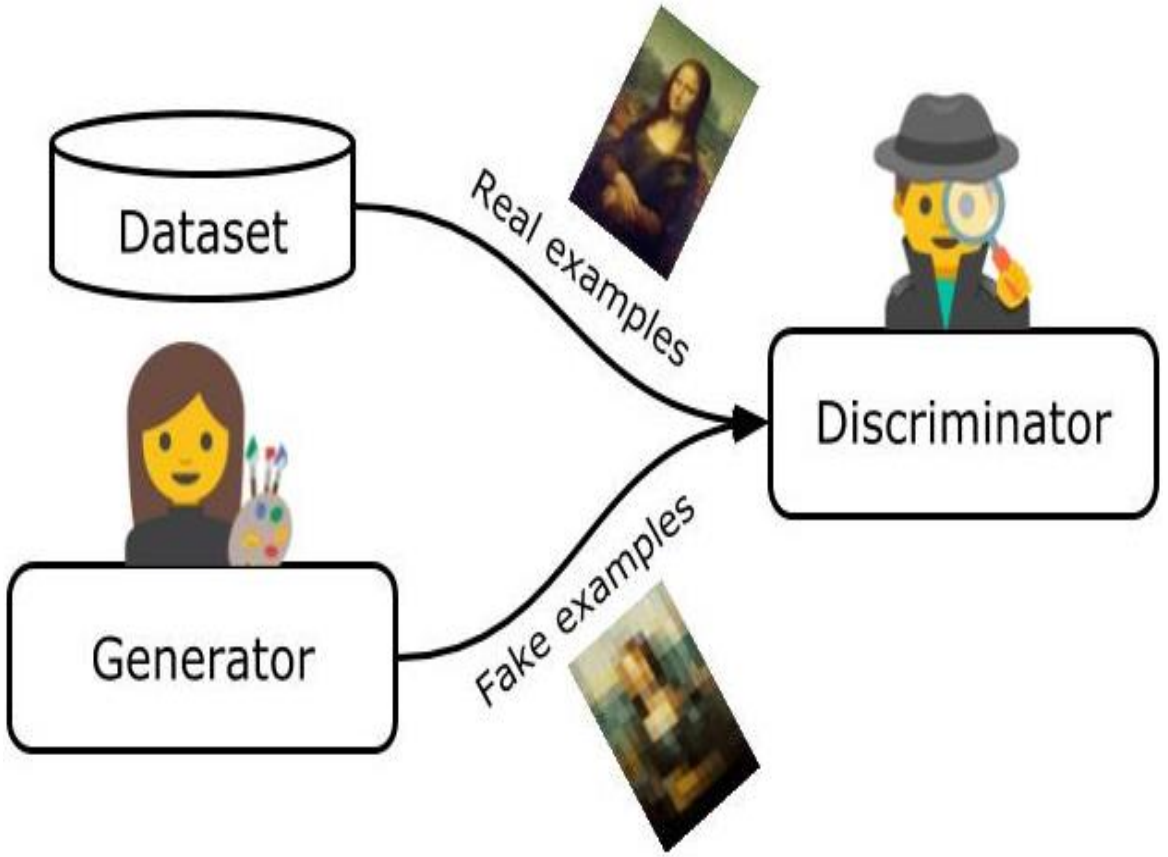
Contributed to OpenAI

Now with Apple

www.iangoodfellow.com

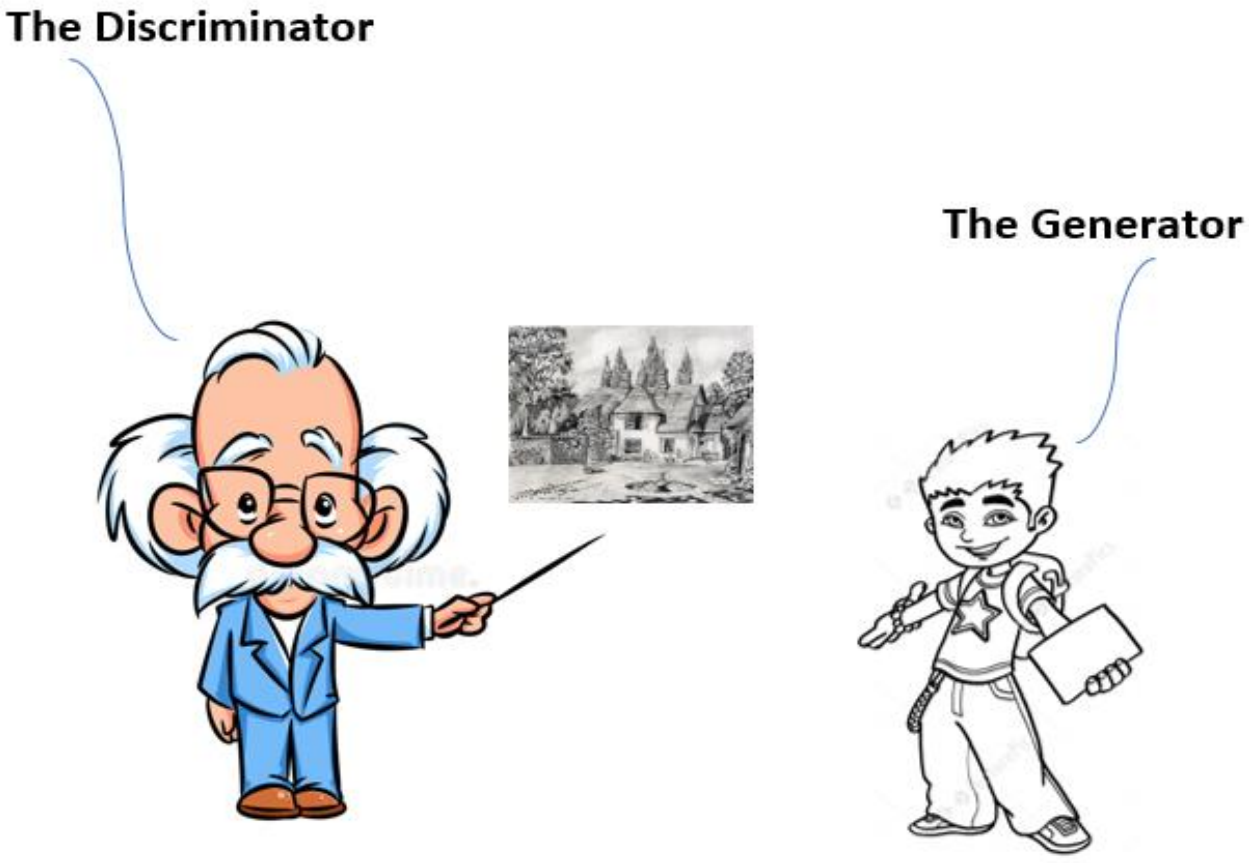
What are GANs?

- An art forger (the “Generator”) trying to fool an art expert (the “Discriminator”).
- The more convincing the fake paintings the forger makes, the better the art expert needs to be at determining their authenticity and vice versa.



What are GANs?

A teacher keeps on giving feed back until student creates a perfect picture so that teacher Can not differentiate between th Original picture and the one Created by student.



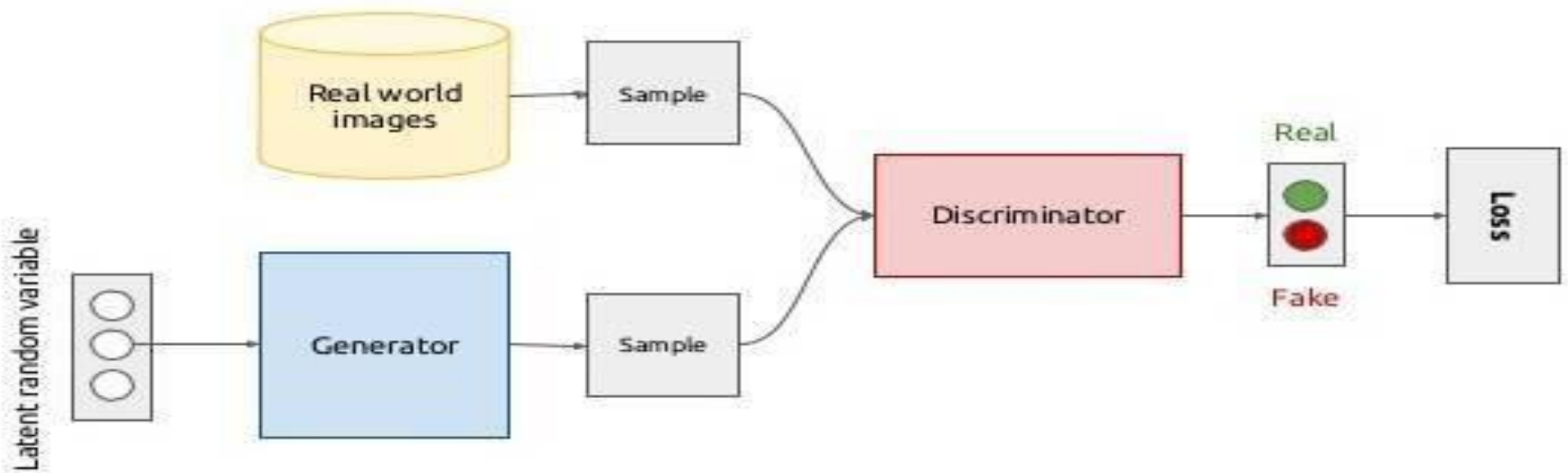
GANs

The word “generative” indicates the overall purpose of the model: creating new data.The data that a GAN will learn to generate depends on the choice of the training set—for example, if we want a GAN to paint like Leonardo da Vinci, we would use a training dataset of Leonardo’s artwork.

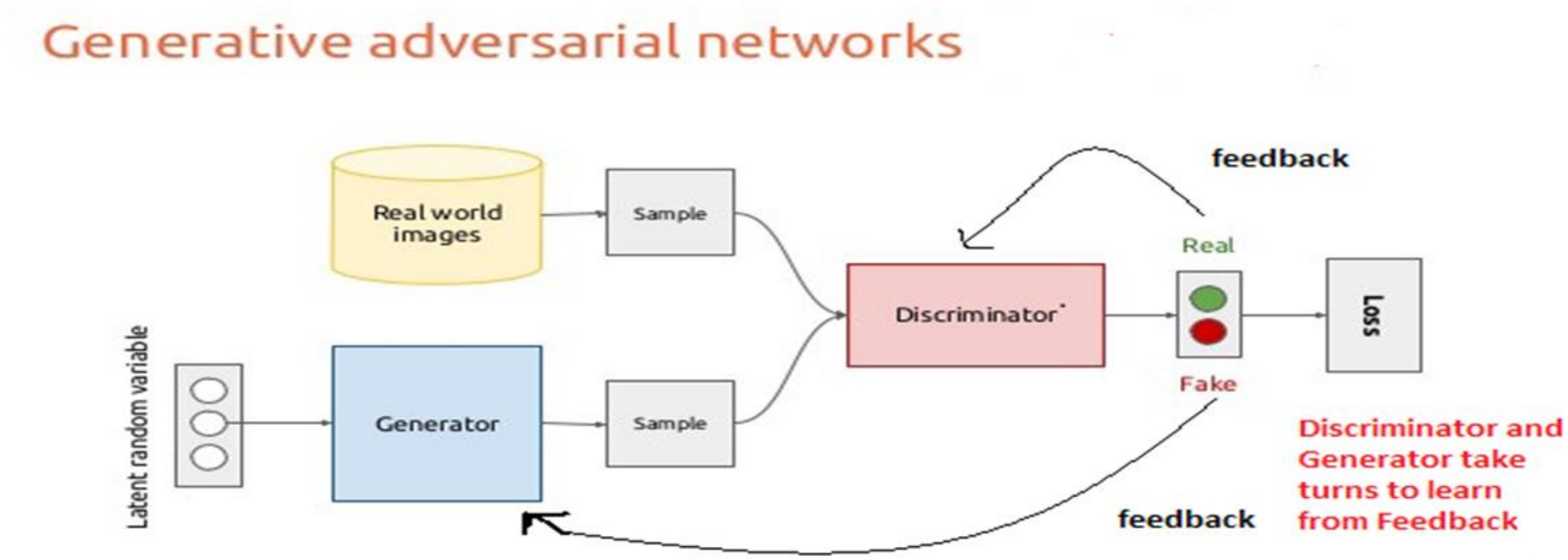
The term “adversarial” points to the game-like, competitive dynamic between two algorithms that constitute the GAN framework.

The Generator and the Discriminator. The Generator’s goal is to create examples that are indistinguishable from the real data in the training set.

