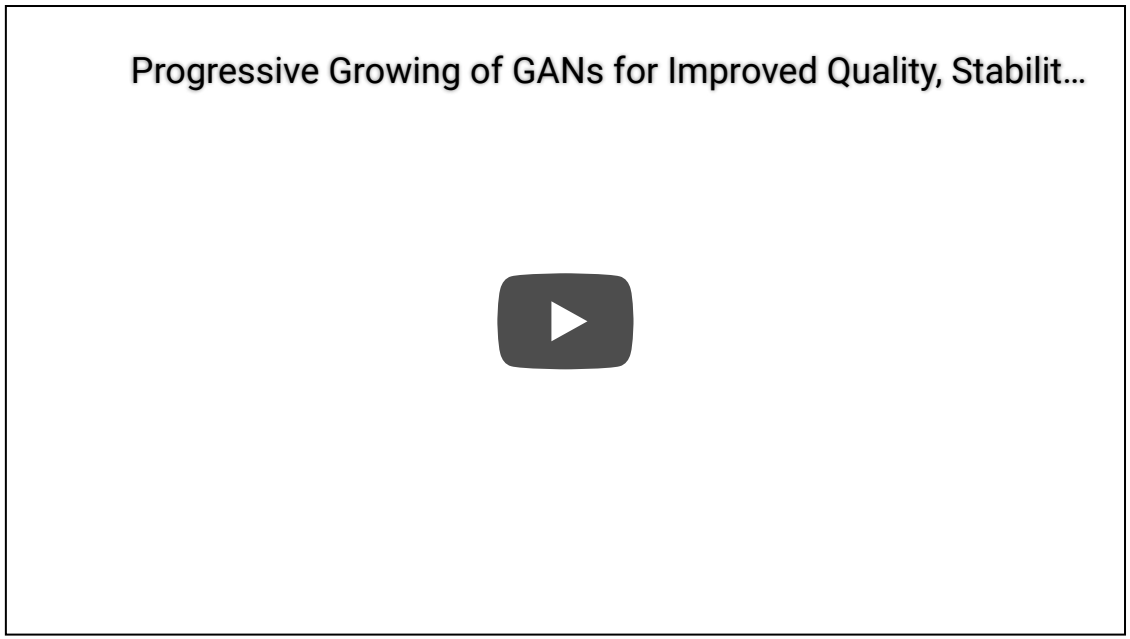
Session 3 & Assignment

Due No Due Date **Points** 5,000 **Submitting** a website url

Deadline extended: 18th August for everyone except those in EVA
Deadline for EVA students is 25th August.

GENERATIVE ADVERSARIAL NETWORKS



Adversarial: (of a trial or legal proceedings) in which the parties in a dispute have the responsibility for finding and presenting evidence

GANs are DNNs comprised of two nets, pitted one against the other (thus adversarial)

GAN Applications

- Generate Examples for Image Datasets
- Generate Photographs of Human Faces
- Generate Realistic Photographs
- Generate Cartoon Characters
- Image-to-Image Translation
- Text-to-Image Translation
- Semantic-Image-to-Photo Translation
- Face Frontal View Generation
- Generate New Human Poses
- Photos to Emojis
- Photograph Editing
- Face Aging
- Photo Blending

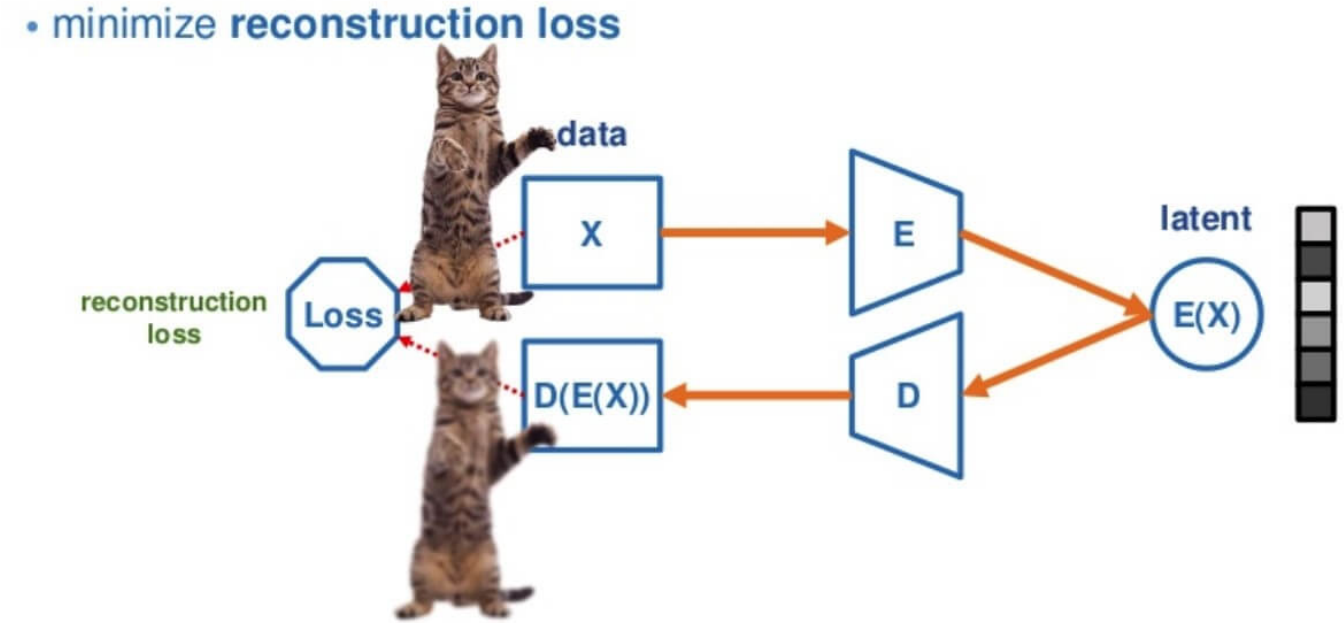
- Super Resolution
- Photo Inpainting
- Clothing Translation
- Video Prediction
- 3D Object Generation

