# GENERATIVE ADVERSARIAL NETWORKS

SureStart

# GENERATIVE MODELS

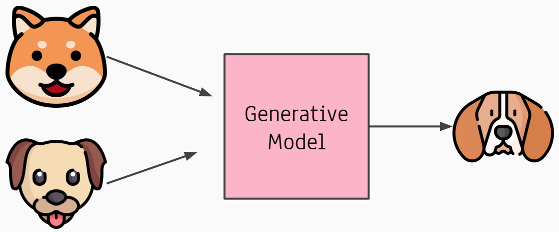
**Generative** Adversarial Network

a class of statistical models that contrasts with *discriminative* models.

**Generative** models can generate new data instances.

**Discriminative** models discriminate between different kinds of data instances.

- A generative model for images might capture correlations like "things that look like boats are probably going to appear near things that look like water" and "eyes are unlikely to appear on foreheads."
- In contrast, a discriminative model might learn the difference between "sailboat" or "not sailboat" by just looking for a few tell-tale patterns. It could ignore many of the correlations that the generative model must get right.




# GENERATIVE ADVERSARIAL NETWORK STRUCTURE

A generative adversarial network (GAN) has two parts:

- The generator learns to generate plausible data. The generated instances become negative training examples for the discriminator.
- The discriminator learns to distinguish the generator's fake data from real data. The discriminator penalizes the generator for producing implausible results.

When training begins, the generator produces obviously **fake data**, and the discriminator quickly learns to **tell that it's fake**.