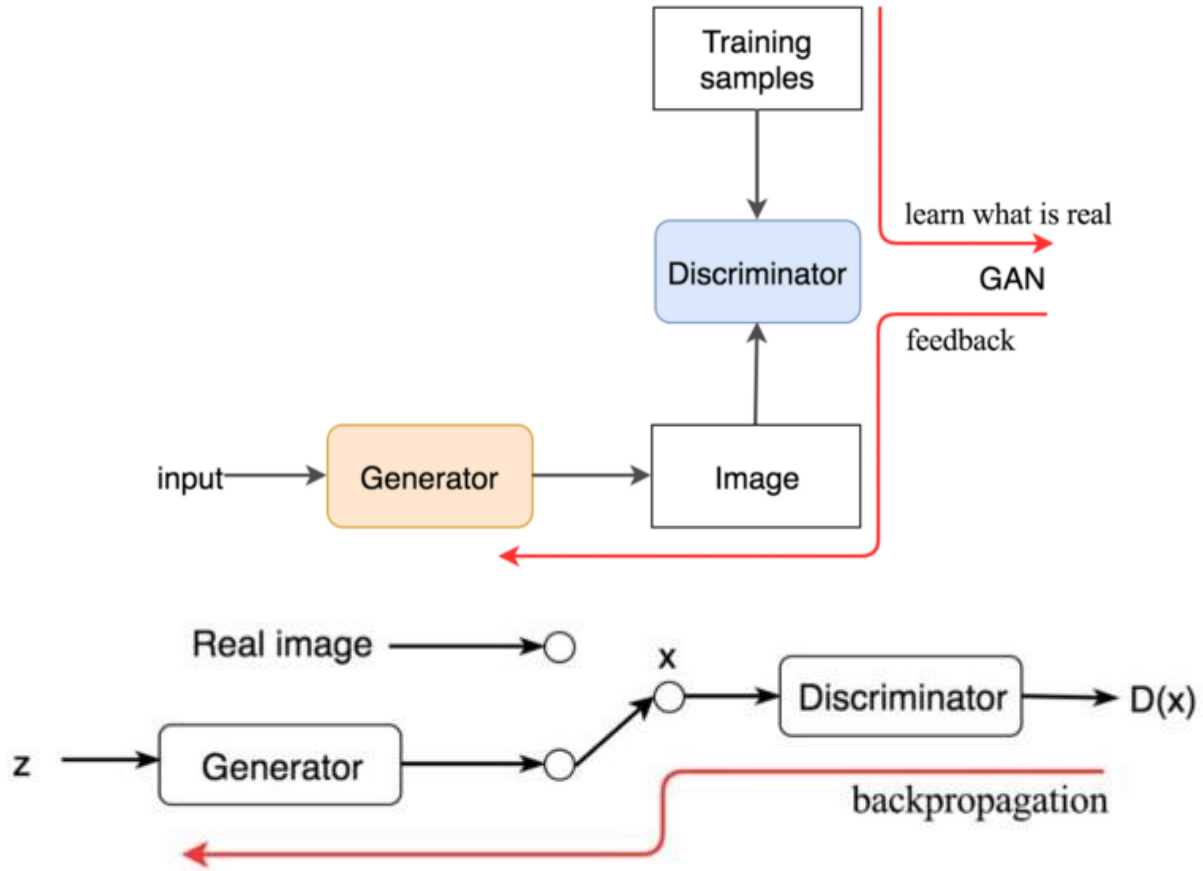
Architecture of GAN



How Discriminator and Generator learns?



Learning in GAN



Min max game

$$\max_D V(D) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$

recognize real images better — recognize generated images better

$$\min_G V(G) = \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$

Optimize G that can fool the discriminator the most.

We often define GAN as a minimax game which ***G*** wants to minimize ***V*** while ***D*** wants to maximize it.

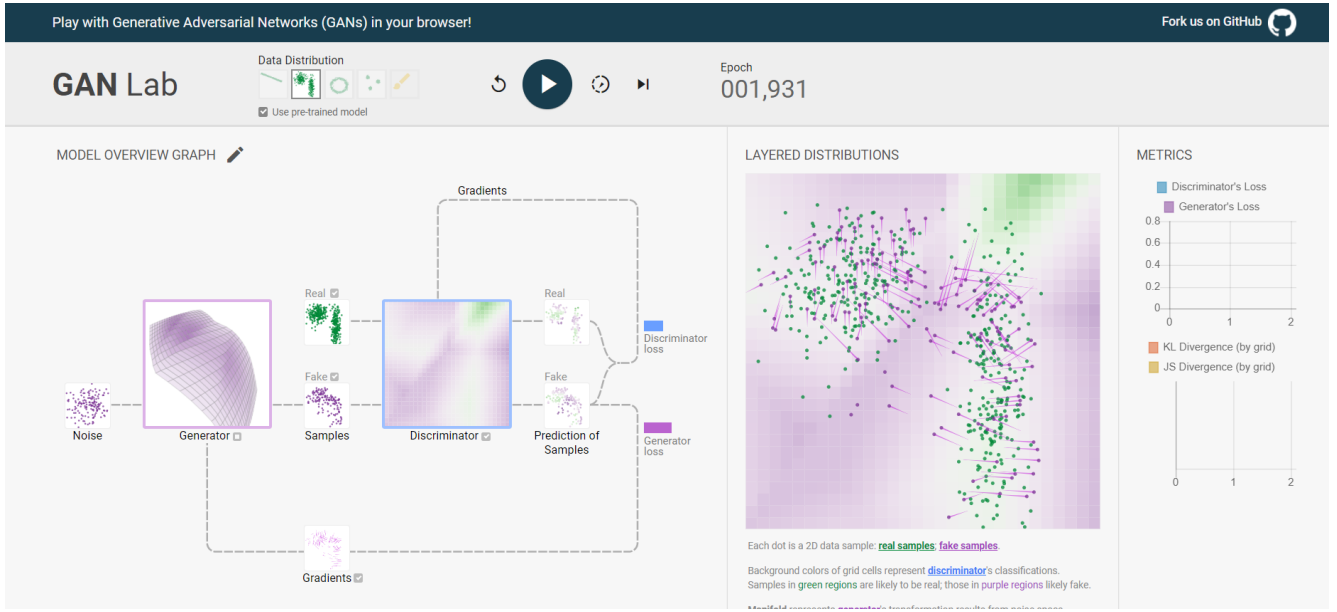
$$\min_G \max_D V(D, G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))].$$

Working of GANs

- Gradient descent
- Back propagation
- Min-max functions for Generators and discriminators
- Two distinct adversarial roles
- Optimization techniques -> making the model better over time
- Training loop is where we teach G to trick D and D to beware G
- Components:
    1. Original data set
    2. Noise
    3. Generator (G)
    4. Discriminator (D) with loss function

GAN lab

- https://poloclub.github.io/ganlab/
- Let us play with the GANs
- Key Takeaways:
- Pick a data distribution
- Let training begin
- Visualizing generator and discriminator
- Understanding interplay
- Parameters to choose in GAN – interactive experimentation



Types of GANs

- Deep Convolutional GAN (DCGAN)
- Conditional GAN (CGAN)
- Auxiliary Class GAN (ACGAN)
- Wasserstein GAN (WGAN)

Deep Convolutional GAN (DCGAN)

DCGANs uses Convolution Neural Networks (CNNs) to build generator and discriminator

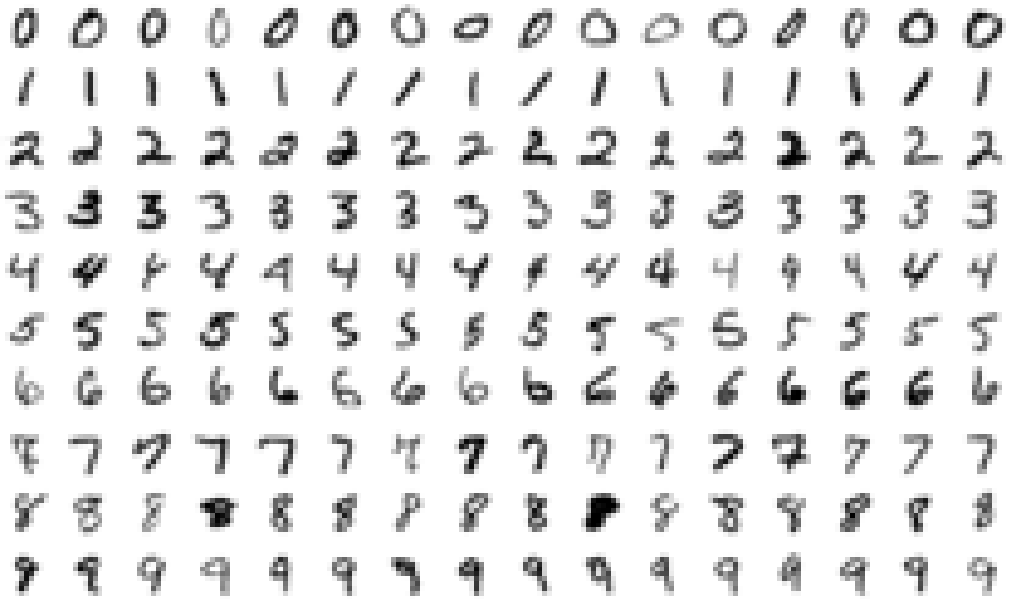
What is Noise or random numbers?

- Input to Generator

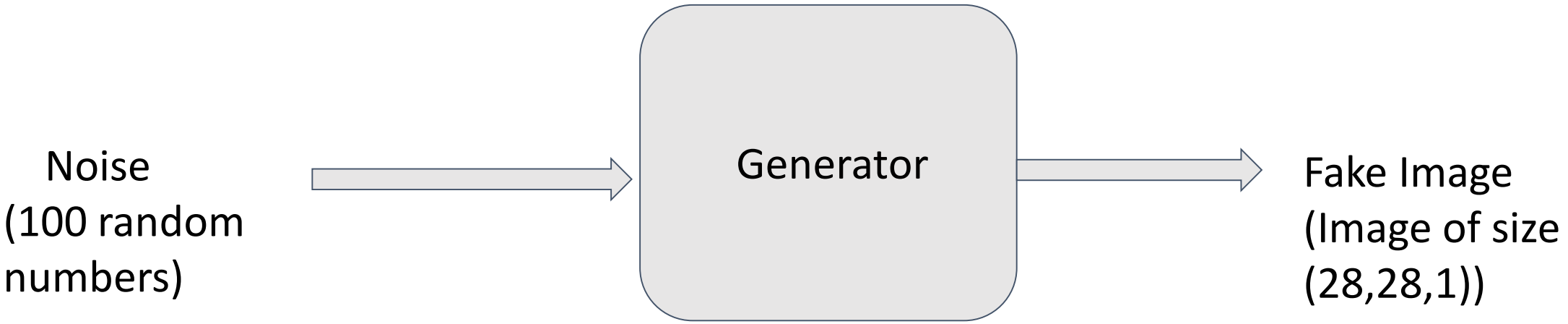
What is Fake data?