Deep Generative Models

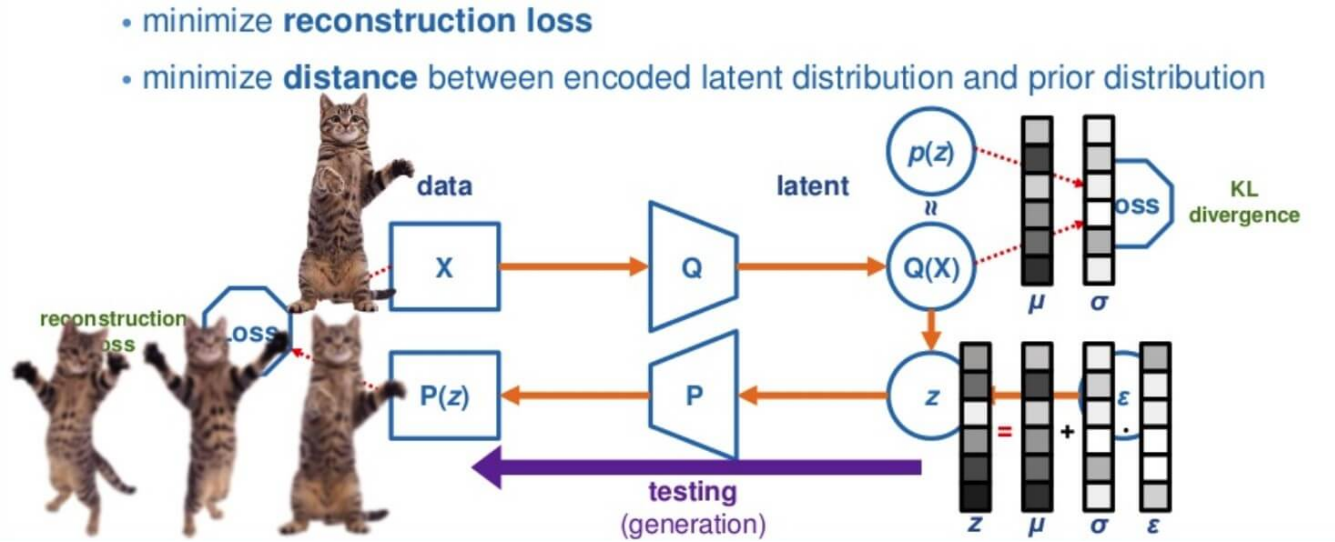
- AutoEncoders
- Variational AutoEncoders
- Ganerative Adversarial Networks
- Adversarial AutoEncoders
- VAE/GAN
- Adversatial Domain Adaption

Quick Look

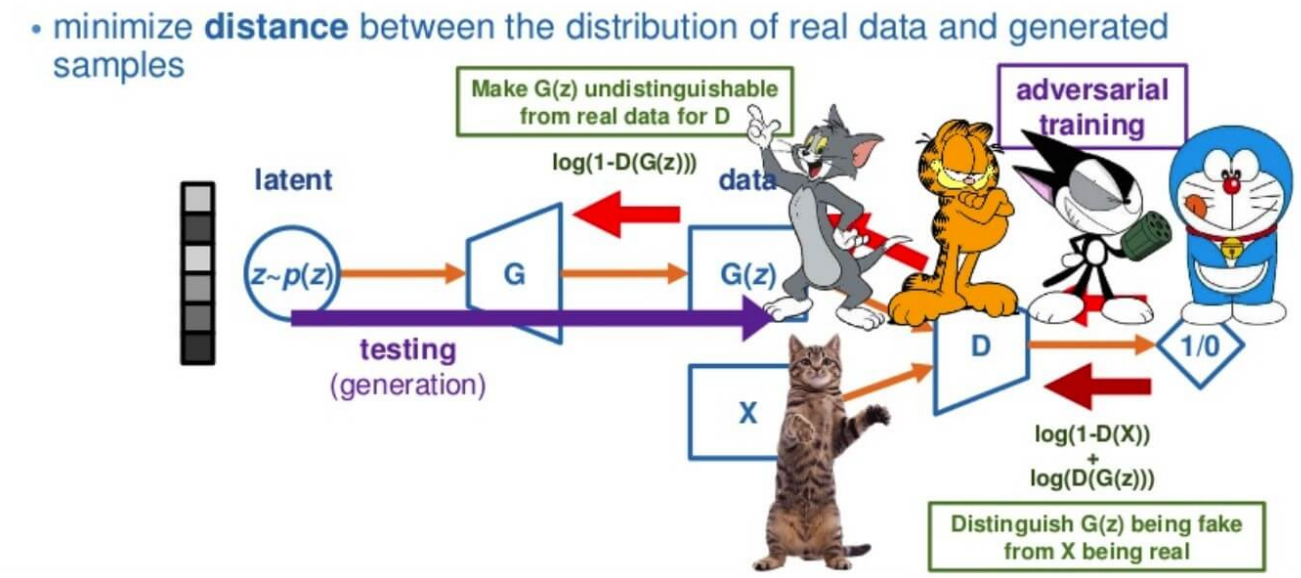
AUTOENCODERS



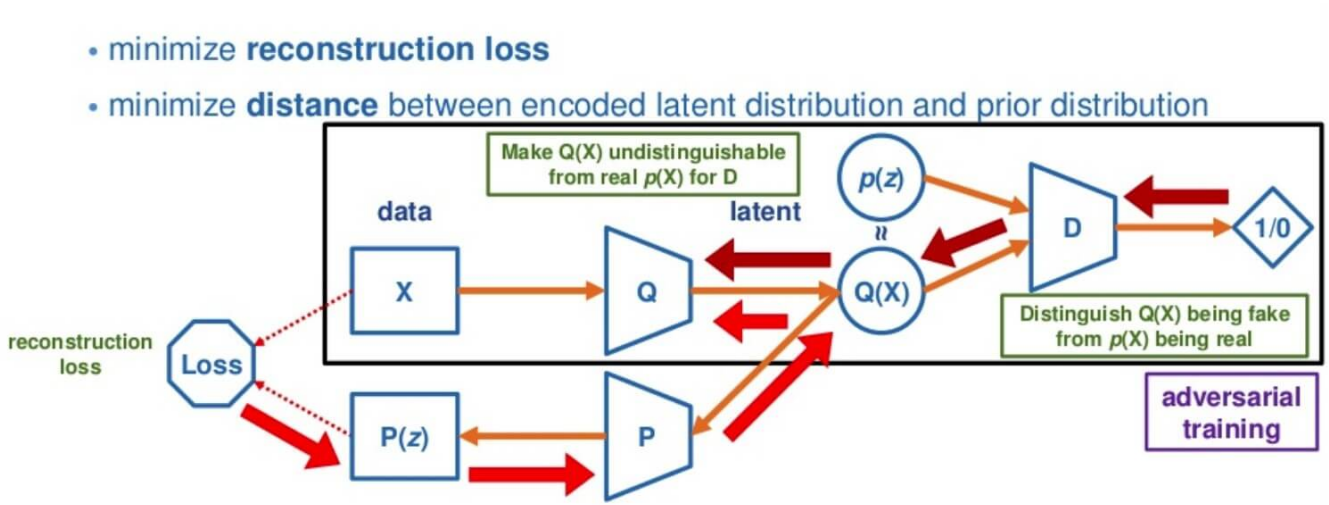
VARIATIONAL AUTOENCODERS


• minimize reconstruction loss
• minimize distance between encoded latent distribution and prior distribution
data
X
Q
latent
p(z)
Q(X)
KL divergence
μ
σ
reconstruction loss
P(z)
P
z
ε
testing
(generation)

