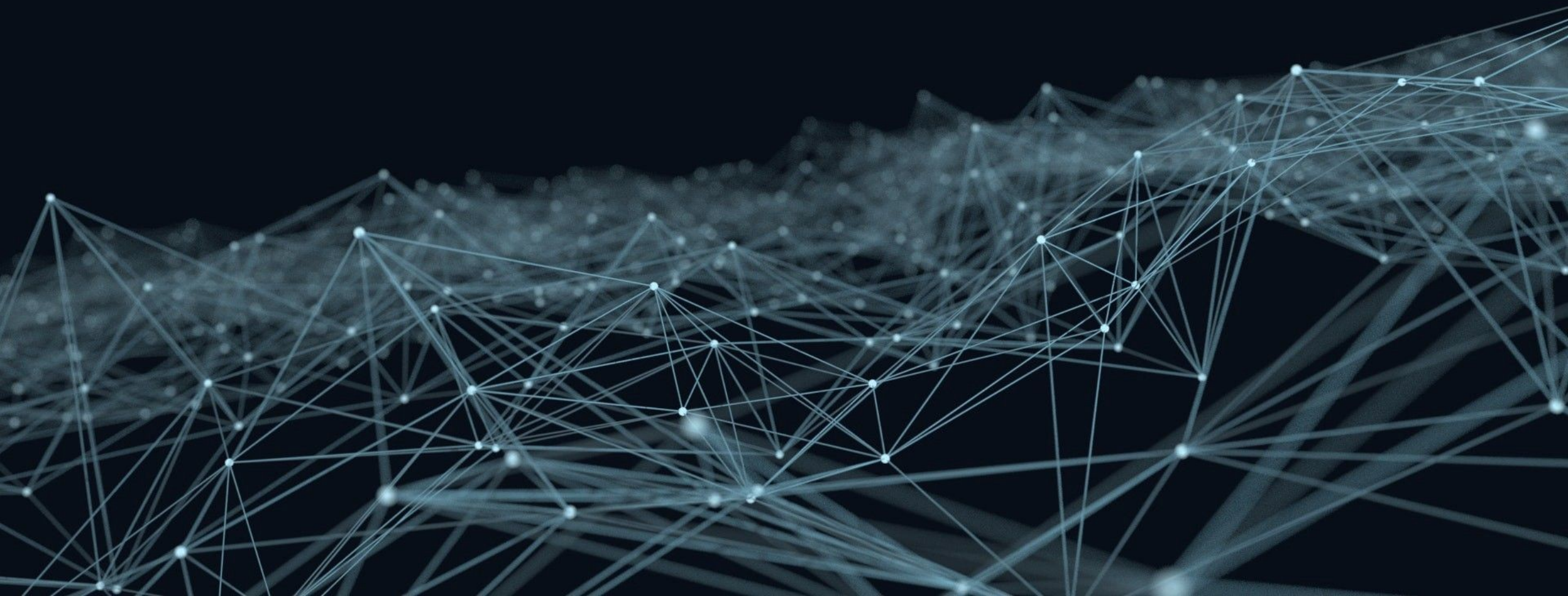
Generative Artificial Intelligence

Module Four: Adversarial Generative Networks



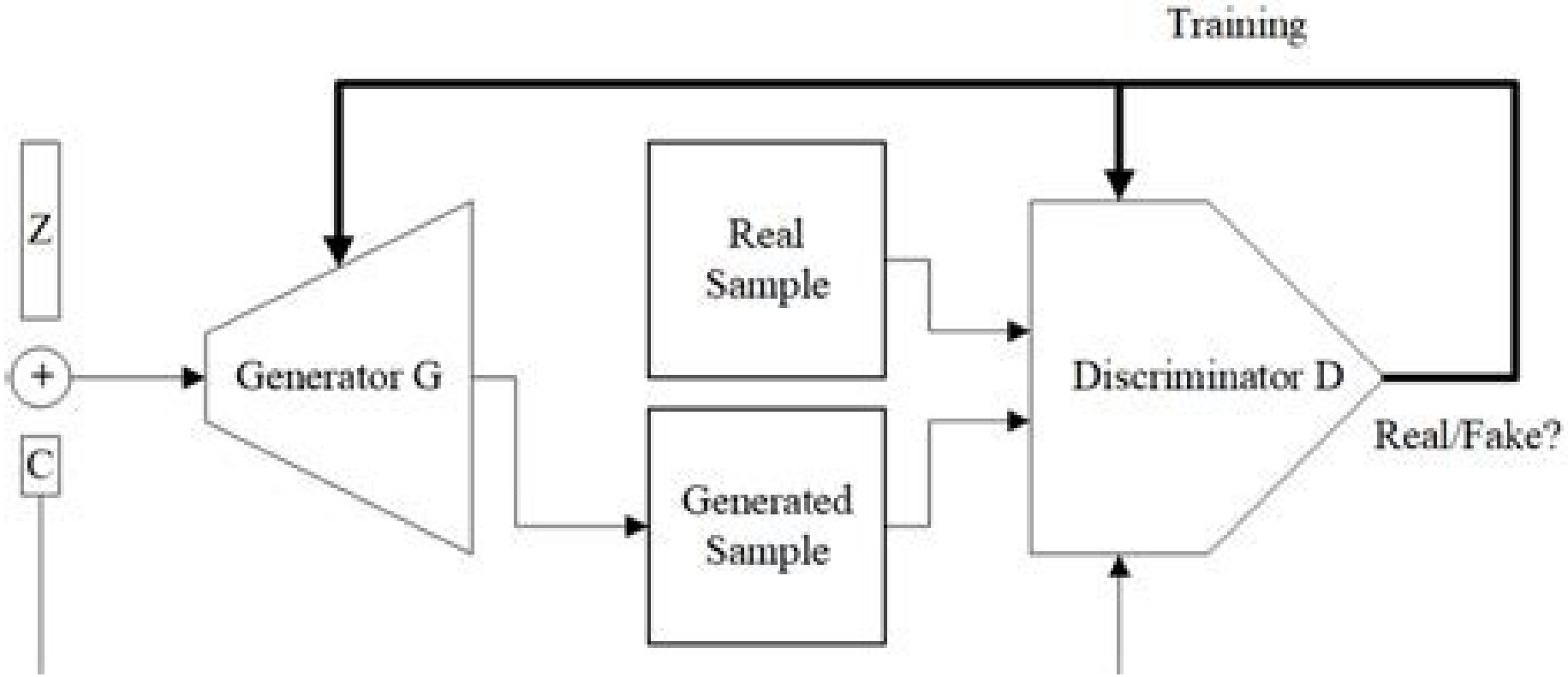
Generative AI Problems

- In all of the generative models we have looked at so far
  - There is always the problem of unrealistic output

Generative Adversarial Network

- Two neural nets “fighting it out”
  - The generator NN produces images
  - The discriminator NN evaluates the generated images as fake or real
  - The discriminator provides feedback to the generator as to how well it is doing
  - Think of this as a form of supervised learning
- Consier an art forger creating fakes of masterpieces
  - The forger is the generator
  - Each forgery is examined by an art expert (the discriminator)
  - The expert declares each forgery to be real or fake and tell the forger where he failed