- Any type of data which we want Generator to create
- Ex: image of a face

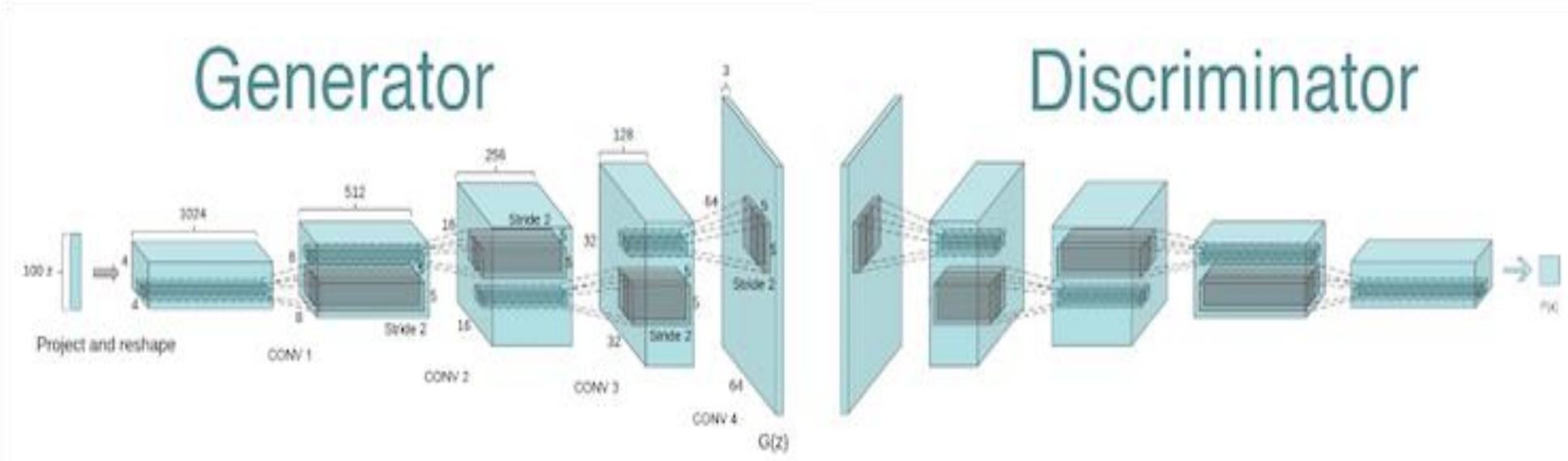
DCGANs of MNIST



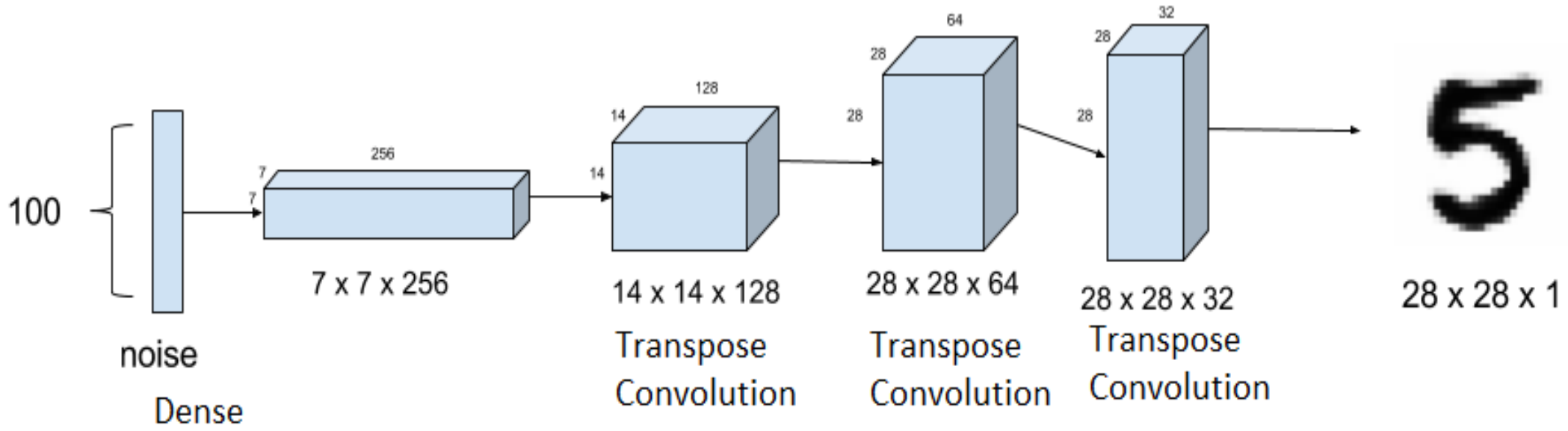
Input and Output of Generator for MNIST



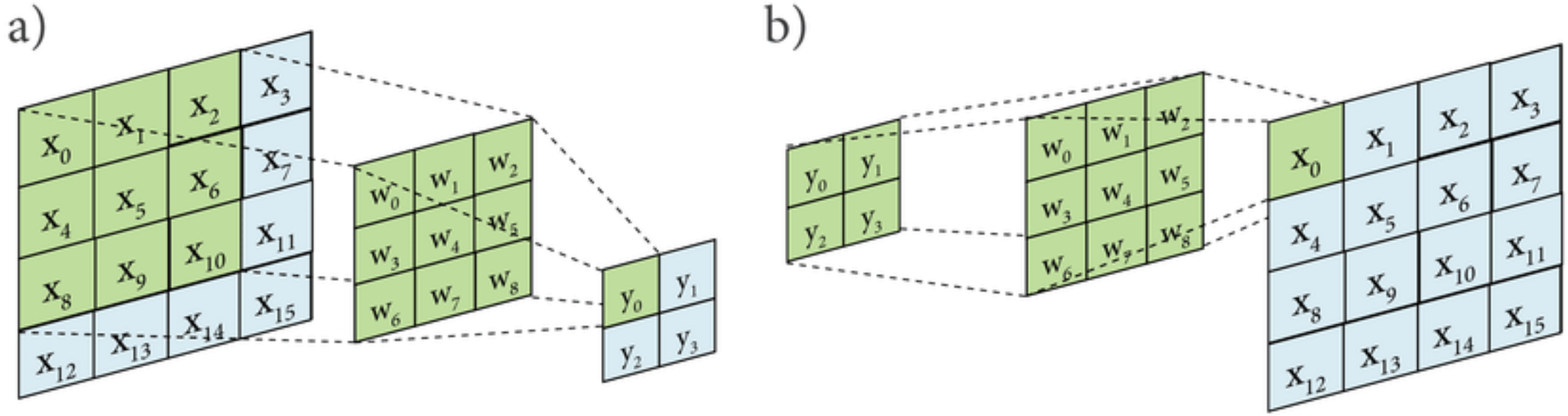
Architecture of DCGAN



DCGAN Generator Simplified



What is Transpose Convolution?

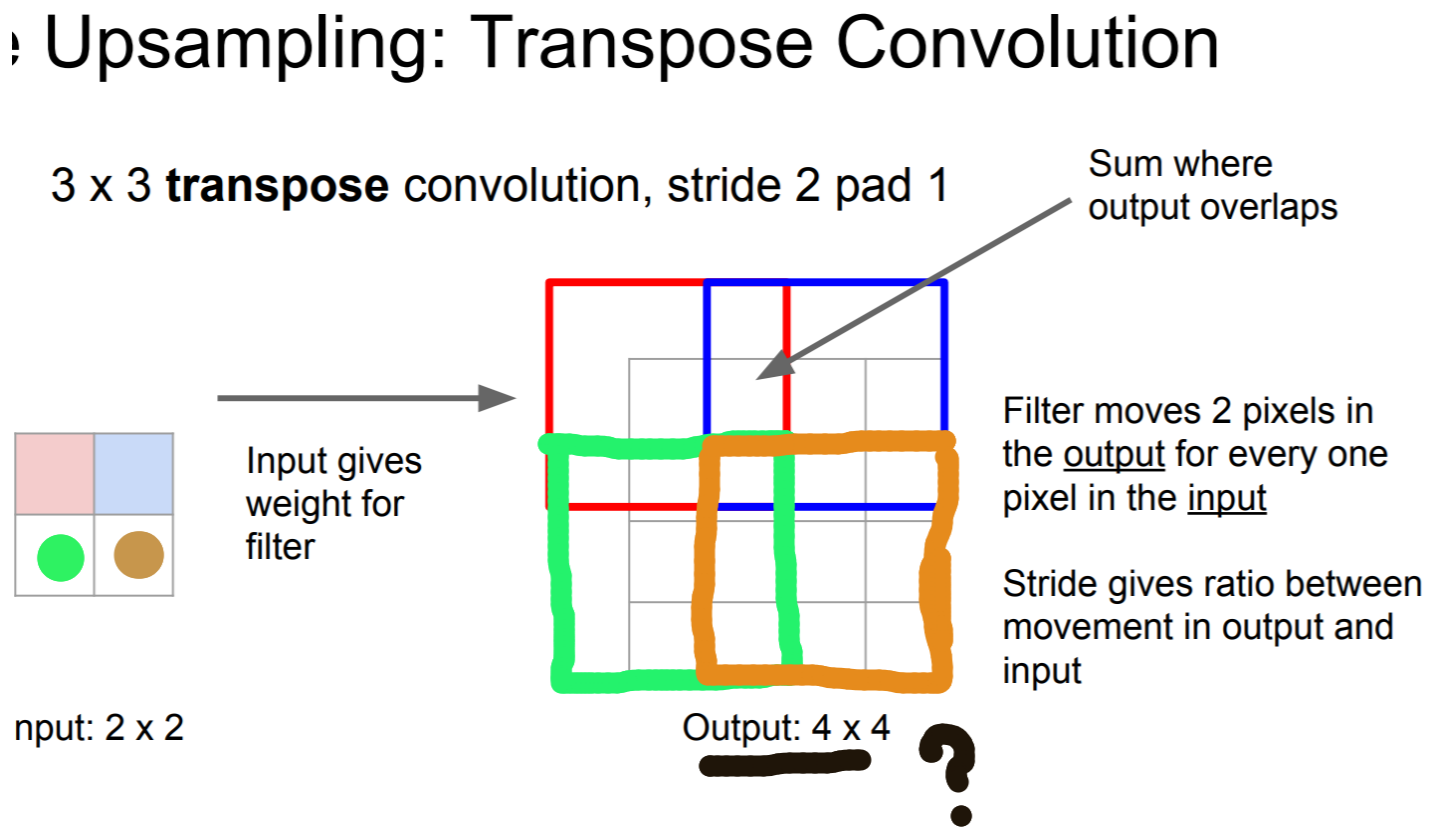


a) Traditional Convolution
b) Transpose Convolution

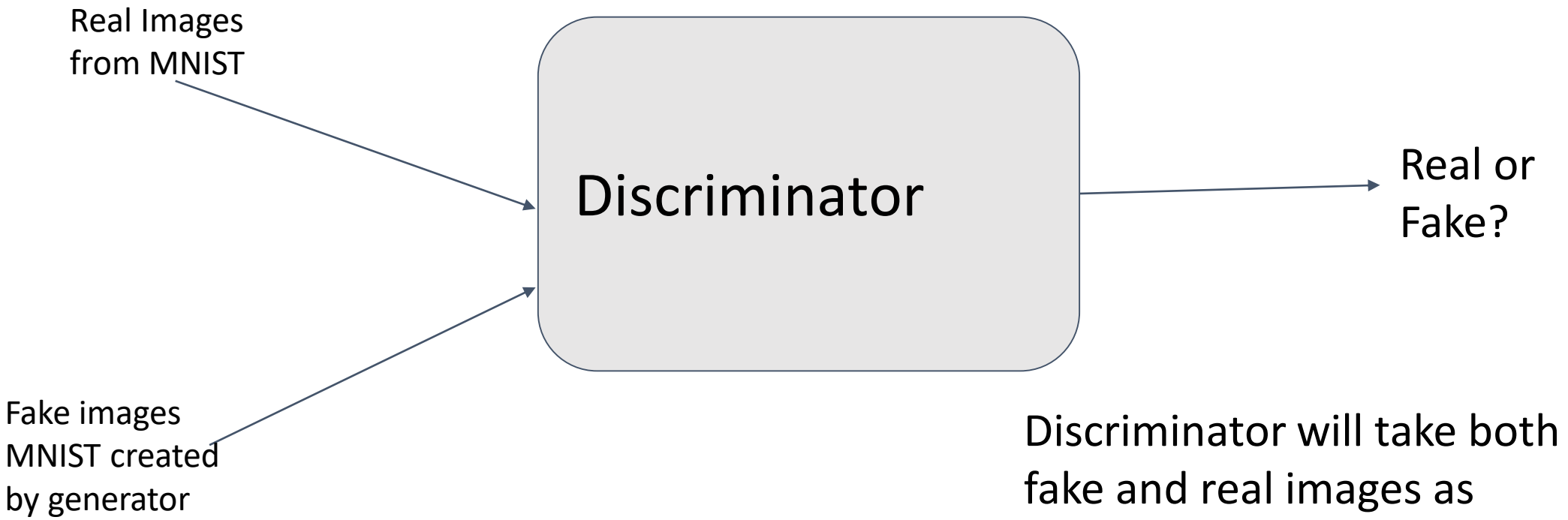
Question?

What happens to overlapping region?

Ans - Added together.

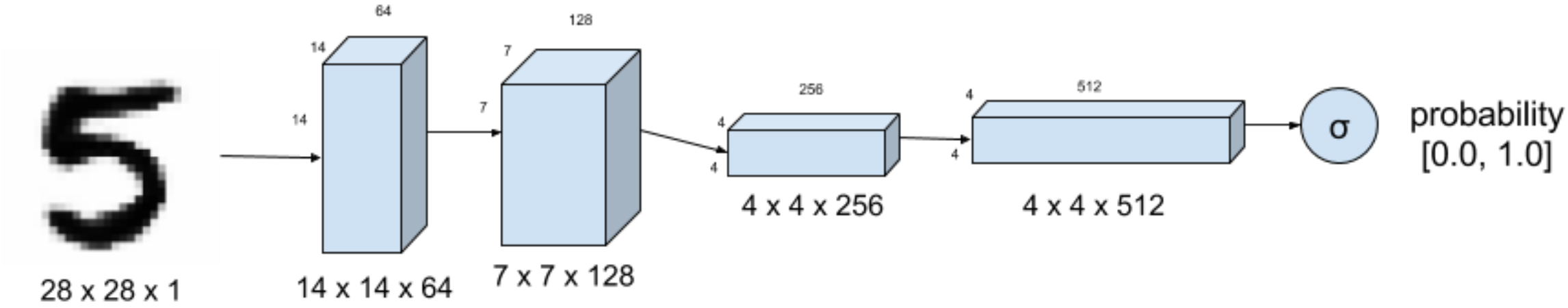


DCGAN Discriminator



Discriminator will take both fake and real images as input and ...classify them as Real OR Fake

Discriminator DCGAN

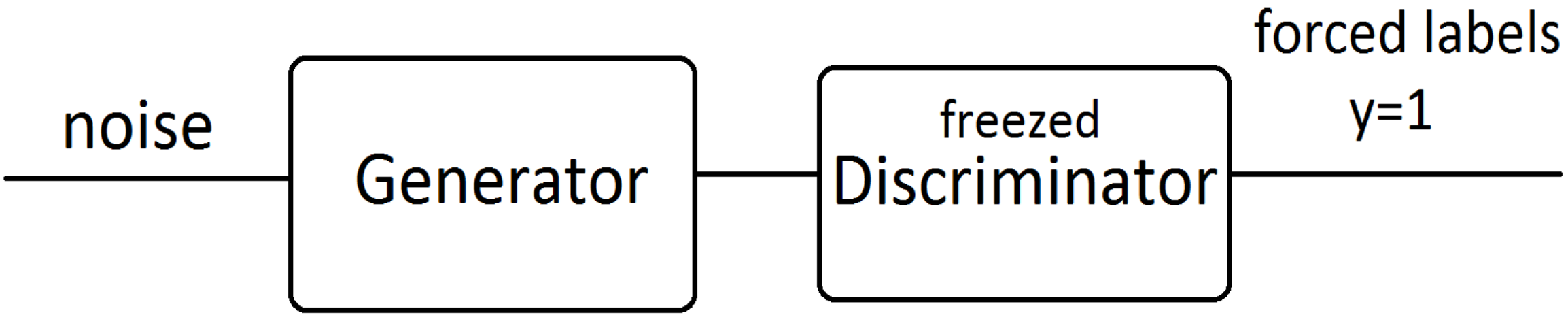


Discriminator of DCGAN tells how real an input image of a digit is. MNIST Dataset is used as ground truth for real images. Strided convolution instead of max-pooling down samples the image.