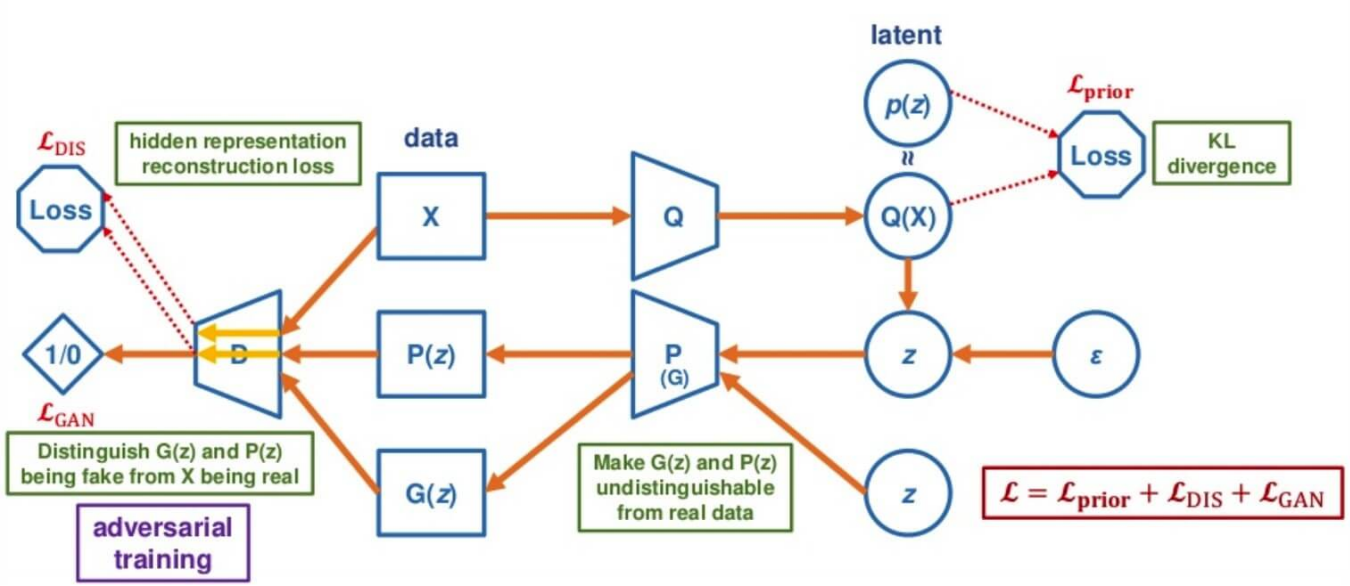
GANs



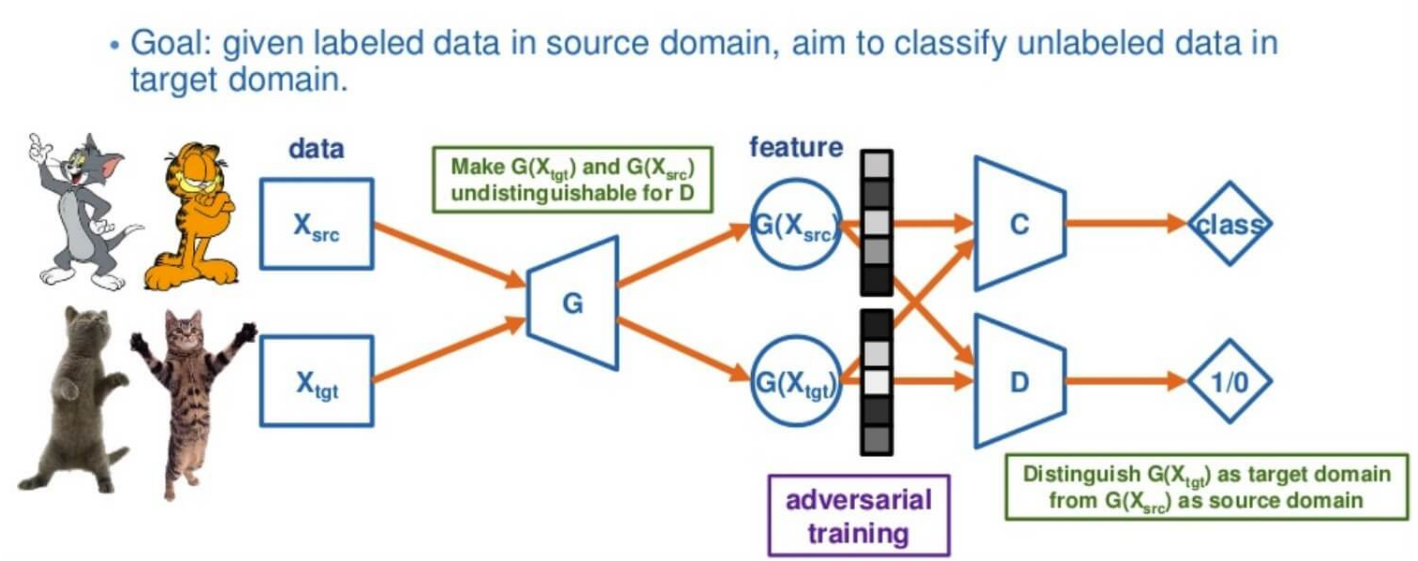
ADVERSARIAL AUTOENCODERS


• minimize **reconstruction loss**
• minimize **distance** between encoded latent distribution and prior distribution
Make Q(X) undistinguishable from real p(X) for D
data
latent
p(z)
D
1/0
X
Q
Q(X)
Distinguish Q(X) being fake from p(X) being real
reconstruction loss
Loss
P(z)
P
adversarial training

VAE-GAN



ADVERSARIAL DOMAIN ADAPTATION



GANs

GANs were introduced in a paper (https://arxiv.org/abs/1406.2661) by Ian Goodfellow and other researchers at the University of Montreal, including Yoshua Bengio, in 2014. Referring to GANs, Facebook's AI research director Yann LeCun **called adversarial training** (https://www.quora.com/What-are-some-recent-and-potentially-upcoming-breakthroughs-in-deep-learning) "the most interesting idea in the last 10 years in ML."

GANs' potential is huge, because they can learn to mimic any distribution of data. That is, GANs can be taught to create worlds eerily similar to our own in any domain: images, music, speech, prose. They are robot artists in a sense, and their **output is impressive (https://www.nytimes.com/2017/08/14/arts/design/google-how-ai-creates-new-music-and-new-artists-project-magenta.html)** – poignant even.