Generated Data | Real Data | Discriminator Verdict

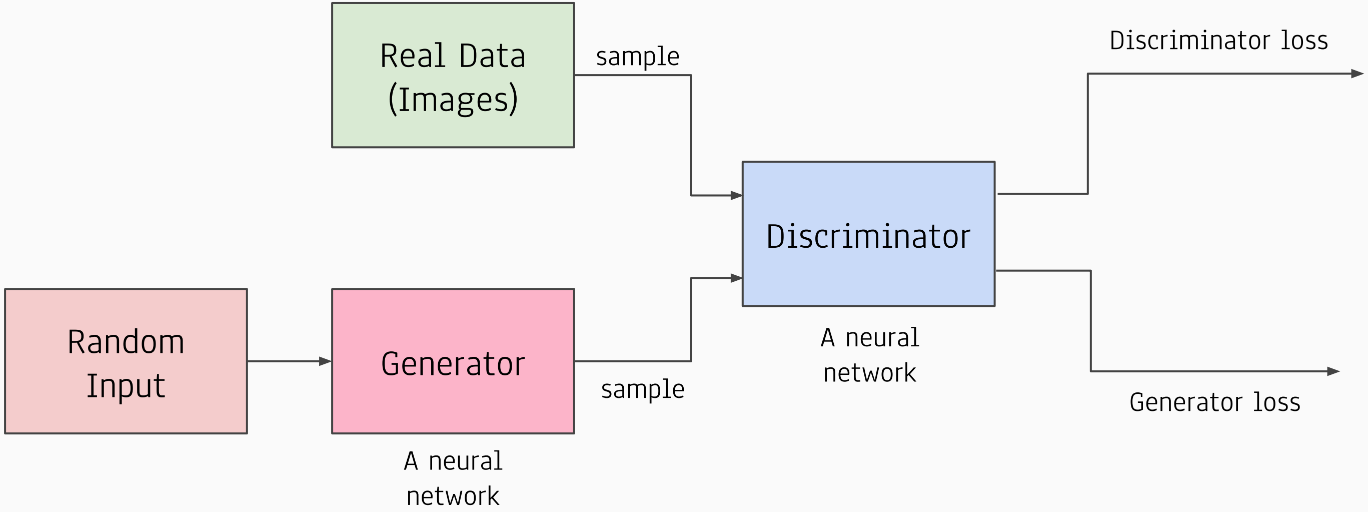
Fake

Fake

Real

SureStart

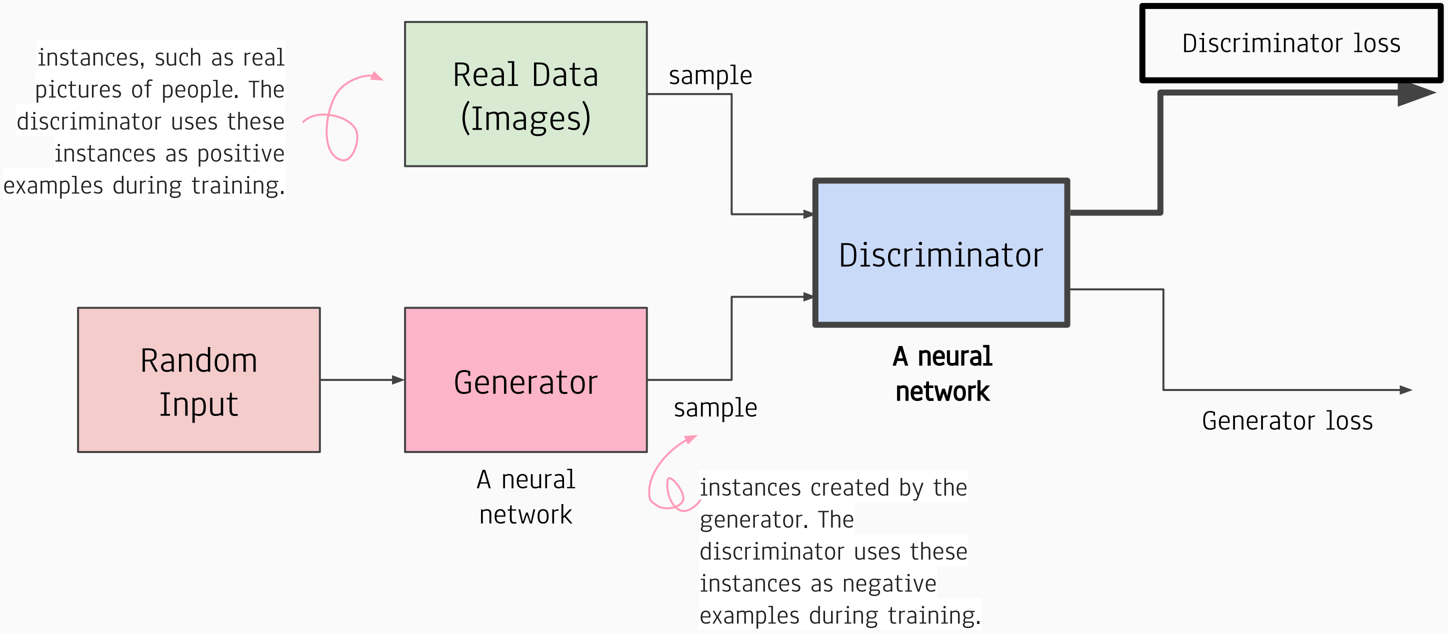
# GAN STRUCTURE




The generator output is connected directly to the discriminator input. Through backpropagation, the discriminator's classification provides a signal that the generator uses to update its weights.

# DISCRIMINATOR

The discriminator in a GAN is simply a classifier. It tries to distinguish real data from the data created by the generator. It could use any network architecture appropriate to the type of data it's classifying.



# DISCRIMINATOR TRAINING

- During discriminator training the generator does not train.
- Its weights remain constant while it produces examples for the discriminator to train on.

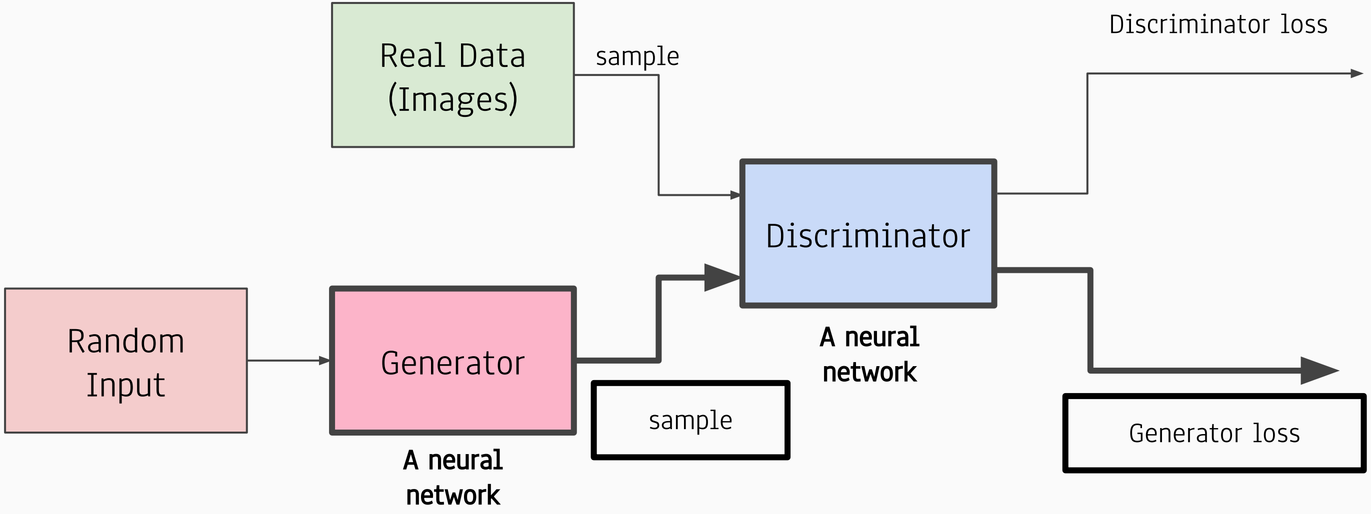
The discriminator connects to two loss functions. During discriminator training, the discriminator ignores the generator loss and just uses the discriminator loss. We use the generator loss during generator training.

During discriminator training:

1. The discriminator classifies both real data and fake data from the generator.
2. The discriminator loss penalizes the discriminator for misclassifying a real instance as fake or a fake instance as real.
3. The discriminator updates its weights through backpropagation from the discriminator loss through the discriminator network.

# GENERATOR

The generator part of a GAN learns to create fake data by incorporating feedback from the discriminator.
It learns to make the discriminator classify its output as real.




SureStart

# GENERATOR TRAINING

Generator training requires tighter integration between the generator and the discriminator than discriminator training requires. The portion of the GAN that trains the generator includes:

- random input
- generator network, which transforms the random input into a data instance
- discriminator network, which classifies the generated data
- discriminator output
- generator loss, which penalizes the generator for failing to fool the discriminator

In its most basic form, a GAN takes **random noise** as its input.
The generator then transforms this noise into a meaningful output.
By introducing noise, we can get the GAN to produce a wide variety of data, sampling from different places in the target distribution.

The generator feeds into the discriminator net, and the discriminator produces the output we're trying to affect.
The generator loss penalizes the generator for producing a sample that the discriminator network classifies as fake.

SureStart

# GENERATOR TRAINING

We train the generator with the following procedure:

1. Sample random noise.
2. Produce generator output from sampled random noise.
3. Get discriminator "Real" or "Fake" classification for generator output.
4. Calculate loss from discriminator classification.
5. Backpropagate through both the discriminator and generator to obtain gradients.
6. Use gradients to change only the generator weights.

This is one iteration of generator training.

Because a GAN contains two separately trained networks, its training algorithm must address two complications:

- GANs must juggle two different kinds of training (generator and discriminator).
- GAN convergence is hard to identify.

We keep the generator constant during the discriminator training phase. As discriminator training tries to figure out how to distinguish real data from fake, it has to learn how to recognize the generator's flaws.
Similarly, we keep the discriminator constant during the generator training phase. Otherwise the generator would be trying to hit a moving target and might never converge.

# LOSS FUNCTIONS

A GAN can have two loss functions: one for generator training and one for discriminator training.

Minimax loss: the generator tries to minimize the following function while the discriminator tries to maximize it

$$E_x[log(D(x))] + E_z[log(1 - D(G(z)))]$$

the expected value over all real data instances

the discriminator's estimate of the probability that real data instance x is real

the expected value over all random inputs to the generator

is the discriminator's estimate of the probability that a fake instance is real

the generator's output when given noise z

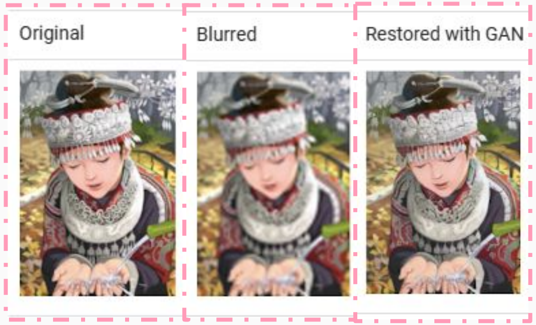
The formula derives from the cross-entropy between the real and generated distributions.

SureStart

# APPLICATIONS


Image to image translation

Image transformation




Super Resolution

"This flower has petals that are yellow with shades of orange."

Text to image synthesis