  - The feedback from the expert is used to create better forgeries

Generative Adversarial AI



Generative Adversarial Network

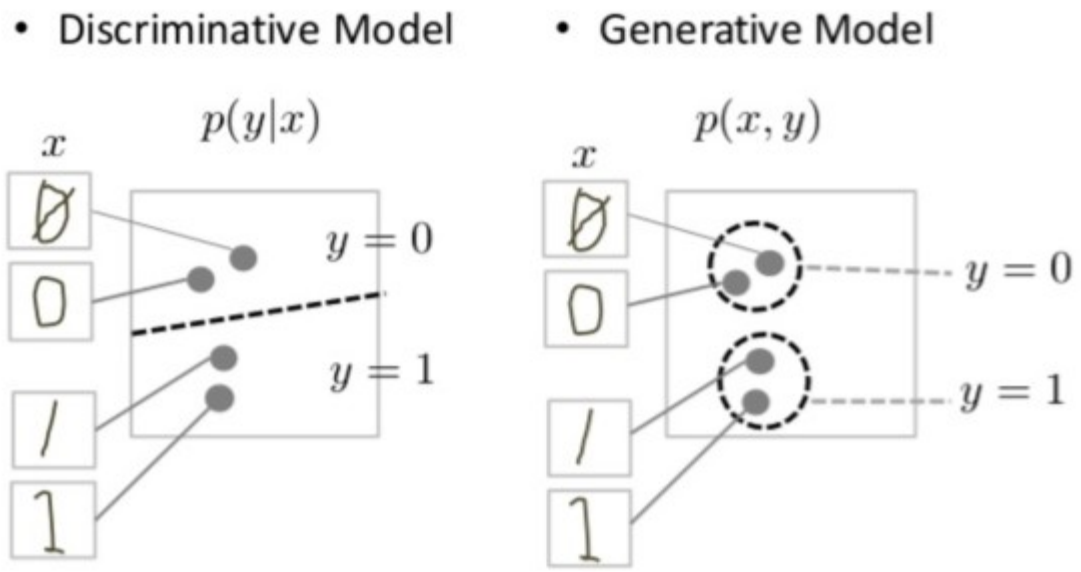
- The GAN is an implicit generative network that
  - Appears to learns to model the distribution of the data
  - Learns the density distributions of the data
  - Develops meta-representations of lower level latent features
  - This allows predictive capability so that gaps in the generation can be filled in realistically
  - For example, when building a face, it learns that the the presence of eyes can predict the presence of a nose
- Producing realistic output is important for synthetic data
  - Data that can be used in real world model training, like cancer tumor predicition based on tumor photographs