In a surreal turn, **Christie's sold a portrait (https://www.theverge.com/2018/10/23/18013190/ai-art-portrait-auction-christies-belamy-obvious-robbie-barrat-gans)** for $432,000 that had been generated by a GAN, based on **open-source code written by Robbie Barrat of Stanford (https://github.com/robbiebarrat/art-DCGAN)**. Like most true artists, he didn't see any of the money, which instead went to the French company, Obvious.

Generative vs. Discriminative Algorithms

To understand GANs, you should know how generative algorithms work, and for that, contrasting them with discriminative algorithms is instructive. Discriminative algorithms try to classify input data; that is, given the features of an instance of data, they predict a label or category to which that data belongs.

A Discriminative algorithm could predict whether a given image is Dog or a Cat, and it does that by learning the features that constitute the input image. So a DA maps features to labels. They are concerned solely with that correlation.

Generative algorithms, on the other hand, try to do the opposite. Instead of predicting a label given certain image, they attempt to predict the image given a certain label.

Another way to think about it is to distinguish discriminative from generative like this:

- Discriminative models learn the boundary between classes
- Generative models model the distribution of individual classes

How GANs work?

One neural network, called the generator, generates new data instances, while the other, the discriminator, evaluates them for authenticity; i.e. the discriminator decides whether each instance of data that it review belongs to the actual training dataset or now.

Let’s say we’re trying to do something more banal than mimic the Mona Lisa. We’re going to generate hand-written numerals like those found in the MNIST dataset, which is taken from the real world. The goal of the discriminator, when shown an instance from the true MNIST dataset, is to recognize those that are authentic.

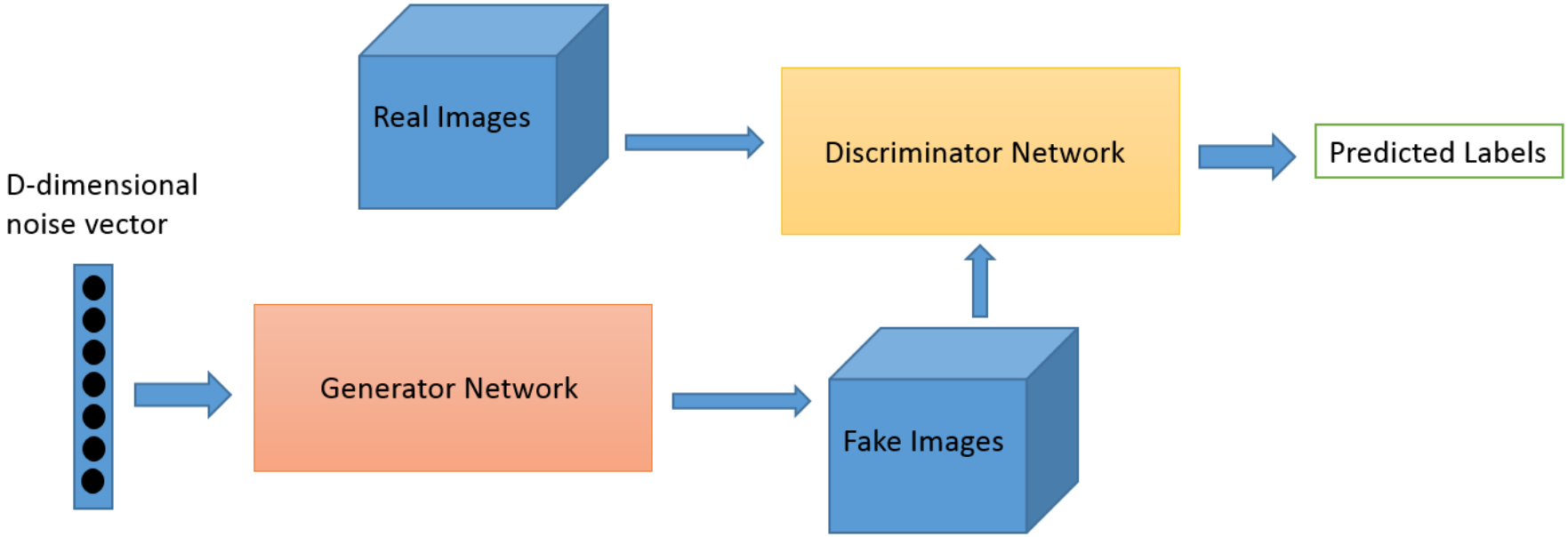
Meanwhile, the generator is creating new, synthetic images that it passes to the discriminator. It does so in the hopes that they, too, will be deemed authentic, even though they are fake. The goal of the generator is to generate passable hand-written digits: to lie without being caught. The goal of the discriminator is to identify images coming from the generator as fake.

Here are the steps a GAN takes:

- The generator takes in random numbers and returns an image
- This generated image is fed into the discriminator alonside a stream of iamges taken from the actual, ground-truth dataset
- The discrinimator takes in both real and fake images and returns values between 0 and 1, with 1 representing a prediction of authenticity and 0 representing a fake.

So you have a double feedback loop:

- The discriminator is in a feedback loop with the ground truth of the images, which we know.
- The generator is in a feedback loop with the discriminator.



You can think of a GAN as the opposition of a counterfeiter and a cop in a game of cat and mouse, where the counterfeiter is learning to pass false notes, and the cop is learning to detect them. Both are dynamic; i.e. the cop is in training, too (to extend the analogy, maybe the central bank is

flagging bills that slipped through), and each side comes to learn the other's methods in a constant escalation.