The adversarial network

- We setup the GAN, which chains the generator and the discriminator.
- This is the model that, **when trained, will move the generator in a direction that improves its ability to fool the discriminator**.
- This model turns latent space points into a classification decision, “fake” or “real”, and it is meant to be trained with labels that are always “these are real images”.
- So training GAN **will update the weights of generator** in a way that makes discriminator more likely to predict “real” when looking at fake images.
- Very importantly, we set the discriminator to be frozen during training (non-trainable): its weights will not be updated when training GAN.

Adversarial Network



Steps for building DCGAN

Build Models

- Generator using Convolution Transpose and noise as input
- Discriminator with Noise and Real data as input
- Adversarial (Generator + Frozen Discriminator)

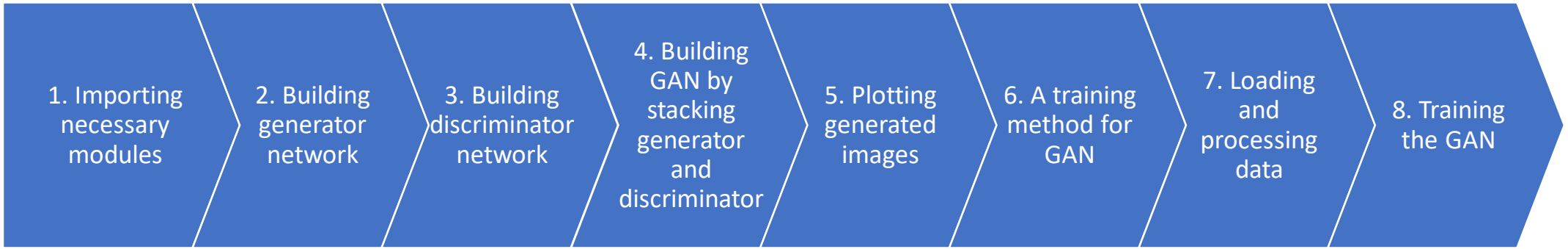
Training Models

- Step 1: Train Discriminator using both Real Data and Fake data from Generator
- Step 2 : Train Adversarial Network (indirectly training Generator) using fake data only
- Repeat Step 1 and 2 till Generator is producing desired outputs

DCGAN for MNIST in Keras

Notebook in Keras+TF

Building GAN pipeline:



Conditional GANs(CGANs)

What if we only need 2?

Or any other specific number

These are GANs that use **extra label information**.

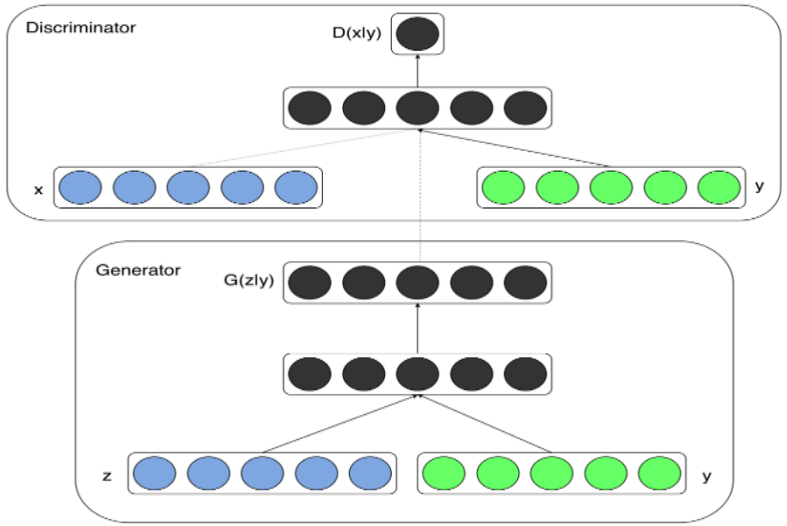
This results in better quality images and being able to control – to an extent – how generated images will look.

Conditional GAN

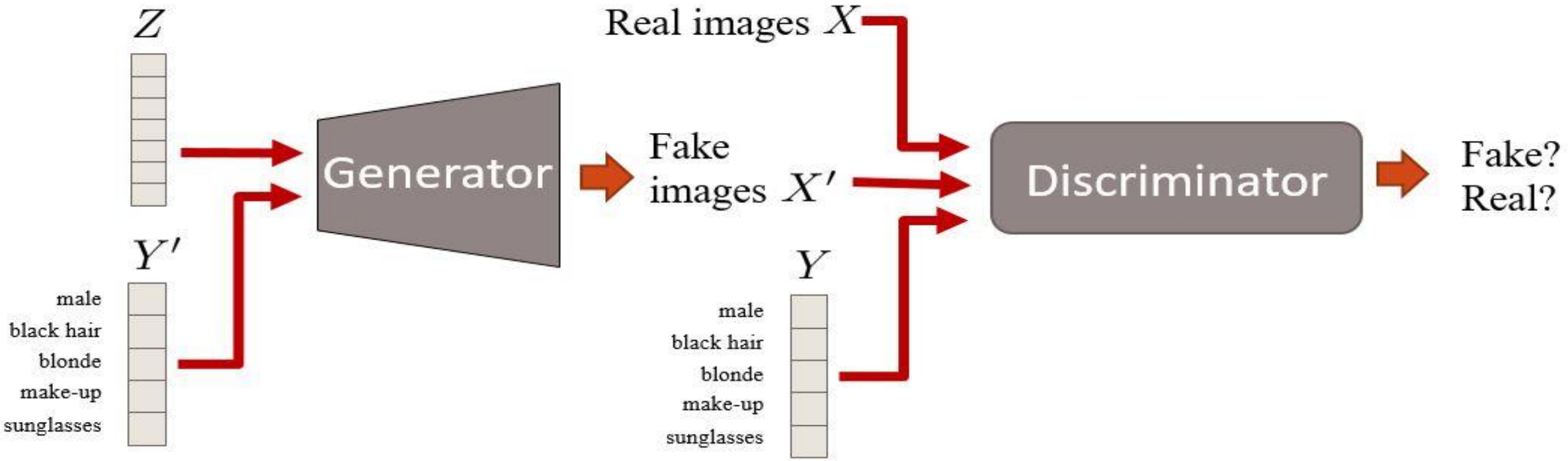
The architecture of CGAN is similar to GAN with extra variable ‘y’ being added to generator and discriminator to generate images of specific label.

Following is the objective of CGAN

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x}|\boldsymbol{y})] + \mathbb{E}_{\boldsymbol{z}\sim p_z(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z}|\boldsymbol{y})))]$$



Conditional GANs



Here we have **conditional information Y** that describes some aspect of the data.
For example, if we are dealing with faces, Y could describe attributes such as hair colour or gender. Then, this attribute information is inserted in both the generator and the discriminator.

CGANs on MNIST

Differences between Z and

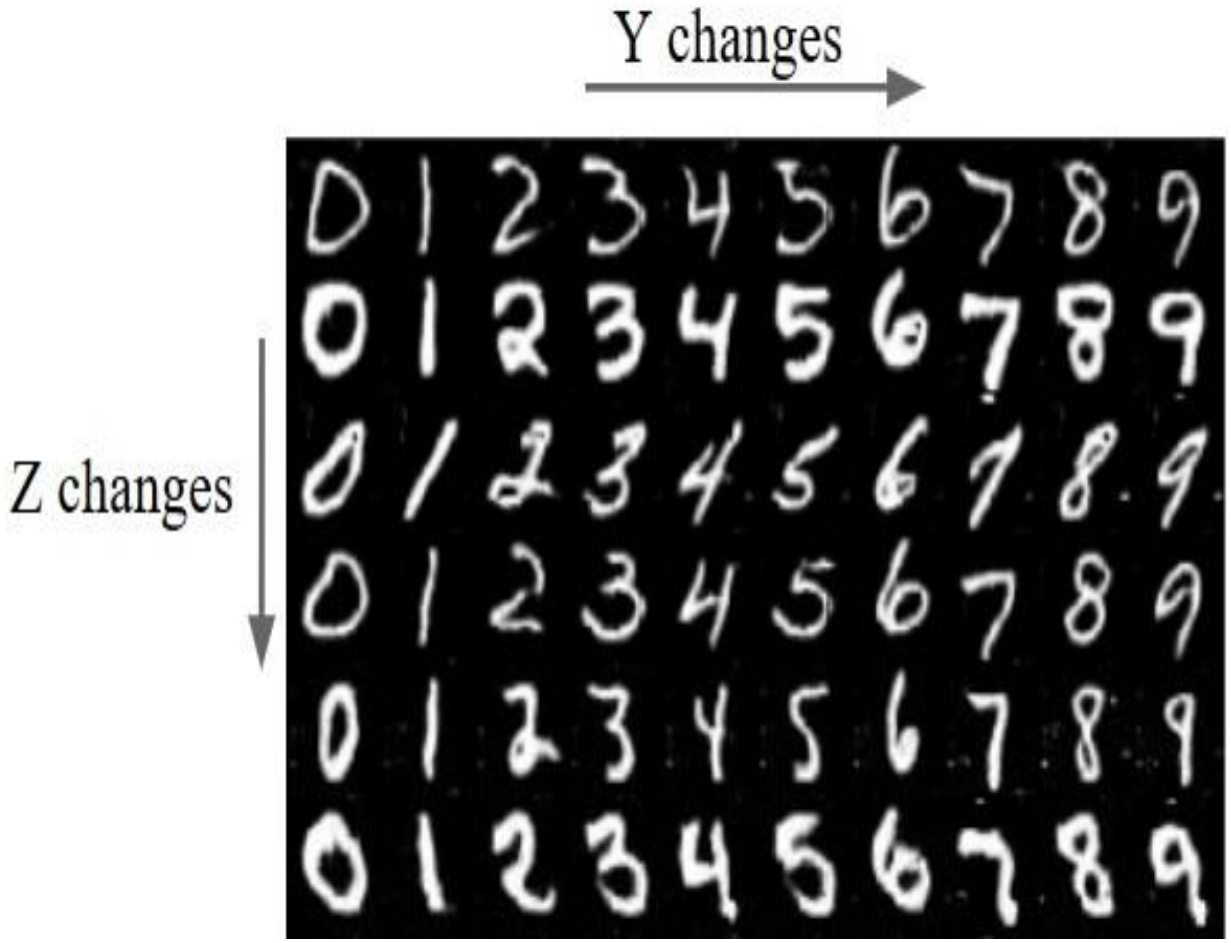
Y on MNIST samples.

Z is fixed on rows and

Y on columns.

Z encodes the style of the number

And Y encodes the number itself.



MNIST Conditional GANs

Notebook

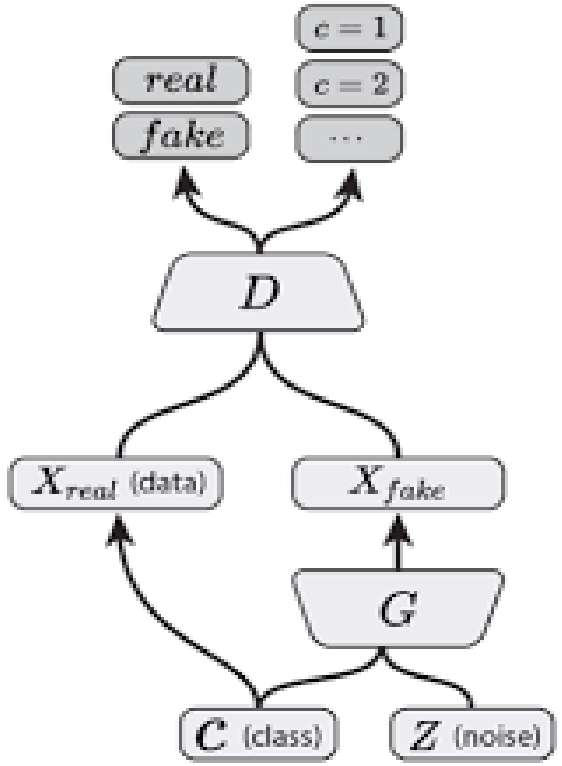
Observe the difference in the code:

- Discriminator
- Labels are considered

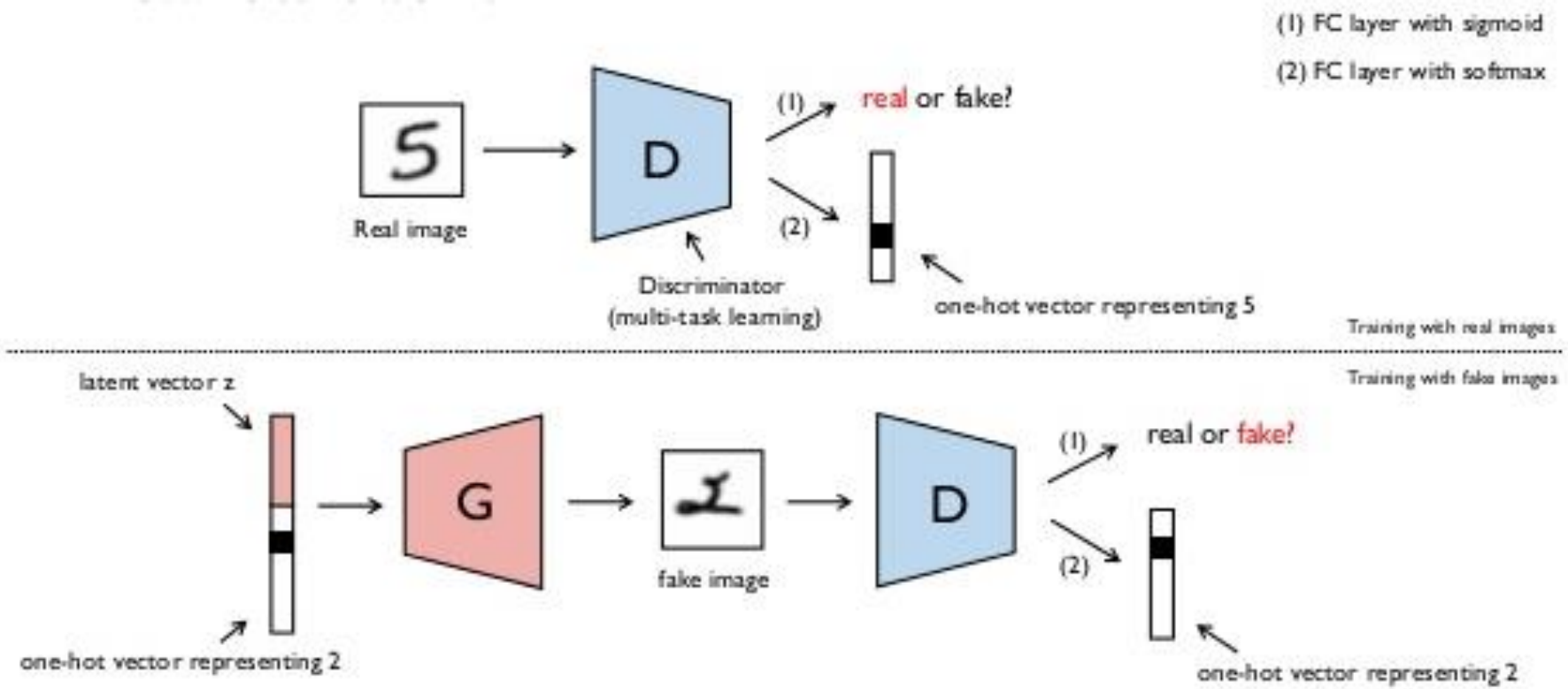
Auxiliary Classifier GAN

- This architecture is similar to condition GAN.
- An additional factor in the architecture is that discriminator not only predicts if an image is fake/real but also the class label of particular image.

- **Architecture of ACGAN:**



ACGAN Architecture



Wasserstein GAN

- To seek an alternate way of training the generator model to better approximate the distribution of data observed in a given training dataset.

- **Earth-mover distance**

- Instead of using a discriminator to classify or predict the probability of generated images as being real or fake, the WGAN changes or replaces the discriminator model with a critic that scores the realness or fakeness of a given image.

- The **Wasserstein loss** function seeks to increase the gap between the scores for real and generated images.
- Critic Loss = [average critic score on real images] – [average critic score on fake images]
- Generator Loss = -[average critic score on fake images]

Challenges in training the GANs

- Vanishing Gradients
- Non-convergence
- Mode Collapse
- Evaluation Metrics
- Unbalance between the generator and discriminator causing overfitting
- Highly sensitive to the hyperparameter selection

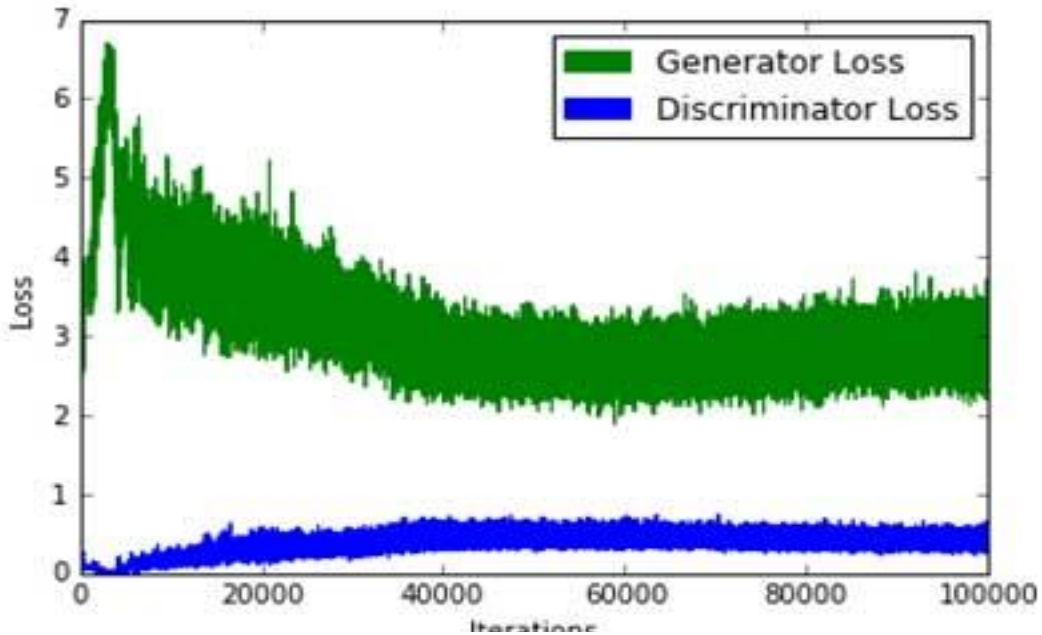
Vanishing Gradients

- Decay of information through time
- **Vanishing Gradient** Problem occurs when we try to train a Neural Network model using ***Gradient based optimization techniques***.
- The discriminator gets too successful that the generator **gradient vanishes and learns nothing.**

Non-Convergence

The model parameters oscillate, destabilize and never converge.

A common question in GAN training is “when do we stop training them?”. Since the Generator loss improves when the Discriminator loss degrades (and vice-versa), we can not judge convergence based on the value of the loss function.



Plot of a typical GAN loss function. Note how convergence cannot be interpreted from this plot. (Source)

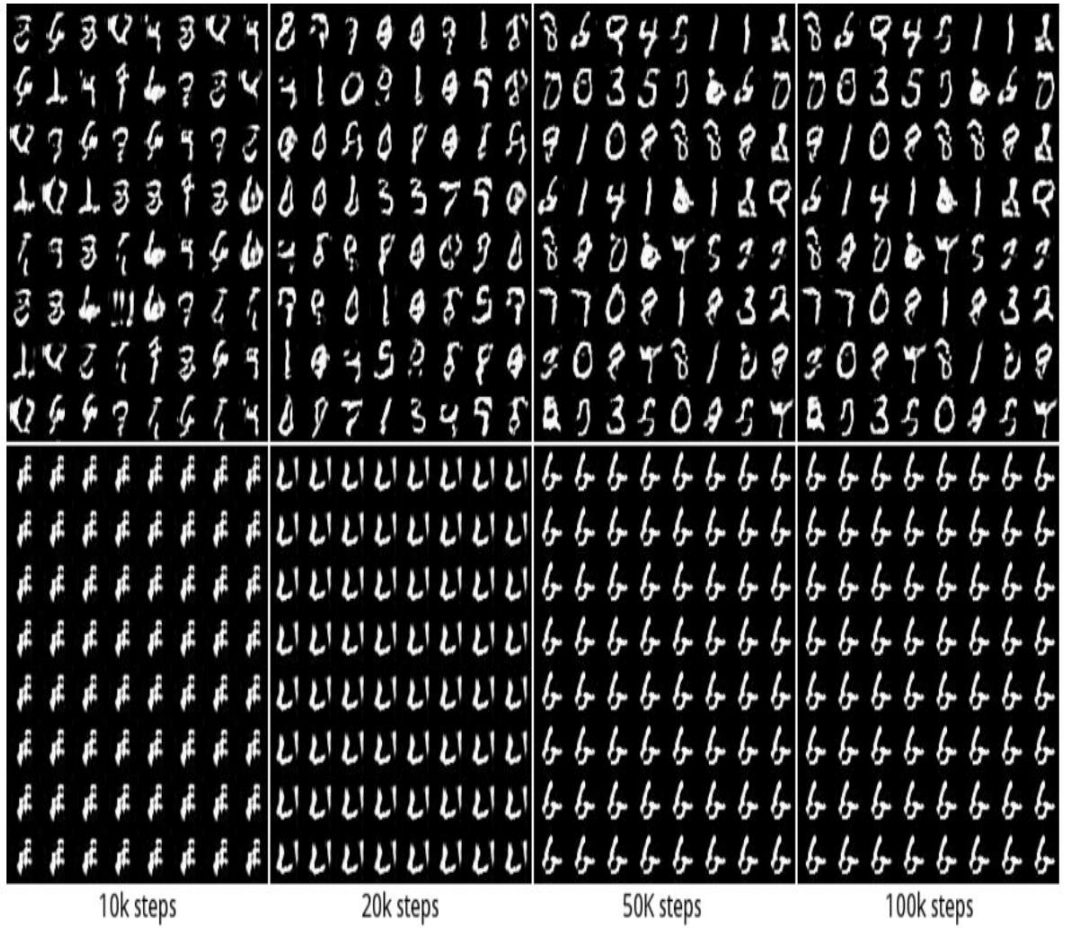
Mode Collapse problem

- Common failure case for GANs - the generator learns to produce **samples with extremely low variety.**
- Most interesting real-world data distributions are highly complex and *multimodal*.
- The probability distribution which describes the data may have multiple "peaks" where different sub-groups of samples are concentrated.
- The generator will exhibit very poor diversity amongst generated samples, which limits the usefulness of the learnt GAN.
- In reality, the severity of mode collapse varies from ***complete collapse*** (all generated samples are virtually identical) to ***partial collapse*** (most of the samples share some common properties)

Mode Collapse

Example
In MNIST, there are 10 major modes from digit ‘0’ to digit ‘9’. The samples below are generated by two different GANs. The top row produces all 10 modes while the second row creates a single mode only (the digit “6”).

This problem is called mode collapse when only a few modes of data are generated.



Source https://arxiv.org/pdf/1611.02163.pdf

Evaluation Metrics

- It’s hard to measure GANs performance without manually visualizing the output
- Add synthetically generated images to original dataset and see the improvement in confusion matrix

To avoid overfitting

- Imbalance between generator and discriminator
  - To balance their training to avoid overfitting
- **Hyperparameter tuning** needs patience.
- No cost functions will work without spending time on the hyperparameter tuning.
- In a discriminative model, the loss measures the accuracy of the prediction and we use it to monitor the progress of the training.
- We need to examine the generated images **manually** to verify the progress
  - Complicated tuning process

Some tips for making GANs train

- Normalize images between -1 and 1, use tanh as last layer in Generator
- Use Gaussian noise and not Uniform
- Modified Loss function
- Flip labels when training generator
- Avoid Sparse Gradients i.e avoid Relu and use LeakyRelu
- Use soft and Noisy Labels
- Use Adam
- Add noise to inputs
- Use Dropouts in Generator

This list might not work in all cases and solve the training problems completely.

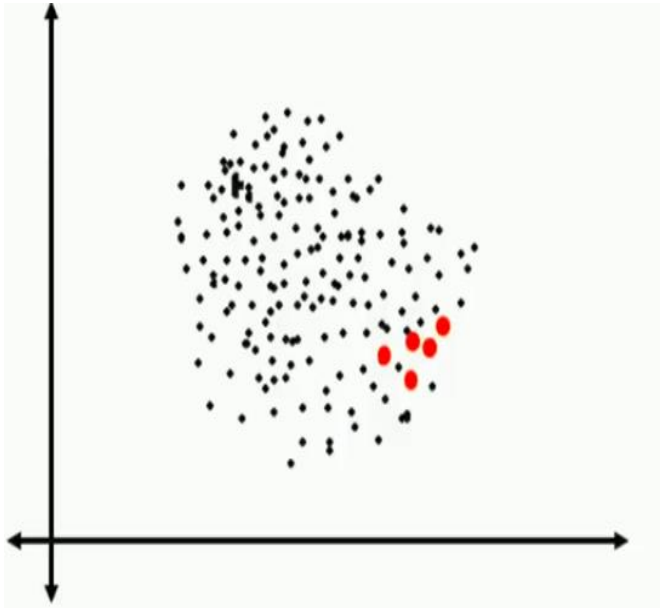
Synthetic class specific image generation with GAN

Unbalanced Data

- Presence of minority class in the dataset (< 15%)
- Challenges related to Imbalanced dataset:
  - Misleading accuracy
  - Biased predictions

Few examples:

- Disease diagnosis
- Credit card frauds
- Machine defects
- Natural disasters



Two Class Classification

No-Fraud → 99.5%
Fraud → 0.5%

How to use GANs for our problem?