Generative Adversarial Network

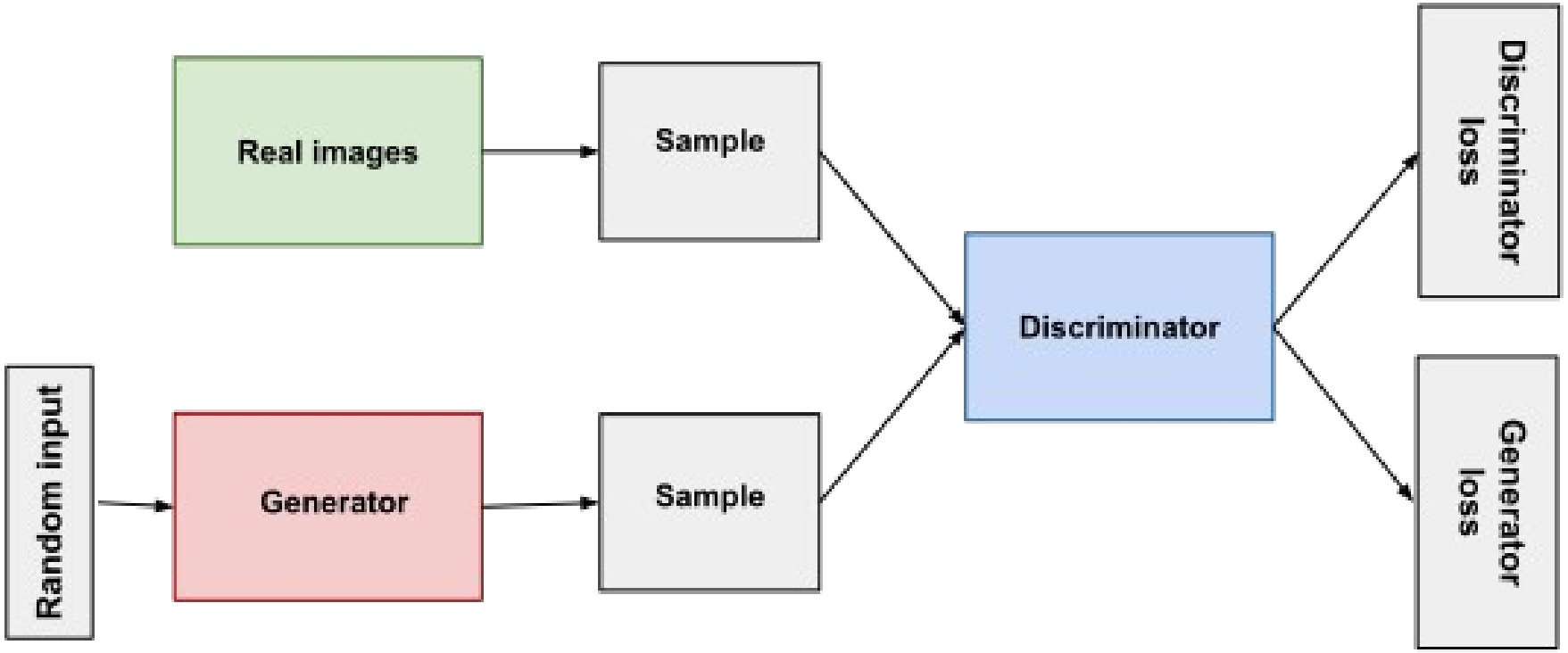
- What do the probabilities mean?
  - Generative models know the probability distribution of the data
  - We assume it’s not random but reflects underlying real world relationships and clusters
  - The model “knows” how likely a given sample is
  - For example, what the missing piece of an image is or what the next work in a sentence is
  - The discriminative model just reports on how likely the data is to have the proposed label
- The generative distributions are very complex along billions of features
  - For example, “cars are likely to appear on or near roads” and “snakes are unlikely to have arms and legs”

Example

- Handwritten digits
  - The discriminative model only has to separate the data into categories with some sort of classifier hyperplane or other separator
  - The generative model has to look at the center of mass of the distributions of all data points that cluster together around a digit (all the different version of a “4” for example)
  - Then the generative model can produce something with a high probability of being classified as a “4”

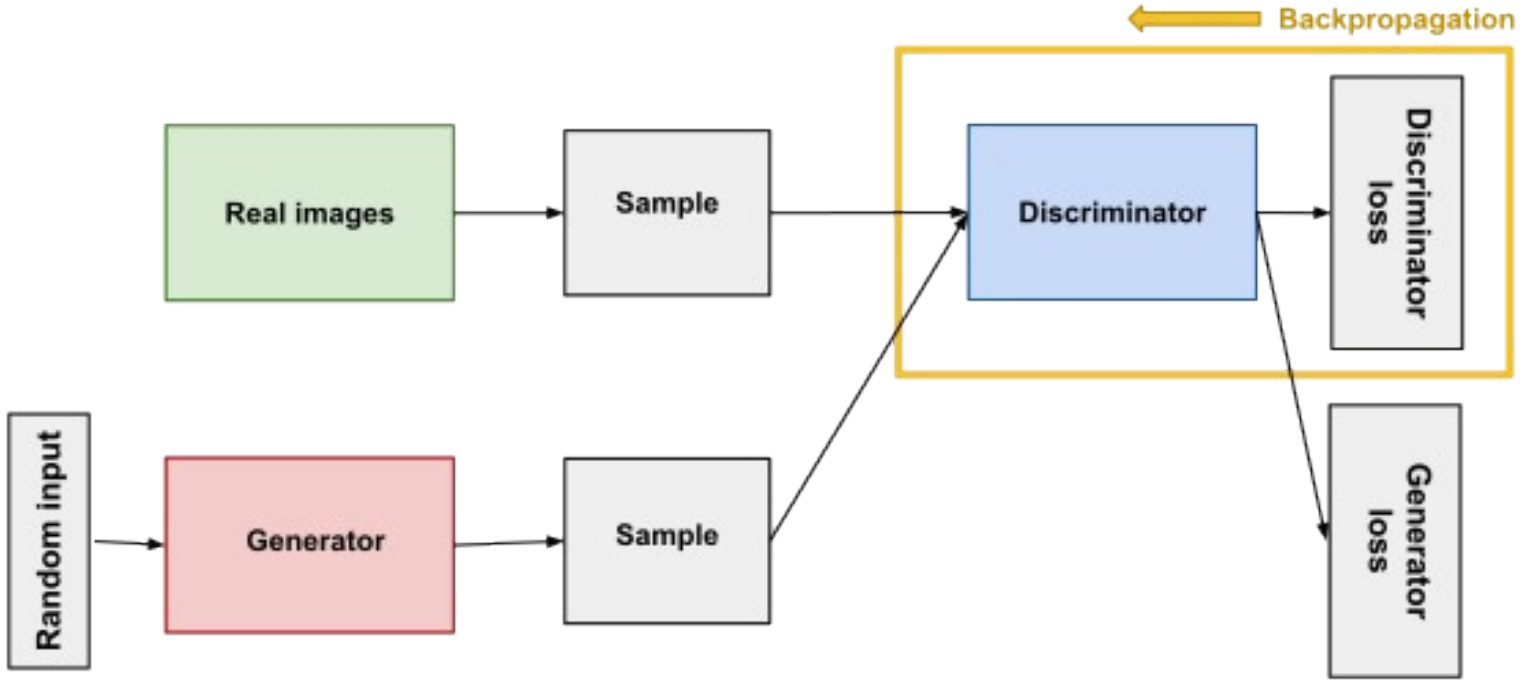


GAN for Images



The Discriminator

- The architecture of the discriminator depends on the type of data being generated
  - If we are working with images for example, we probably will use some sort of convolutional Neural Net

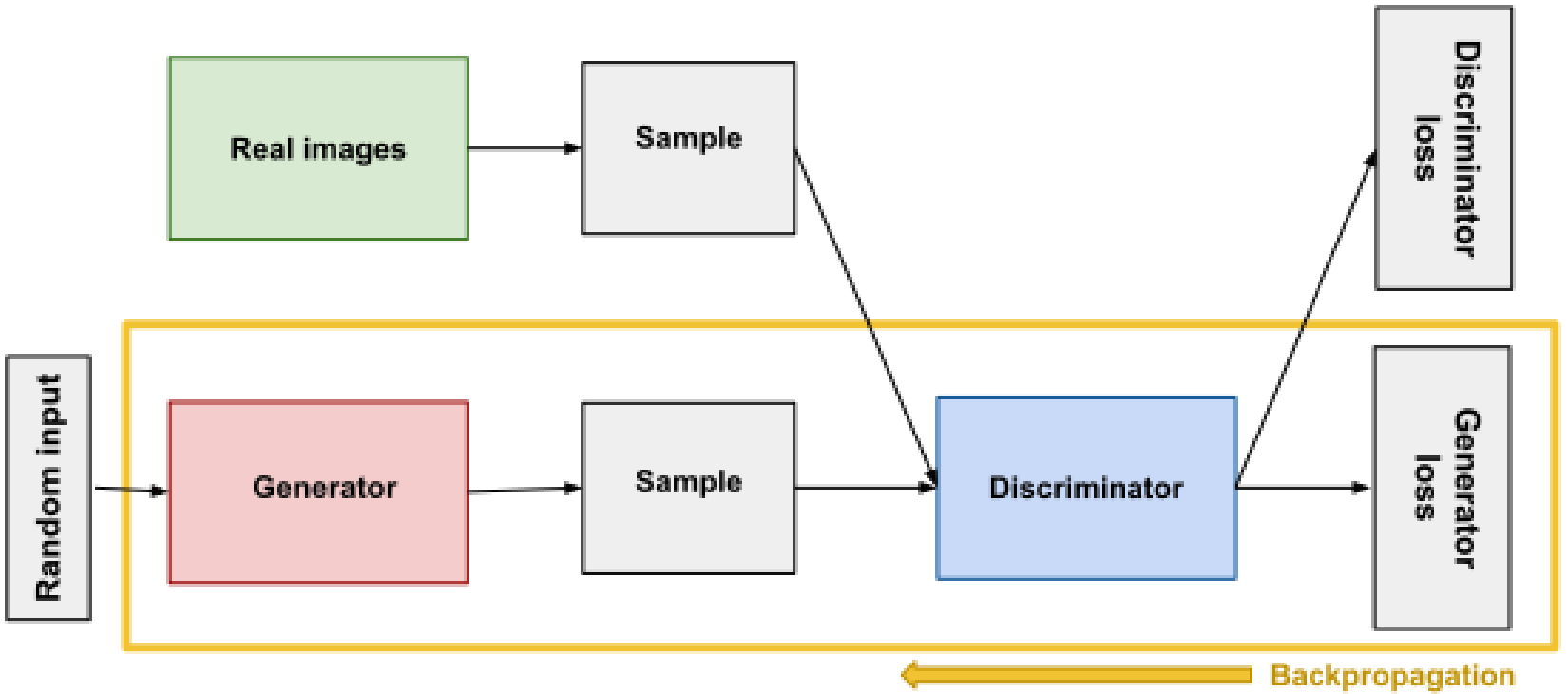


Training the Discriminator

- Use two sources of data for training
  - Positive image data sampled from the domain which should be classified as “real”
  - Negative image data created by the generator which should be classified as “fake”
- To train the discriminator:
  - Real data and fake data from the generator are used to train the classifier
  - Discriminator loss measures the amount the discriminator classified incorrectly
  - Backpropagation is used to reduce the discriminator loss through the discriminator network

The Generator

- The generator learns by using feedback from the discriminator
  - The learning goal to make the discriminator classify its output as real



Training the Generator

- The following is used to train the generator
  - 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
- Training the generator requires a coupling with the discriminator
  - This produces some complications
  - GANs must juggle two different kinds of training (generator and discriminator)
  - GAN convergence is problematic

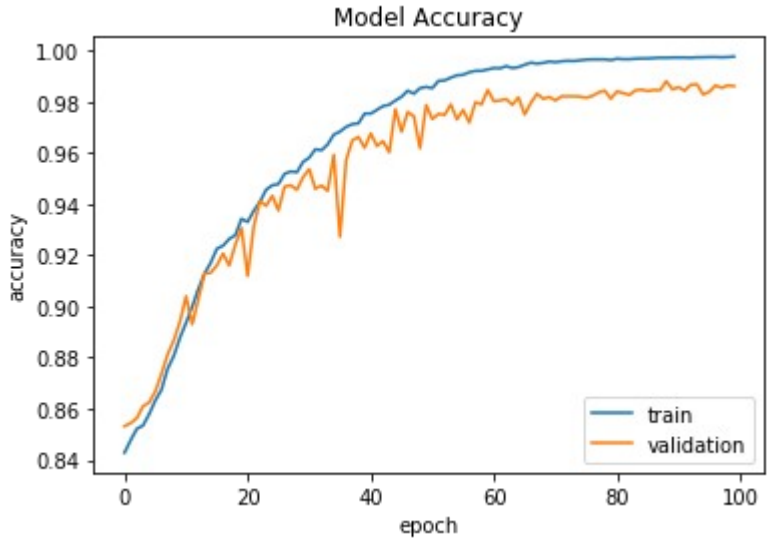
Training the Generator

- The following is used to train the generator
  - 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
- Training the generator requires a coupling with the discriminator
  - This produces some complications
  - GANs must juggle two different kinds of training (generator and discriminator)
  - GAN convergence is problematic

Training the GAN

- Convergence in a neural network is defined as:

  “In neural networks, convergence is defined based on the values of the training loss function: if the values of the training loss function follow a descending trend (assuming we are minimizing the loss function, which is the standard), then it converges. If they follow an ascending trend or they are oscillating, then the training is not converging.”



Training the Generator

- When training the generator
  - The generator output is not directly connected to the loss that we're trying to affect
  - The generator output feeds into the discriminator net which produces the output being measured for accuracy
  - The generator loss penalizes the generator for producing a sample that the discriminator network classifies as fake
- Backpropagation is more complicated
  - Because the generator loss is based on the output of the discriminator, back propagation has to run through the discriminator
  - But if we change the weights of the discriminator, then we are trying to tune two different networks that are interdependent at the same time… like trying to hit a moving target

Training the Generator

- Generator training procedure:
  - Sample random noise
  - Produce generator output from sampled random noise
  - Get discriminator "Real" or "Fake" classification for generator output
  - Calculate loss from discriminator classification
  - Backpropagate through both the discriminator and generator to obtain gradients
  - Use gradients to change only the generator weights
- We want to hold the weights for the discriminator constant while we adjust the weights of the generator

Alternating Training

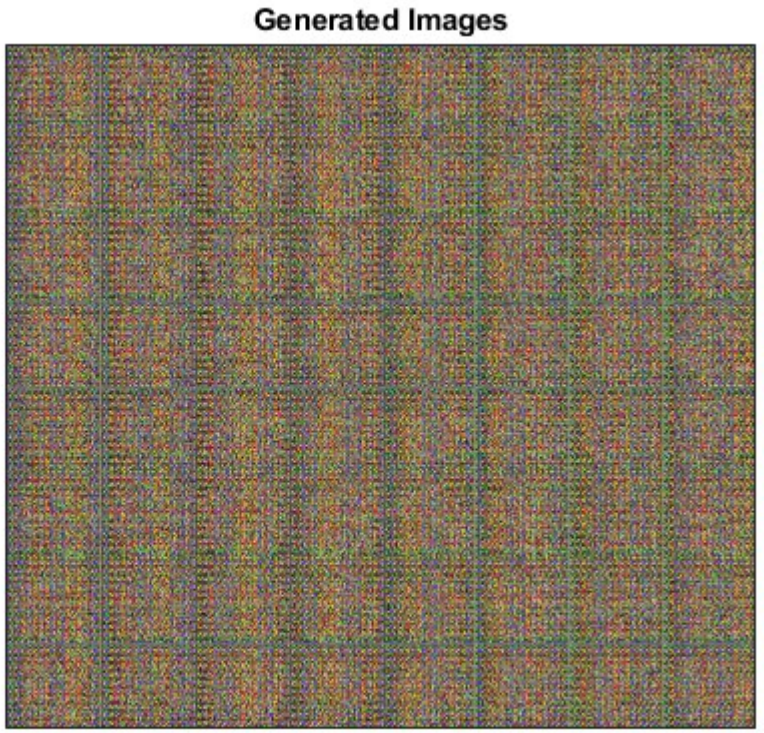
- GAN training alternates between the discriminator and generator
  - The discriminator trains for one or more epochs
  - The generator trains for one or more epochs
  - Iterate to train both the generator and discriminator networks
- The generator is kept immutable during discriminator training
  - The discriminator is being trained on how to recognize the generator's flaws
  - We can’t train properly if the generator is continuously changing while we train the discriminator
- The discriminator is kept immutable during generator training phase
  - Otherwise the feedback to the generator would be constantly changing during training
  - This means that the generator might never converge
- Training starts with a the simpler problem for the discriminator
  - We need to be able to classify real and fake before we can train the generator

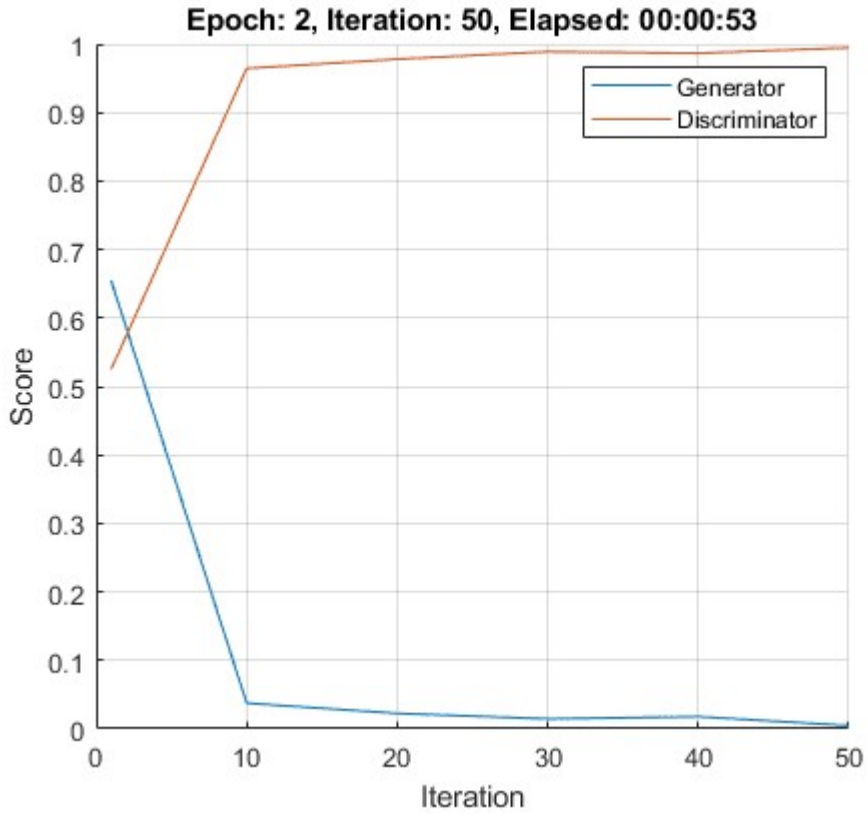
Convergence

- As the generator improves with training
  - the discriminator performance gets worse because the discriminator can't easily tell the difference between real and fake
  - If the generator succeeds perfectly, then the discriminator has a 50% accuracy which is jus random chance
  - The discriminator feedback gets less meaningful over time
  - If the GAN continues training past the point when the discriminator is giving completely random feedback, then the generator starts to train on junk feedback, and its own quality may collapse
- It is a fundamental problem to get this to converge
  - Consider the art forger getting better but the art expert not improving their analytical skills
  - After a while, the art expert is doing little more than guessing if a painting is fake

Convergence Failure - Discriminator Dominates

- Happens when the generator score reaches zero or near zero and the discriminator score reaches one or near one



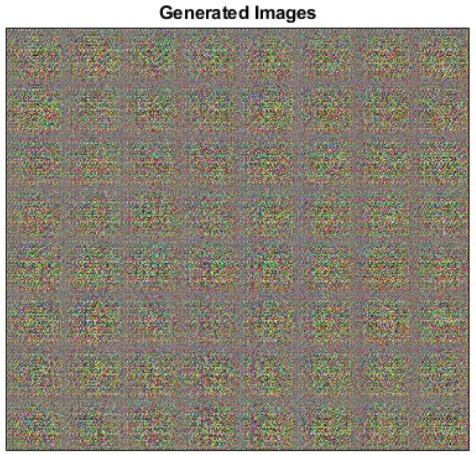


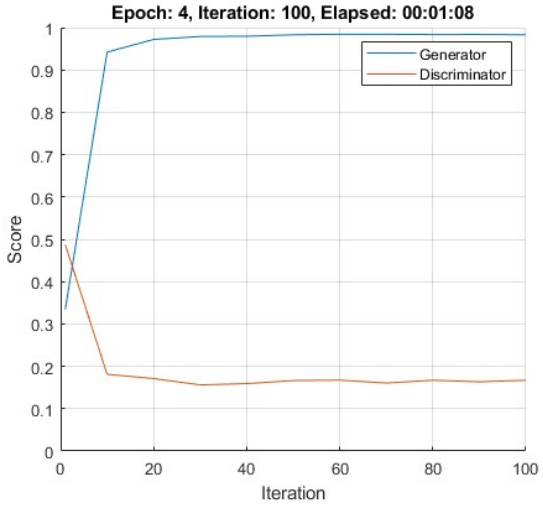
Convergence Failure - Discriminator Dominates

- If the score does not recover from these values for many iterations, then it is better to stop the training
- Try balancing the performance of generator and the discriminator by:
  - Impairing the discriminator by randomly giving false labels to real images
  - Impairing the discriminator by adding dropout layers to reduce overfitting
  - Improving the generator's ability to create more features by increasing the number of filters in its convolution layers
  - Impairing the discriminator by reducing its number of filters

Convergence Failure - Generator Dominates

- Generator score goes to one for a many iterations
  - The generator learned how to always fool the discriminator almost always
  - When this happens very early in the training process, the generator is likely to learn a very simple feature representation which fools the discriminator easily
  - This means that the generated images can be very poor, despite having high scores.



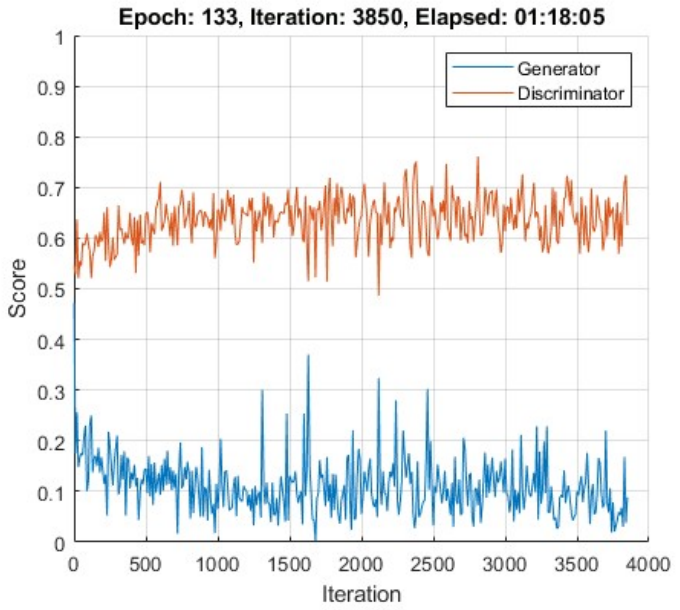


Convergence Failure - Generator Dominates

- If the score does not recover from these values for many iterations, then it is better to stop the training
- Try balancing the performance of generator and the discriminator by:
  - Improving the discriminator's ability to learn more features by increasing the number of filters
  - Impairing the generator by adding dropout layers
  - Impairing the generator by reducing its number of filters

Convergence Failure – Mode Collapse

- GAN produces a small variety of images with many duplicates (modes)
  - Happens when the generator is unable to learn a rich feature representation because it learns to associate similar outputs to multiple different inputs
  - To check for mode collapse, inspect the generated images. If there is little diversity in the output and some of them are almost identical, then there is likely mode collapse.



Convergence Failure – Mode Collapse

- To rectify mode collapse, try to increase the ability of the generator to create more diverse outputs by:
  - Increasing the dimensions of the input data to the generator
  - Increasing the number of filters of the generator to allow it to generate a wider variety of features
  - Impairing the discriminator by randomly giving false labels to real images (one-sided label flipping)

Nash Equilibrium

- From game theory
  - Two opposing players have to choose strategies that maximize their outcome
  - A Nash equilibrium takes place when neither player can improve their outcome by changing a strategy, in otherwords, any change is strategy by a player would negatively impact their outcome
  - Classic example is the prisoner's dilemma

Nash Equilibrium: Prisoner's Dilemma

		Bonnie: RS	Bonnie: C
Clyde	RS	1, 1	4, 0
	C	0, 4	3, 3

(C, C) is the unique Nash Equilibrium in pure-strategies

Nash Equilibrium and GANS

- Is a convergence in a GAN essentially a Nash equilibrium?
- It looks like the answer is “probably not”


Questions?

End Module