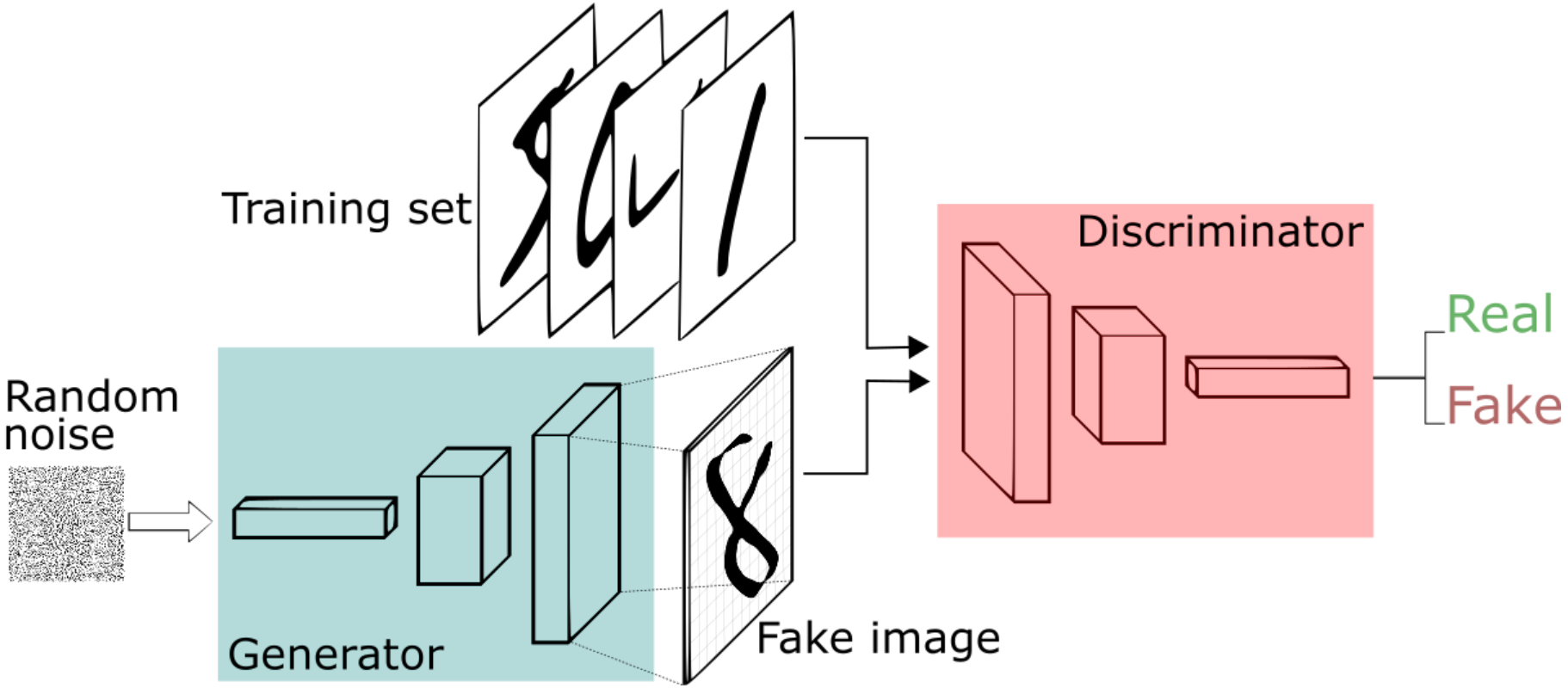
Let us consider MNIST

For MNIST, the discriminator network is a standard convolution network that can categorize the images fed to it, a binomial classifier labeling images as real or fake.

The generator is an **inverse convolution network**. While a standard convolution classifier takes an image and down-samples it to produce a soft-max number, the generator model takes a vector of random noise and up-samples it to an image.

The first throws away the data through down-sampling techniques like max-pooling, and the second generates new data.

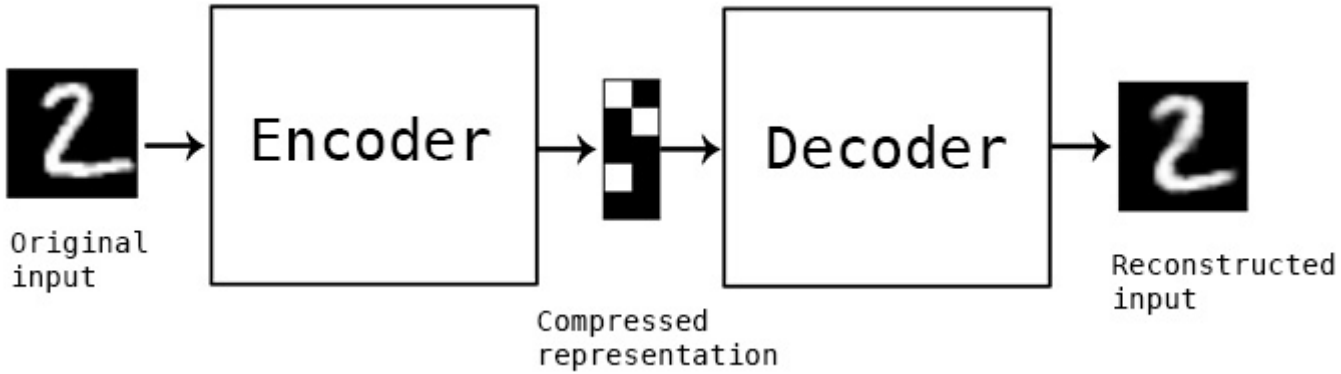
Both networks are trying to optimize a different and opposing objective function, or loss function, in a zero-sum game. This is essentially an actor-critic model. As the discriminator changes its behavior, so does the generator, and vice versa. Their losses push against each other.



GANs, Autoencoders and VAEs

It may be useful to compare generative adversarial networks to other neural networks, such as autoencoders and variational autoencoders.

Autoencoders encode input data as vectors. They create a hidden, or compressed representation of the raw data. They are useful in dimensionality reduction; that is, the vector serving as a hidden representation compress the raw data into a smaller number of salient dimensions. Autoencoders can be paired with a decoder, which allows you to reconstruct input data based on its hidden representation.

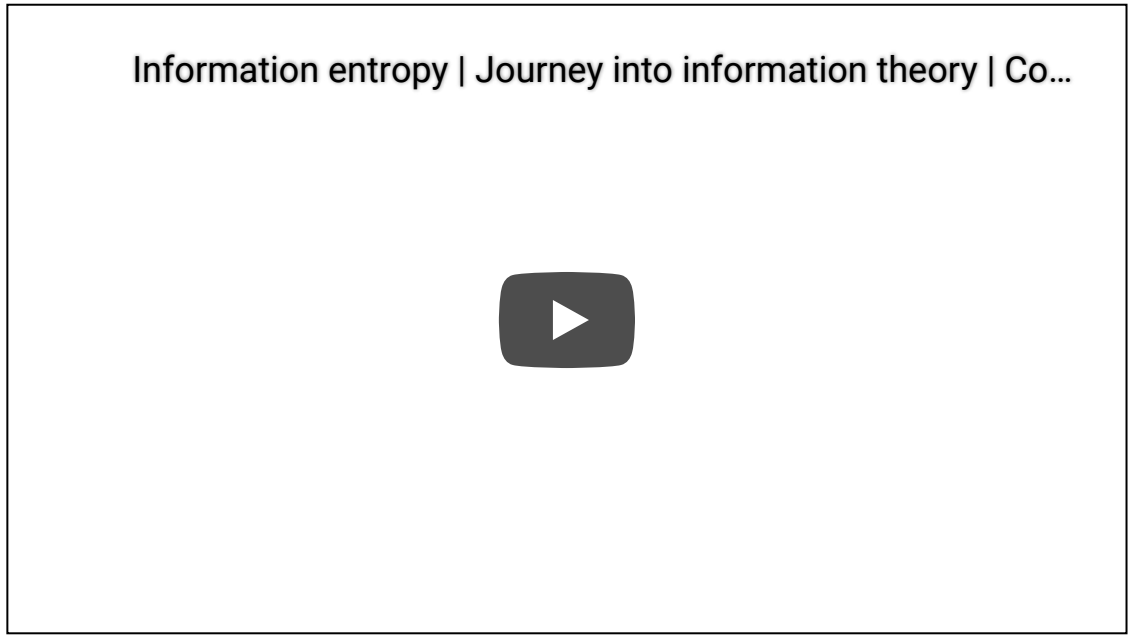


Variational Autoencoders are generative algorithms that add an additional constraint to encode the input data, namely that the hidden representations are normalized. VAEs are capable of both compressing data like an autoencoder and synthesizing data like a GAN. However, while GANs generate data in fine, granular details, images generated by VAEs tend to be more blurred (this statement changed on **9 June 2019 (https://arxiv.org/pdf/1906.00446.pdf)**).