- We explored the GAN's framework in **medical image** context.
- The goal of the project is to generate images of a **particular class** in a multi-class dataset, so that the generated images can be used for data augmentation and a better classifier can be built on top of it.
- We have performed experiments on two datasets (Breast Cancer (Mammography) and Cervical Cancer.
- We conducted experiments using Conditional Generative Adversarial Networks (**CGAN**)
- Auxiliary Classifier Generative Adversarial Networks (**ACGAN**) and Deep Convolution Generative Adversarial Networks (**DCGAN**)

Experiments

- Implemented AC-GAN model to IntelAI cancer dataset, now able to generate images of size 128x128.
- Problem with this is it is generating data of particular data every time and hence facing mode collapse problem.
- Tuning of AC-GAN to see if mode collapse problem is eliminated.
- Experimented AC-GAN on mammography dataset.

- Breast Cancer (Mammography):
  - Data size
- Cervical Cancer:
  - Data size

Experiments

- Experiments are carried out using Conditional GAN and Auxiliary Classifier GAN (ACGAN) and architecture of CGAN is taken from DCGAN.
- Initial experiments are carried out on 256x256 images but results were not good.
- So started with 64x64 which worked with CDCGAN (CGAN + DCGAN).
- Then started with 128x128 images with ACGAN architecture it gave better results in this resolution compared to CDCGAN.
- Experiments with VAE’s did not result in much better results.

Steps to stabilize GAN training

- We have observed that GAN training is too unstable.
- A normal approach for training did not worked with our datasets. At Every instance of training one of the Network used to win which is not desired.
- So we have used the following hacks to stabilize training of GAN. (These hacks are taken from DCGAN paper)
  - Used batch normalization, dropouts, Relu layers.
  - WGAN loss did not help for GAN stabilization.
  - Add Noise at every layer of Generator and at the input of Discriminator. This helped in stabilizing the training and it made sure that no network wins the game.
  - Add label smoothing (0.8-1.2 and 0-0.3). It stabilized training.
  - **Noise plus label smoothing** works like beast. But this takes more iterations to stabilize.

Results with CDCGAN and AC-GAN

With CDCGAN: - (64x64)

With ACGAN: - (128x128)

Number of epochs vs Images Generated:-

After Epoch = 0:-

After Epoch = 10:-

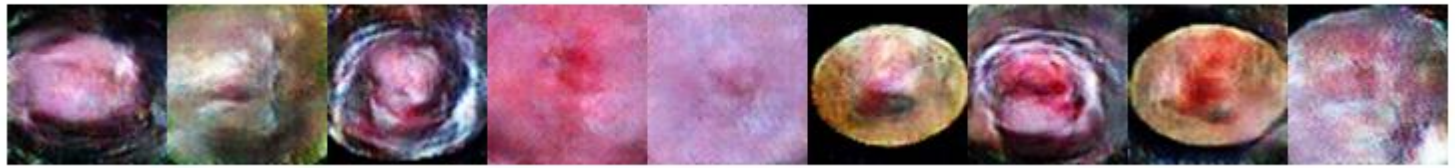
After Epoch = 50:-

Results

After Epoch = 100:-

After Epoch = 300:-

After Epoch = 500:-



After Epoch = 900:-

Key Takeaways from the experiment

- Using CGAN, DCGAN, ACGAN architectures we are able to generate images of 64x64 and 128x128 sizes to some extent.
- At this stage, the generated images cannot be used for production usage. This is due to less number of images in the dataset.
- 400 images in Mammography dataset and 1000 images in Cervical Cancer dataset and varied distribution.
- We verified this with 1000 images from MNIST dataset and recognizable images are getting generated and 5000-10000 images of Chest-X-ray.
- So for the medical image data distribution of cervical cancer data we need more number of images to work in this approach.

Other Cool GAN Applications

Applications of GANs

Towards the automatic Anime

characters creation

(https://arxiv.org/pdf/1708.05509.pdf)

Pose Guided Person Image Generation

https://papers.nips.cc/paper/6644-pose-guided-person-image-generation.pdf

Pose Guided Person Generation Network

- Allows to synthesize person images in arbitrary poses, based on an image of that person and a novel pose.
- This generation framework utilizes the pose information explicitly and consists of two key stages:
  - **Pose integration**: In the first stage the condition image and the target pose are fed into a **U-Net-like** network to generate an initial but coarse image of the person with the target pose.
  - **Image refinement**: The second stage then refines the initial and blurry result by training a U-Net-like generator in an adversarial way

Pose Guided Person Image Generation

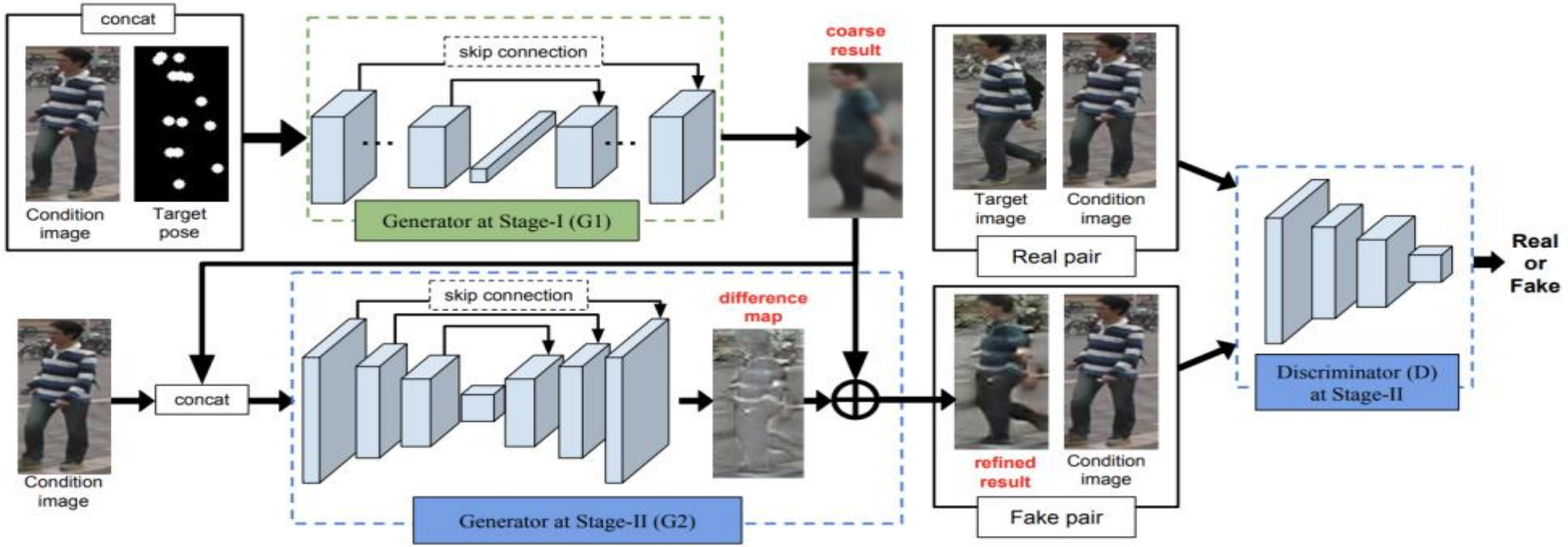
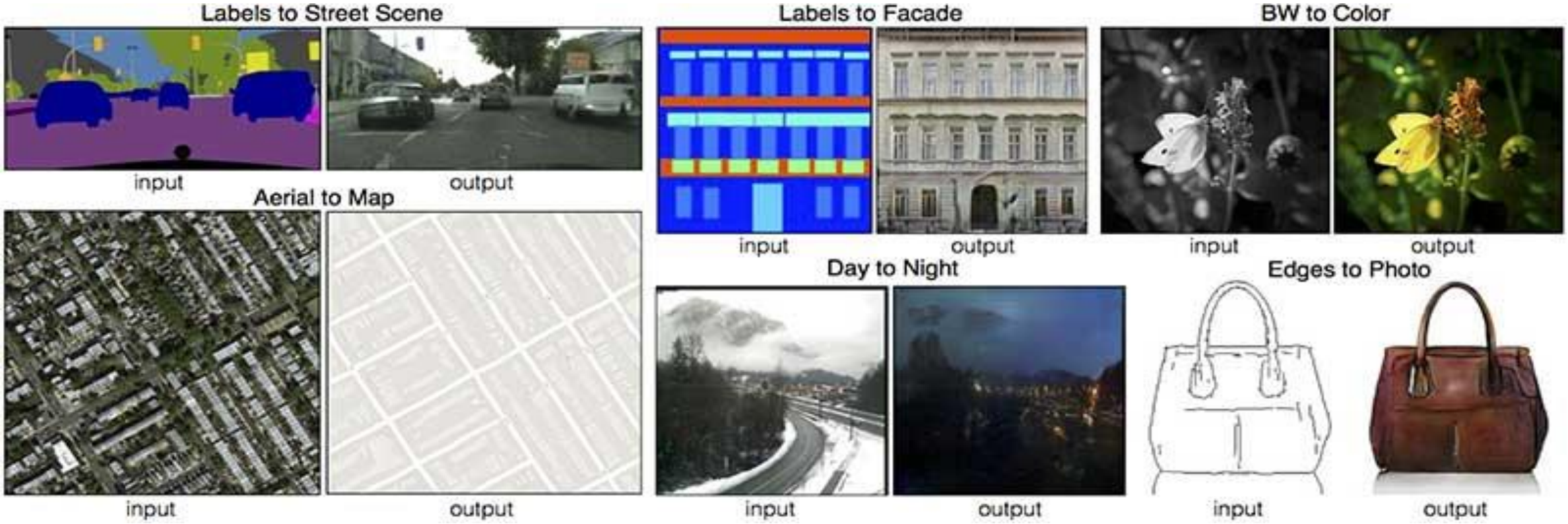


Figure 2: The overall framework of our Pose Guided Person Generation Network (PG^2). It contains two stages. Stage-I focuses on pose integration and generates an initial result that captures the global structure of the human. Stage-II focuses on refining the initial result via adversarial training and generates sharper images.

Image translation using GANs



Source - https://phillipi.github.io/pix2pix/

Cycle GANs

- Unpaired Image-to-Image
- Translation using Cycle-Consistent

Adversarial Networks

- To translate an image from a source domain X

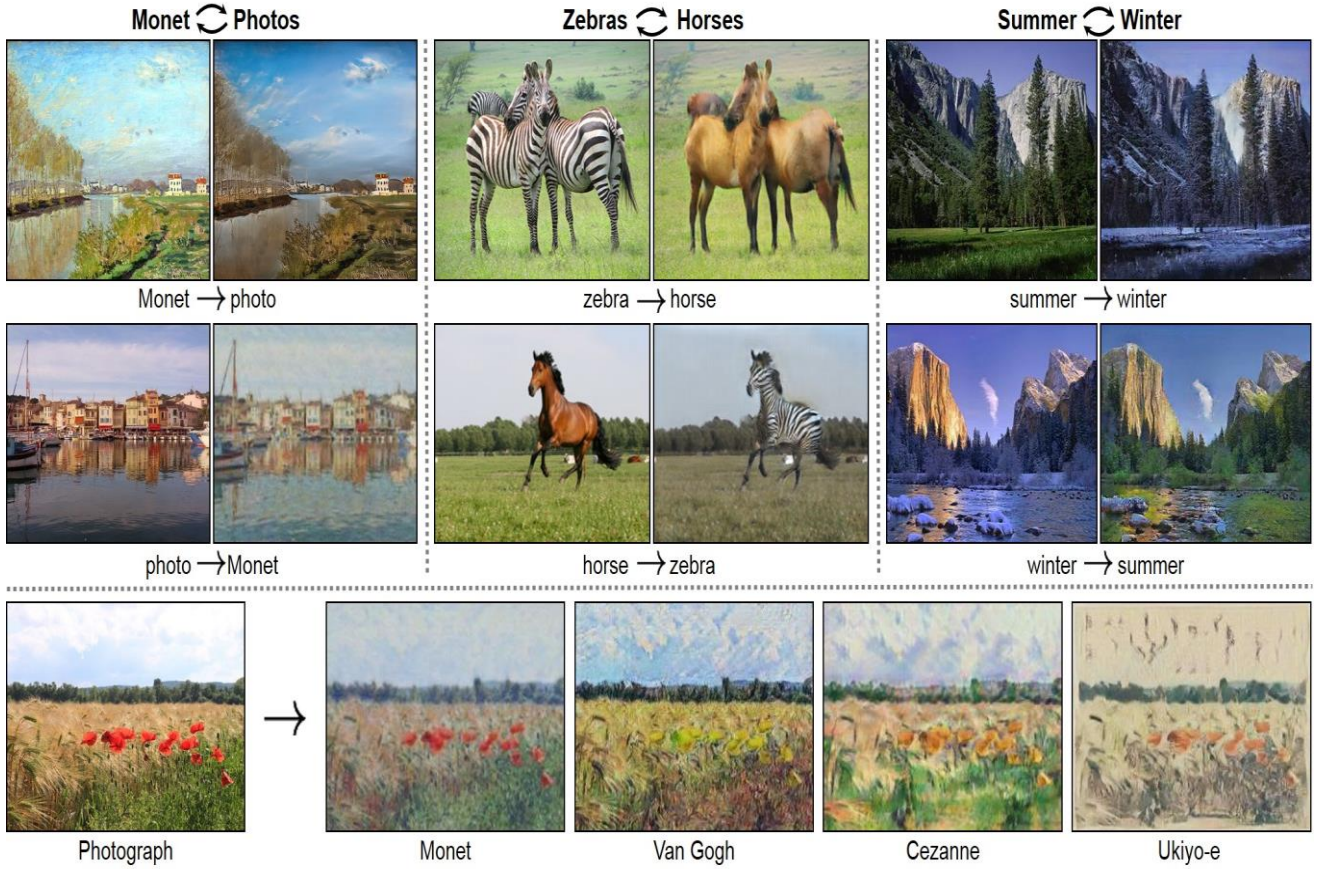
to a target domain Y in the absence of

paired examples

Examples:

  - Season transfer

  - Object Transfiguration



Super Resolution GAN (SRGAN)

https://arxiv.org/pdf/1609.04802.pdf

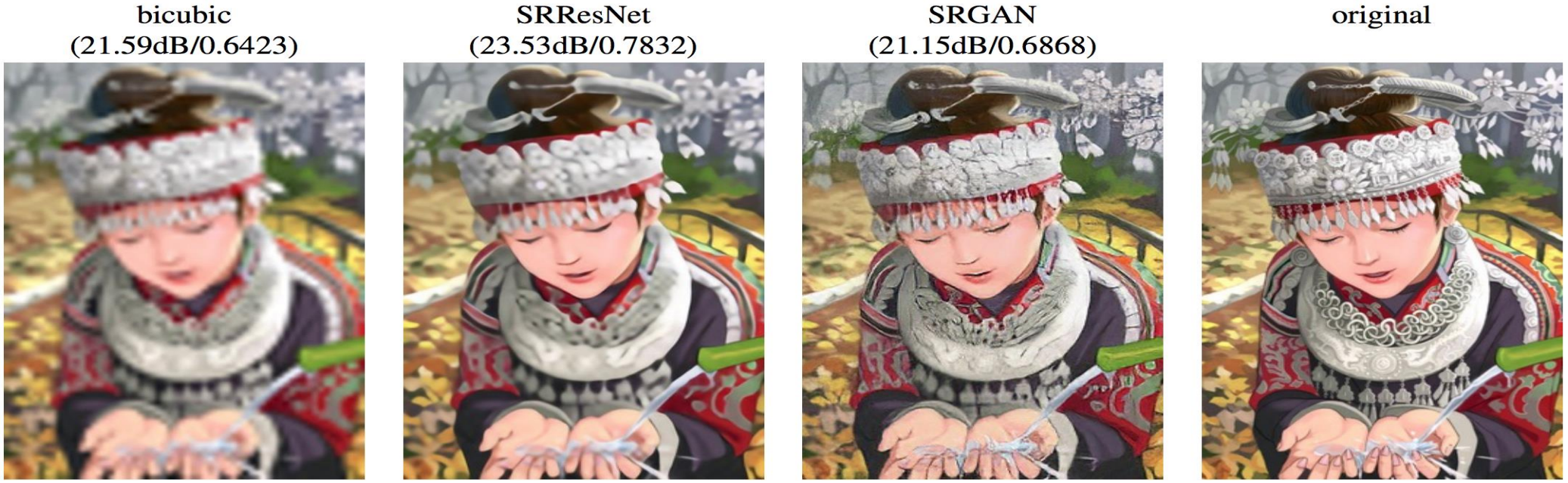


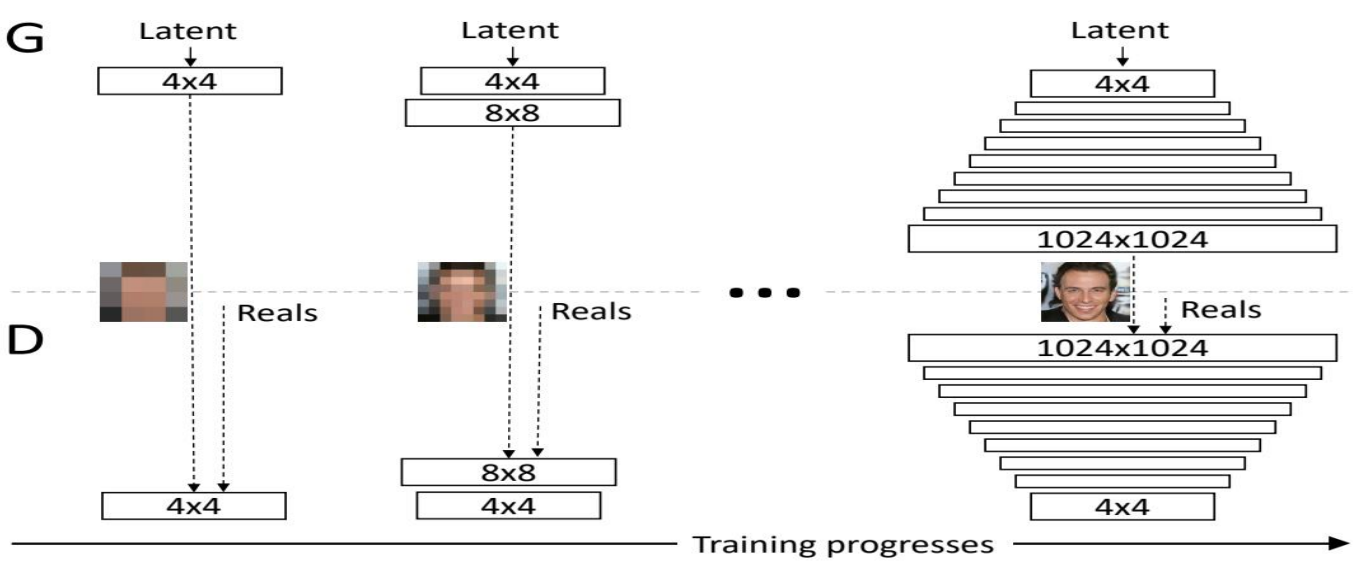
Figure 2: From left to right: bicubic interpolation, deep residual network optimized for MSE, deep residual generative adversarial network optimized for a loss more sensitive to human perception, original HR image. Corresponding PSNR and SSIM are shown in brackets. [4× upscaling]

SRGAN

- Super-resolution imaging (SR) is a class of techniques that enhance the resolution of an imaging system
- How do we recover the finer texture details when we super-resolve at large upscaling factors
- Focused on minimizing the mean squared reconstruction error
- A perceptual loss function which consists of an adversarial loss and a content loss
- PSNR – Peak Signal to Noise Ratio
- SSIM – Structural Similarity

Progressive GANs

https://arxiv.org/pdf/1710.10196.pdf



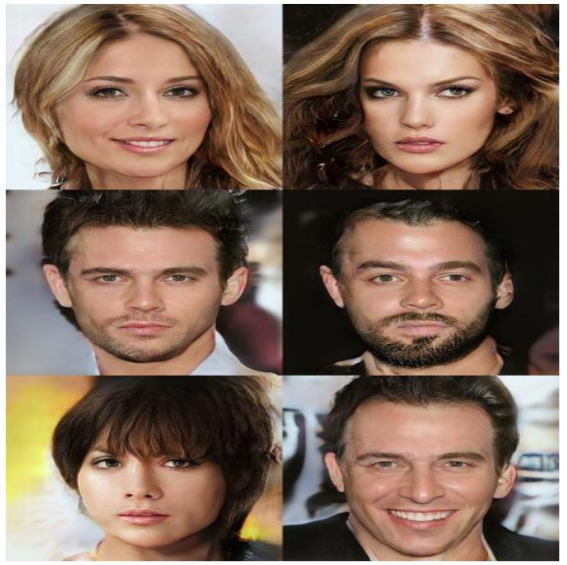
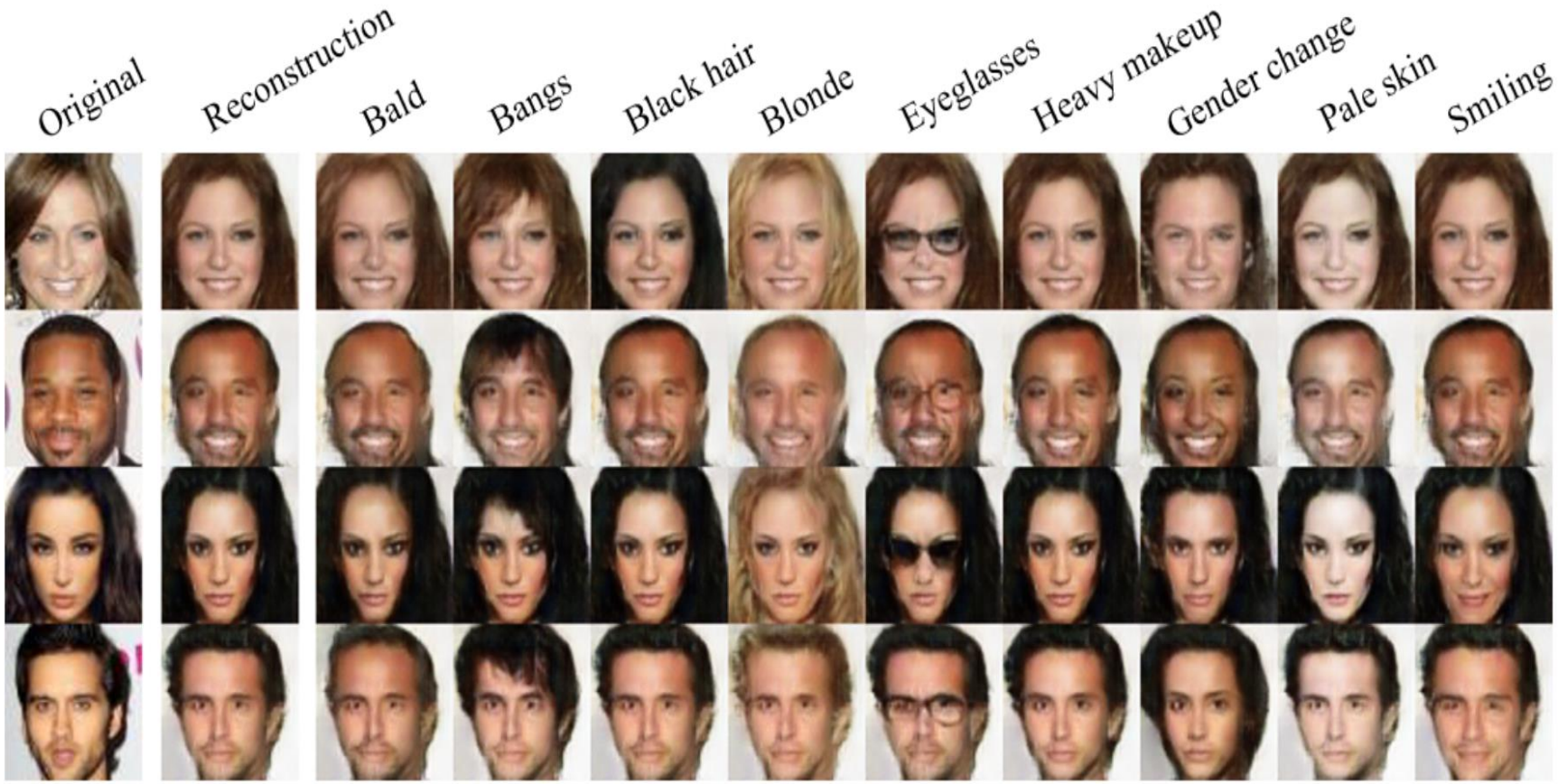


Figure 1: Our training starts with both the generator (G) and discriminator (D) having a low spatial resolution of 4×4 pixels. As the training advances, we incrementally add layers to G and D, thus increasing the spatial resolution of the generated images. All existing layers remain trainable throughout the process. Here $N \times N$ refers to convolutional layers operating on $N \times N$ spatial resolution. This allows stable synthesis in high resolutions and also speeds up training considerably. One the right we show six example images generated using progressive growing at 1024×1024.

Reconstruct or edit images with specific attributes

IcGANs

https://github.com/Guim3/IcGAN

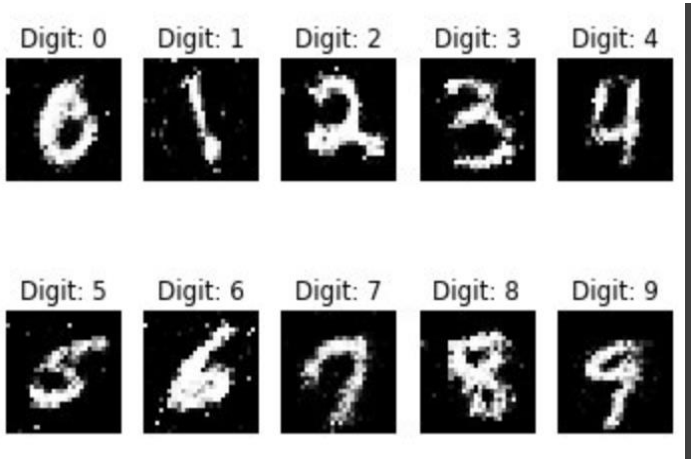


Some more Applications

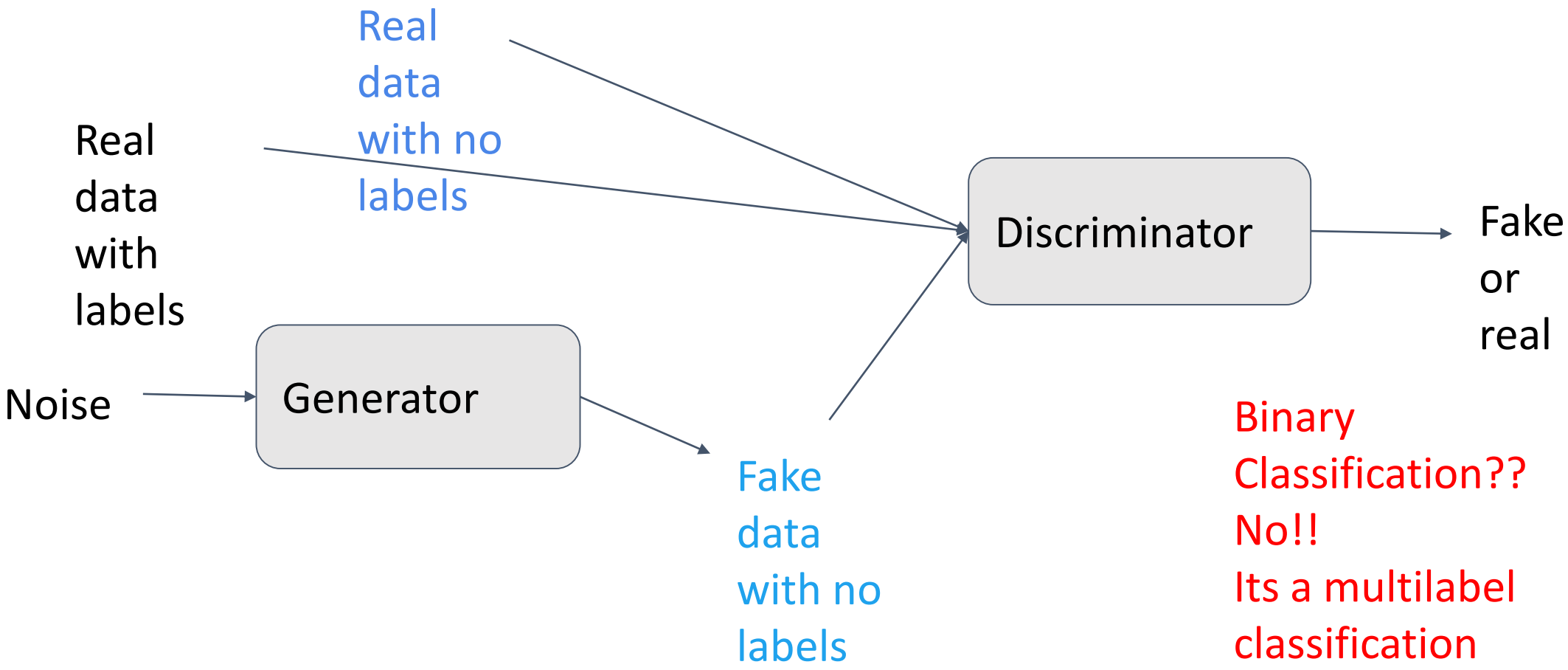
- Video Generation- https://www.youtube.com/watch?v=Pt1W_v-yQhw&feature=youtu.be
- Generate 3D Objects - 3DGAN https://www.youtube.com/watch?v=HO1LYJb818Q&feature=youtu.be
- Music Generation - MidiNet - https://arxiv.org/pdf/1703.10847.pdf
- **The GAN Zoo-** https://github.com/hindupuravinash/the-gan-zoo

Data for Semi-Supervised Learning

1. Use small set e.g. 1.5% of Training images with Labels
2. Remaining training data (98.5%) will be used without Labels
3. Model will be validated using 10,000 images from Test Data



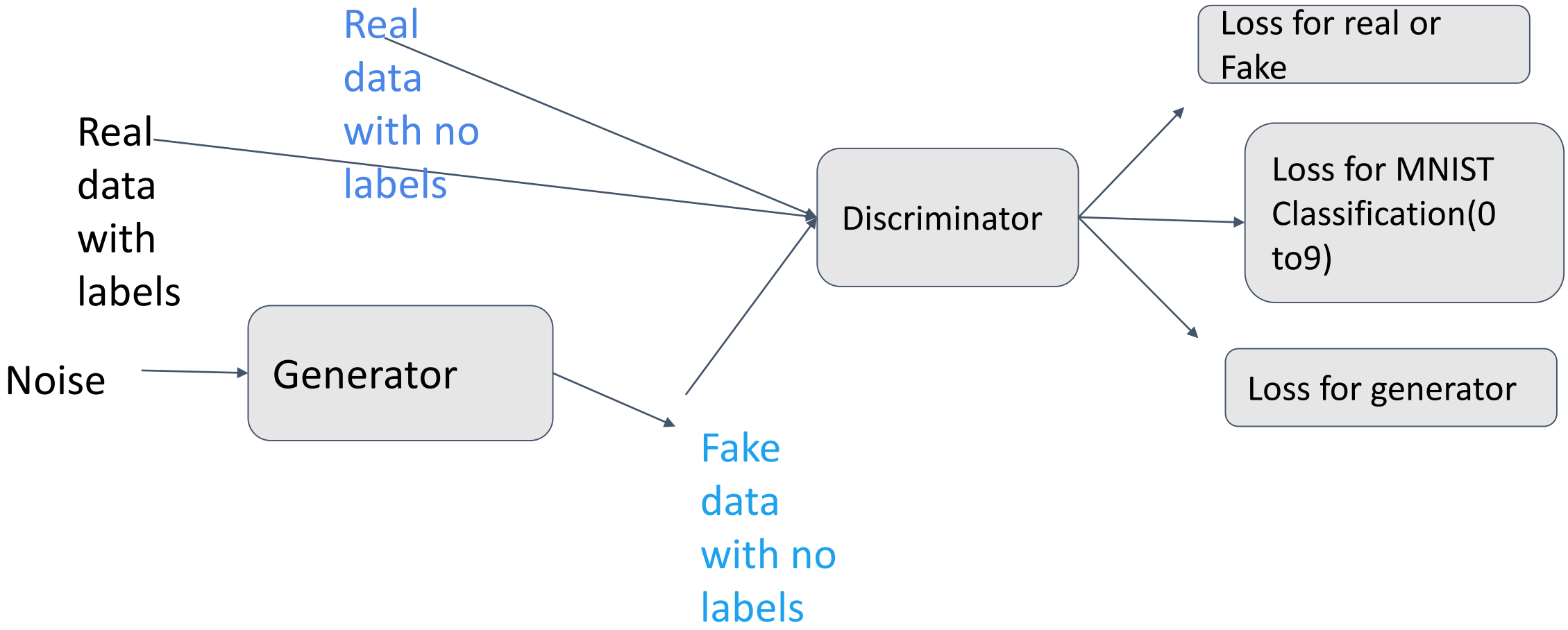
Semi-Supervised Learning with GANs



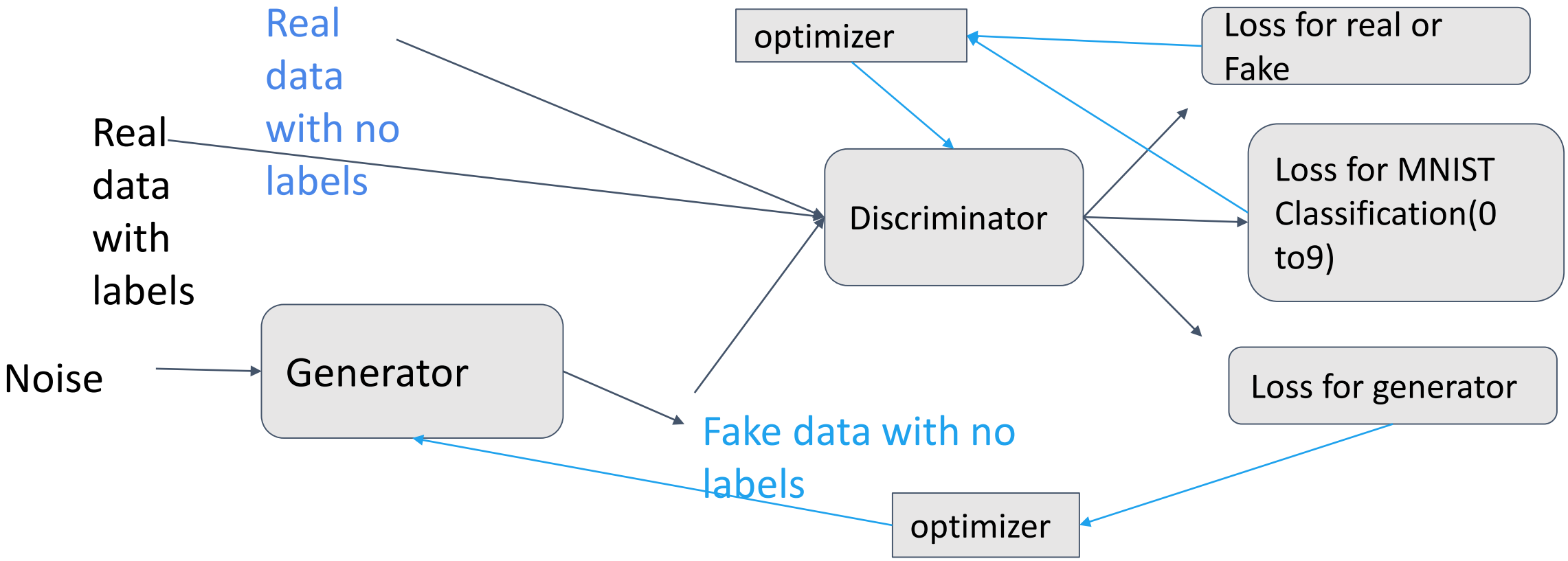
Multilabel Classification?

1. Real or Fake?
2. Probability of a number (0,1,2,3…..9)

How do we train?



How do we train?



Losses

1. Binary Classification (Real or Fake) - Binary Cross Entropy
2. Loss for MNIST Classification - Categorical Cross Entropy
3. Loss for Generator - Binary Cross Entropy

Semi-Supervised Learning in tensorflow

Notebook Practice

Additional Resources

GANs Playground - https://reiinakano.github.io/gan-playground/

GANs Comparison - https://github.com/khanrc/tf.gans-comparison

Summary

- The covered objectives of the GAN module are:
  1. The basic “Wh” questions of GAN –What, Why, Where and How?
  2. How learning happens in GAN
  3. Types of GANs
  4. Applications with one detailed case study
  5. Challenges in real world implementation of GANs
  6. Working code walk through

References set-1

https://jjallaire.github.io/deep-learning-with-r-notebooks/notebooks/8.5-introduction-to-gans.nb.html

https://medium.com/randomai/gan-for-medical-imaging-generating-images-and-annotations-8ad7c778809c

https://medium.com/randomai/gan-for-medical-imaging-generating-images-and-annotations-8ad7c778809c

https://towardsdatascience.com/gan-by-example-using-keras-on-tensorflow-backend-1a6d515a60d0

https://medium.com/apache-mxnet/transposed-convolutions-explained-with-ms-excel-52d13030c7e8

https://medium.com/@sharimpervez/a-simple-introduction-to-generative-adversarial-networks-gans-8e327ff65f7c

http://guimperarnau.com/blog/2017/03/Fantastic-GANs-and-where-to-find-them

https://www.topbots.com/advances-in-gans/

https://medium.com/@jonathan_hui/gan-progressive-growing-of-gans-f9e4f91edf33

https://medium.com/@jonathan_hui/gan-whats-generative-adversarial-networks-and-its-application-f39ed278ef09

References set-2

- Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, Alexei A. Efros, Image-to-Image Translation with Conditional Adversarial Networks (2016).
- Liqian Ma, Xu Jia, Qianru Sun, Bernt Schiele, Tinne Tuytelaars, Luc Van Gool, Pose Guided Person Image Generation (2017).
- Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, Dimitris Metaxas, StackGAN: Text to Photo-realistic Image Synthesis with Stacked Generative Adversarial Networks (2016).
- Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, Thomas S. Huang, Generative Image Inpainting with Contextual Attention (2018).
- Junting Pan, Cristian Canton Ferrer, Kevin McGuinness, Noel E. O'Connor, Jordi Torres, Elisa Sayrol, Xavier Giro-i-Nieto, SalGAN: Visual Saliency Prediction with Generative Adversarial Networks (2017).
- Mehdi Mirza, Simon Osindero, Conditional Generative Adversarial Nets (2014).
- Augustus Odena, Christopher Olah, Jonathon Shlens, Conditional Image Synthesis with Auxiliary Classifier GAN (2016).
- Alec Radford, Luke Metz, Soumith Chintala, Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks (2015).
- Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio, Generative Adversarial Networks (2014).
- Martin Arjovsky, Soumith Chintala, Léon Bottou, Wasserstein GAN (2017).
- Karol Kurach, Mario Lucic, Xiaohua Zhai, Marcin Michalski, Sylvain Gelly, The GAN Landscape: Losses, Architectures, Regularization, and Normalization (2018).
- Swaminathan Gurumurthy, Ravi Kiran Sarvadevabhatla, Venkatesh Babu Radhakrishnan, DeLiGAN : Generative Adversarial Networks for Diverse and Limited Data (2017).

Thank you