Kullback Leibler Divergence

To understand VAEs further you need to understand what is KL Divergence

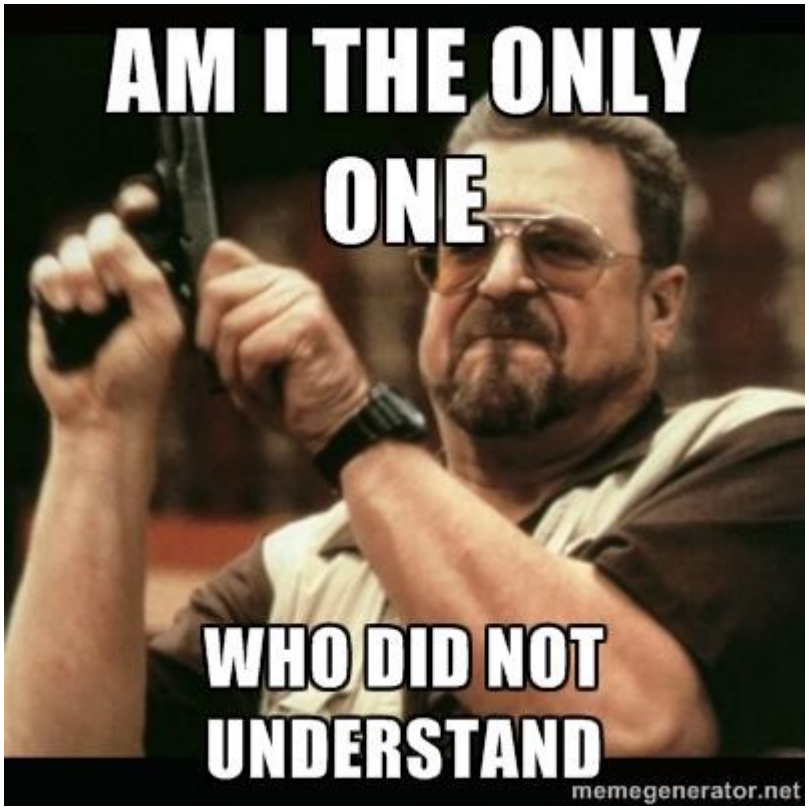
Imagine being asked with generating a model for p(x) and you end up creating a candidate model, **q(x)**. Now, how do you quantitively determine how good your model compared to **p(x)**? To give a qualitative answer, you need to understand KL Divergence, which we can't go much deeper into it given time we have, but we will cover few short videos to quickly understand the background concepts:



KL divergence (relative entropy)

Homework Video

Explaining the Kullback-Liebler divergence through secret c…


AM I THE ONLY ONE
WHO DID NOT UNDERSTAND
memegenerator.net

Coming back to GAN

When you train the discriminator, hold the generator values constant, and when you train the generator, hold the discriminator constant. You need to train the discriminator first. You also need to train the discriminator more then generator.

Each side of the GAN can overpower the other. If the discriminator is too good, it will return values so close to 0 or 1, that the generator will struggle to read the gradient. If the generator is too good, it will persistently exploit weakness in the discriminator that lead to false negatives. This may be me mitigated by the their respective learning rates.

GANs take a long time to train. On a single GPU a GAN might take hours, and on a single CPU more than a day.

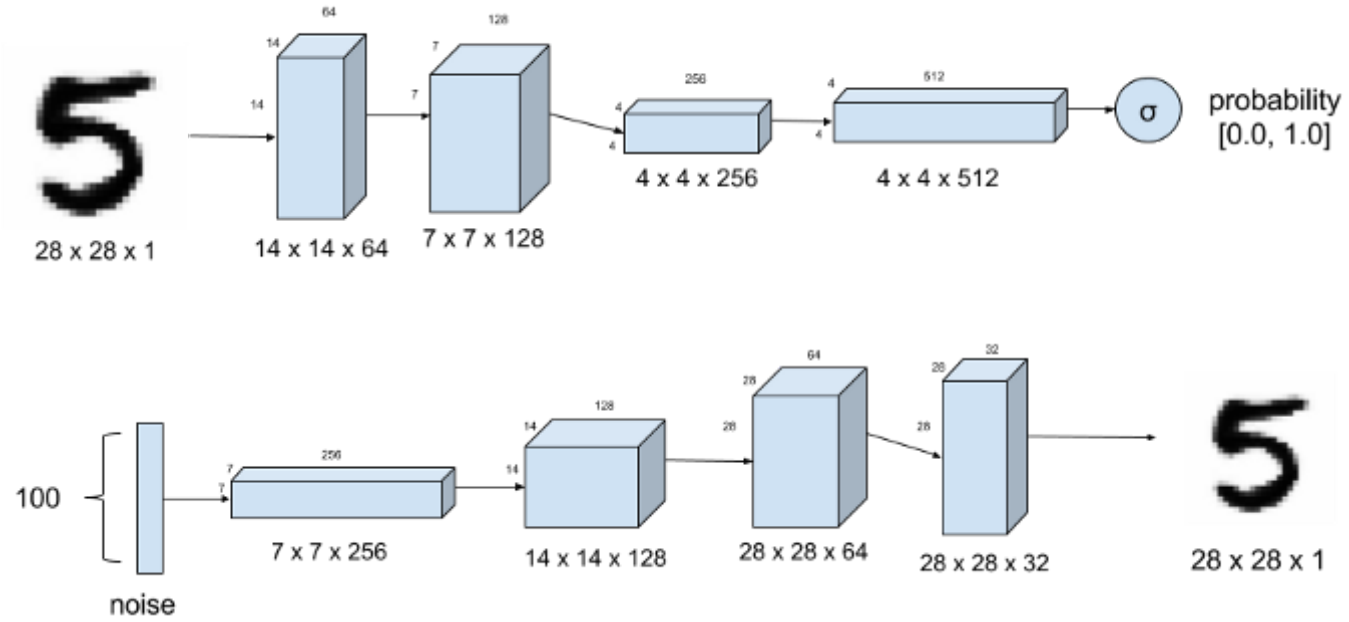
Let's understand GANs through a code:

CODE (https://github.com/eriklindernoren/Keras-GAN/blob/master/gan/gan.py)

TYPES OF GANs

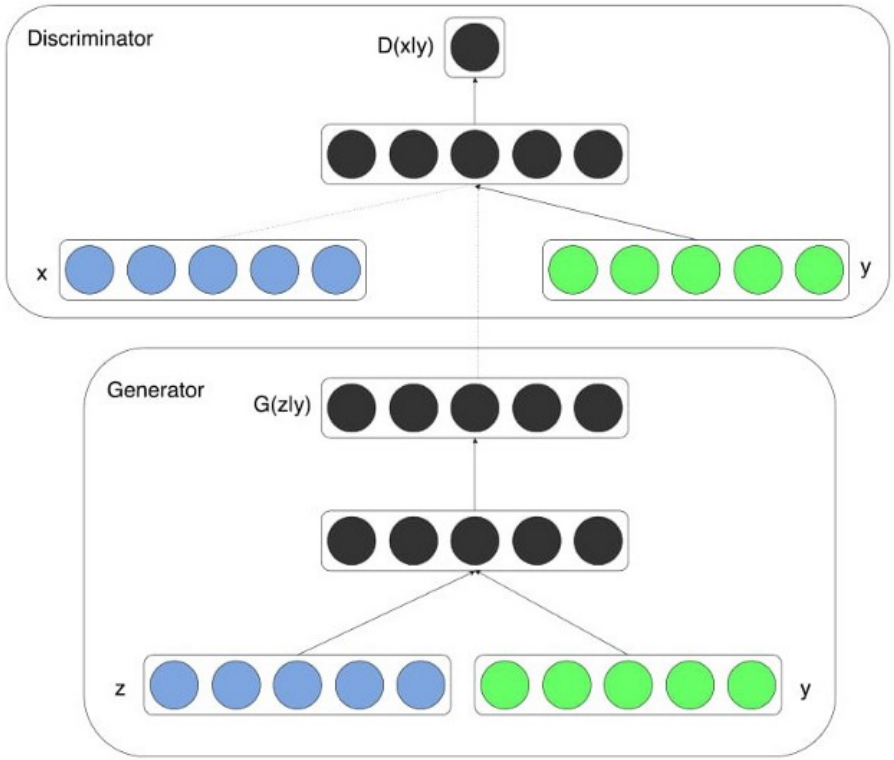
Deep Convolutional GANs (DCGAN)

The core idea is we use a convolutional neural network instead of vanilla neural network at both discriminator and generator



Conditional GANs

The core idea is to train a GAN with a conditioner, e.g. for MNIST data, we provide the label as well.



Auxiliary Classifier GAN (ACGAN)

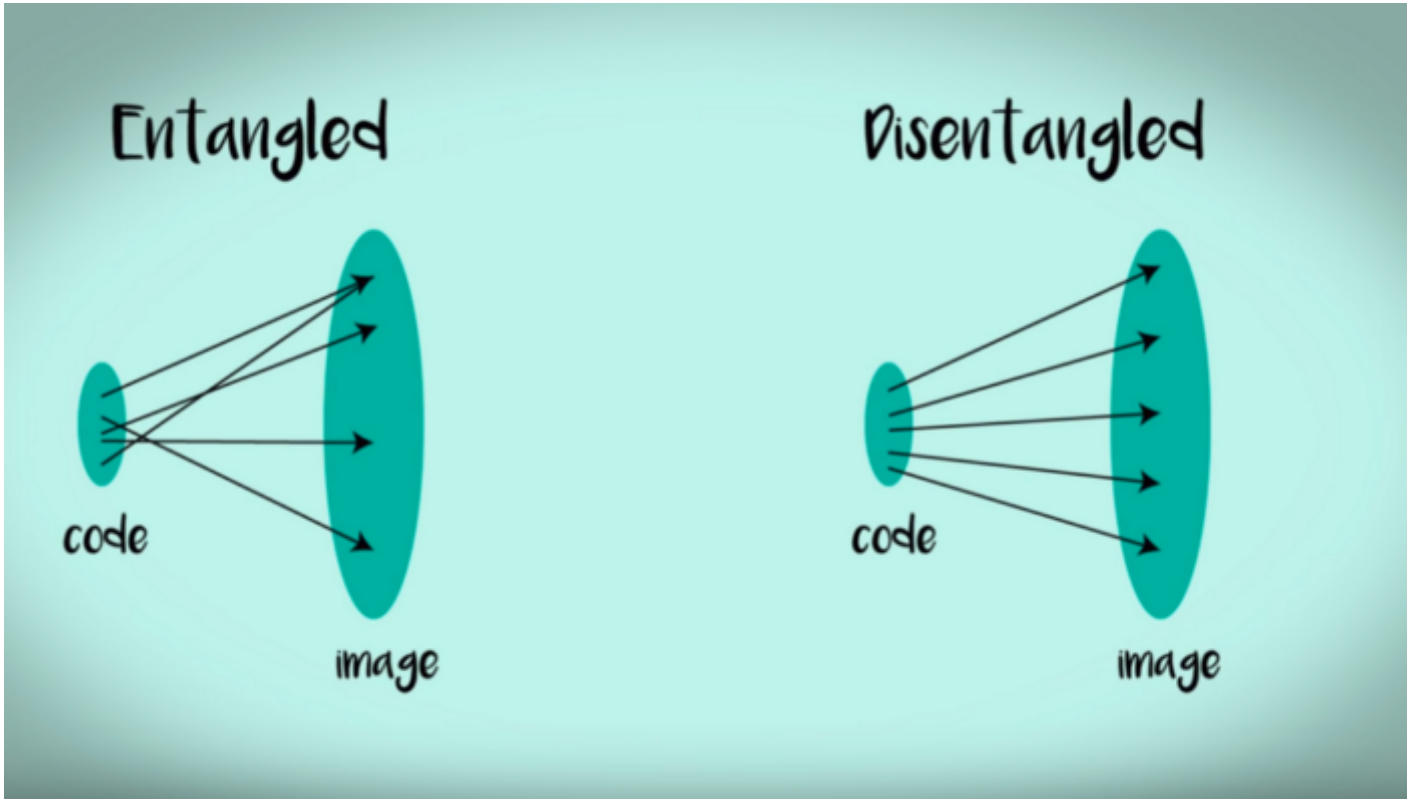
every generated sample has corresponding class label:

$$X_{fake} = G(c, z)$$

Think of ACGAN as CGAN except D is not told about the label, though additional network is added to give class probs at D.

infoGAN

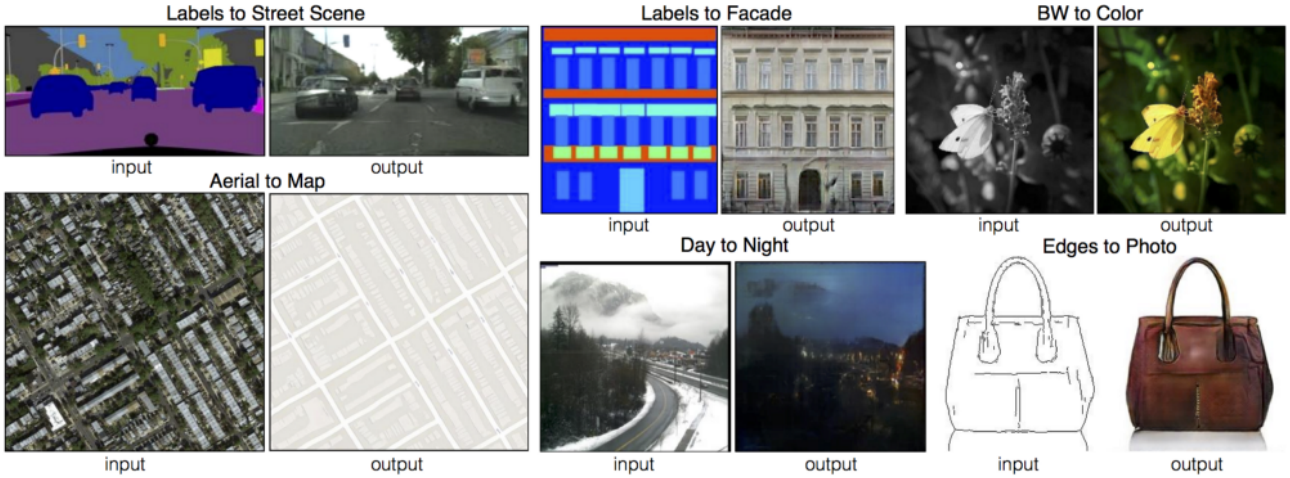
We seed GAN with random noise. We are not sure what is generating what. This is spooky. Canwe we change this?can we untangle this?

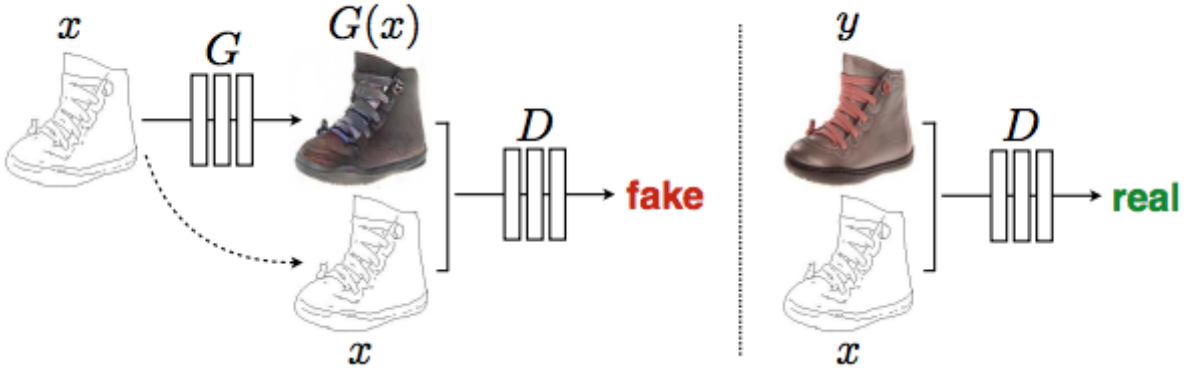


1. take a separate network called Q.
2. pick a random number (say x, like a label)
3. feed it to G, along with the noise
4. train D as you'd
5. feed the image generated by G and make Q network predict **x**


DO YOU ACTUALLY
UNDERSTAND WHATS
BEING DISCUSSED
memeshappen.com

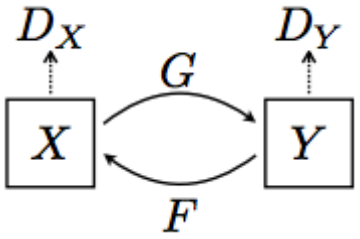
PixelGAN

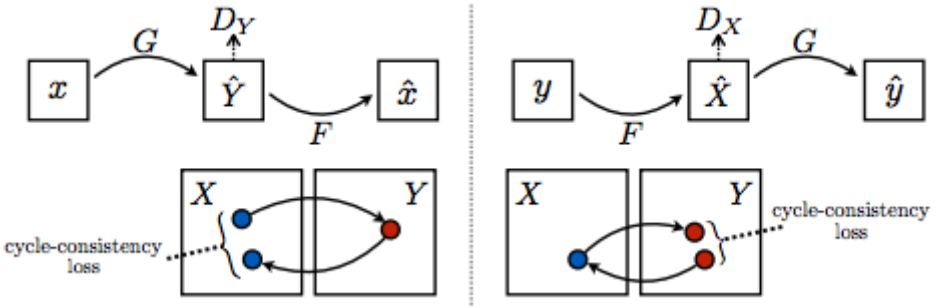




Basically feed both the image to both the networks.

CycleGAN


D_X
D_Y
G
X
Y
F


x
G
D_Y
$\hat{Y}$
$\hat{x}$
F
y
D_X
$\hat{X}$
$\hat{y}$
X
Y
cycle-consistency loss
cycle-consistency loss

1. Take the 2 images x and y (1 from Domain X and 1 from Domain Y)
2. Run the two generators (x2y and y2x) → generate 2 fake images(y',x')
3. Run the two discriminators (DX and DY) DX takes x and x' , DY takes y and y'
4. Calculate the discriminators losses from above equations
5. Run again the two generators, x2y (x' as the input) and y2x (y' is the input) generates two cycle images (y_cycle, and x_cycle)
6. Calculate the cycle l1 loss from (x,x_cycle and y,y_cycle)
7. Finally calculate the generator loss

Reference:

https://deephunt.in/the-gan-zoo-79597dc8c347 (https://deephunt.in/the-gan-zoo-79597dc8c347)

PHASE III PROJECT:

Yes, we are directly moving to Classifier Project:

Project Description:

1. Pick Either
    1. **Conditional Adversarial AutoEncoder Model (https://github.com/ZZUTK/Face-Aging-CAAE)**
    2. **Generative Model (https://github.com/sungnam0/Face-Aging-with-CycleGAN)**
2. Move to Keras

3. Minimum submission threshold is to reproduce the similar results
4. Free to use any dataset
5. Free to use any machine
6. Free to add any improvements
7. Result is highly dependent on the quality of reproduction and additional results.
8. Move the project to private repo on GitHub (dont forget to add us as collaborator)
9. Submit

VIDEO: