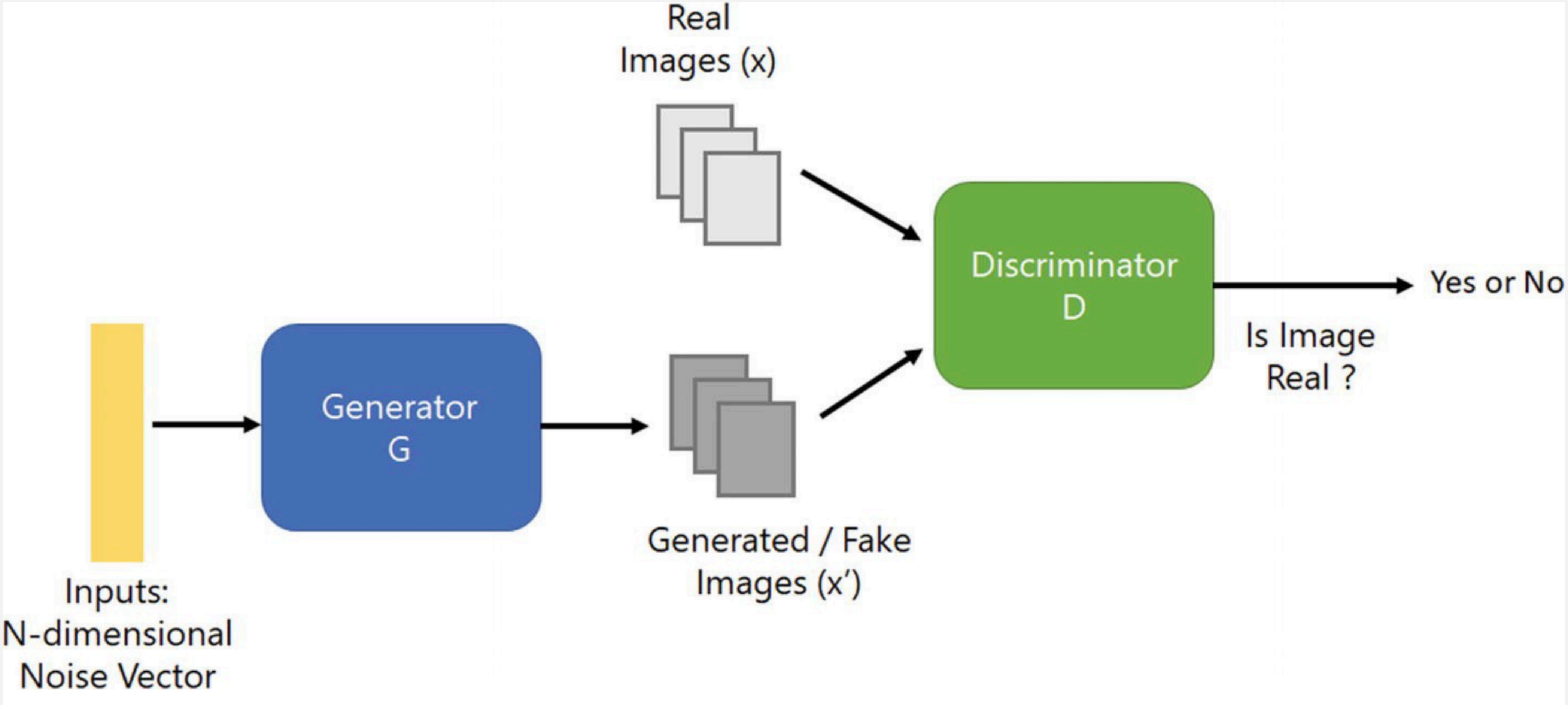
GANS AND GRAPHS

GENERATIVE ADVERSARIAL NETWORKS

- Deep Learning becomes a game
- Two neural networks compete against each other
- One of them generates fake data, and other tries to distinguish real from fake


Real
Images (x)
Discriminator
D
Yes or No
Is Image
Real ?
Generator
G
Generated / Fake
Images (x′)
Inputs:
N-dimensional
Noise Vector

THE GENERATOR

- Input to generate: noise sampled from a uniform distribution
- By putting in different types of noise, we generate all different types of data
- Say we want to generate fake pictures of apples
- There can be green apples, red apples, yellow apples, etc.

THE GENERATOR

- The generator can be any type of neural network
- MLP, CNN are two possibilities (depending on the type of data
- Generator: makes fake data to try to trick the discriminator

THE DISCRIMINATOR

- The discriminator can be any type of neural network
- MLP, CNN are two possibilities (depending on the type of data)
- Discriminator: learns from both real and fake data (tries to tell them apart)

TRAINING A GAN

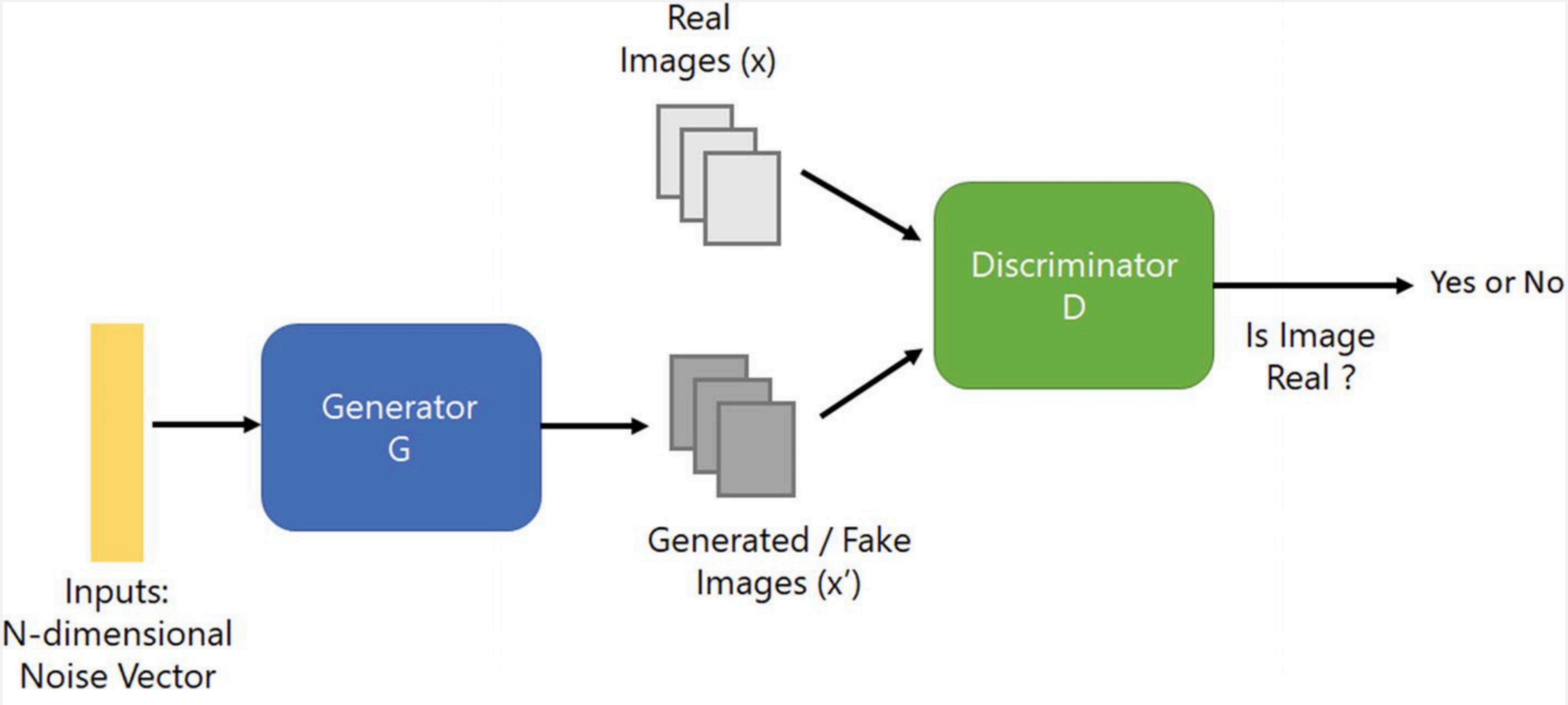
- Remember, this is a game
- If the discriminator does well, the generator must be losing
- If the generator does well, the discriminator must be losing
- Lower loss for the discriminator = higher loss for the generator
- Higher loss for the discriminator = lower loss for the generator

TRAINING A GAN

- If we train both networks at the same time, it will be like trying to hit a moving target (much harder)
- We train the generator for a few epochs, and do not change the weights of the discriminator
- We then train the discriminator for a few epochs, do not change the weights of the generator
- Repeat for a certain number of epochs until the discriminator fails half the time (random guessing)

GENERAL GUIDELINES

- Use the same things you would on a regular neural network
- Activation functions
- Optimization functions
- Loss functions
- Batch normalization (prevents GANs from replicating the same sample over and over)
- Dropouts

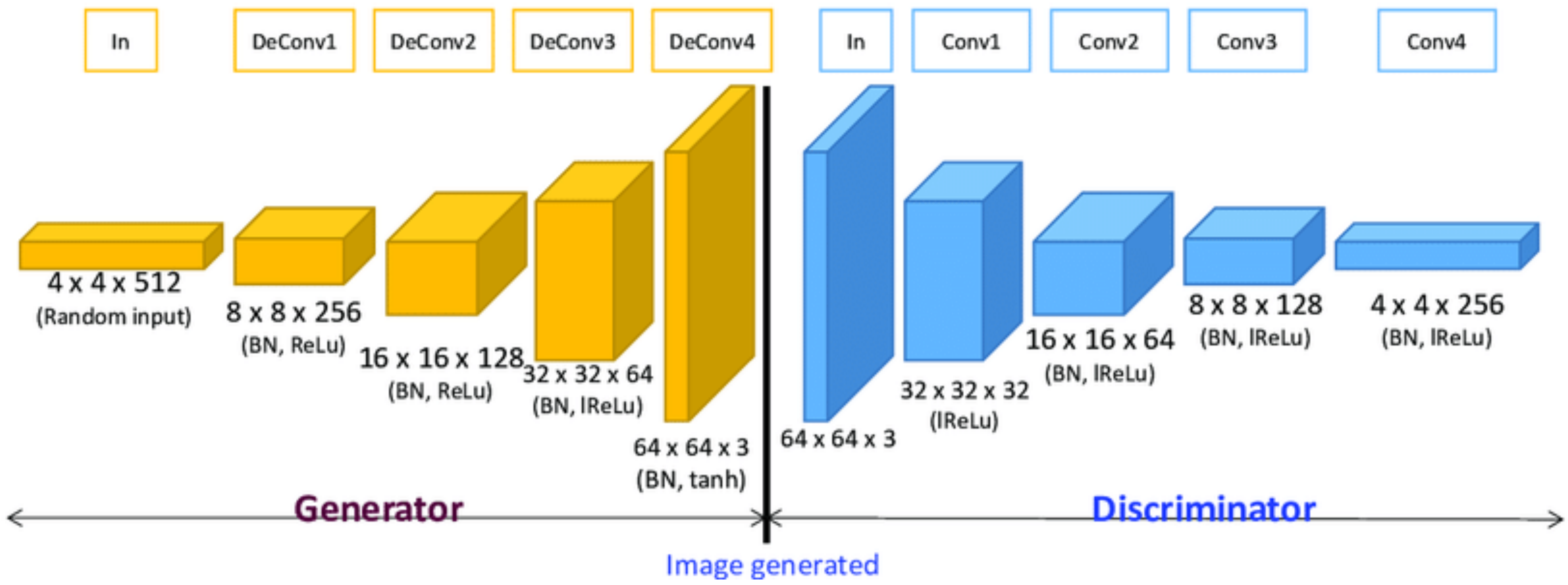

Real
Images (x)
Discriminator
D
Yes or No
Is Image
Real ?
Generator
G
Generated / Fake
Images (x′)
Inputs:
N-dimensional
Noise Vector

OTHER ARCHITECTURES

- We just reviewed the vanilla GAN – the most basic model
- Now, we will discuss more complicated architectures!
- DC-GAN, Cycle GAN, text-to-image
- All can be implemented quickly via Pytorch/online resources
- It is important to understand how these work so you can design projects

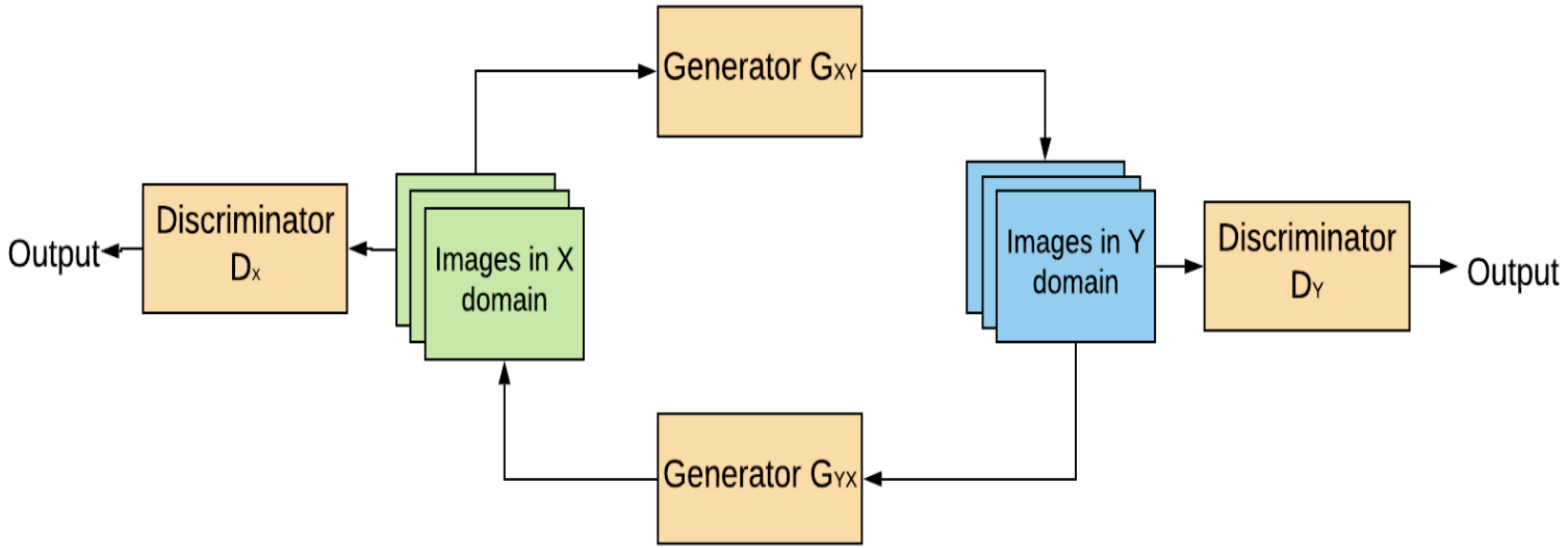
DC-GAN

- TLDR – GAN for images
- Integrates CNN with vanilla GAN for improved results on images


In
DeConv1
DeConv2
DeConv3
DeConv4
In
Conv1
Conv2
Conv3
Conv4
4 x 4 x 512
(Random input)
8 x 8 x 256
(BN, ReLu)
16 x 16 x 128
(BN, ReLu)
32 x 32 x 64
(BN, lReLu)
64 x 64 x 3
(BN, tanh)
64 x 64 x 3
32 x 32 x 32
(lReLu)
16 x 16 x 64
(BN, lReLu)
8 x 8 x 128
(BN, lReLu)
4 x 4 x 256
(BN, lReLu)
Generator
Discriminator
Image generated

CYCLE GAN

- Maps data from one type to the other
- For example, horse to zebra
- Generators: uses domain #1 to create domain #2 and domain #2 to create domain #1
- Discriminator: checks the quality of the translation


Generator GXY
Output
Discriminator
Dx
Images in X
domain
Images in Y
domain
Discriminator
DY
Output
Generator GYX

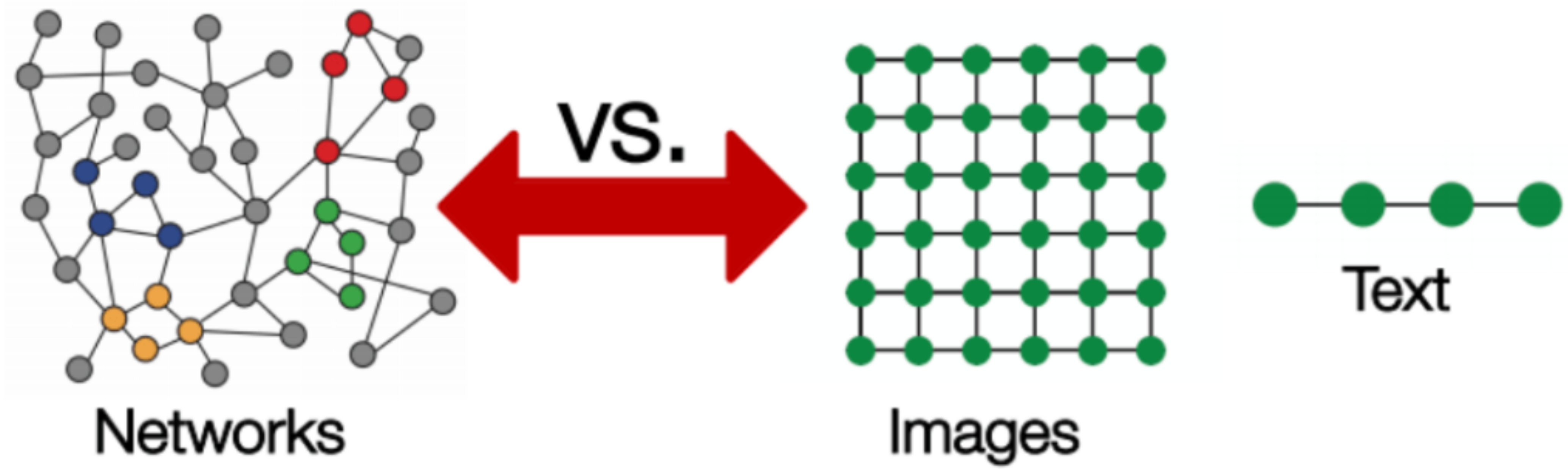
BIOMED APPLICATIONS?

GRAPHS

Graphs let us represent relationships between data points

A data point can be a node with edges connected to neighboring (similar) data points

We can learn from the network, not just individual data points


vs.
Networks
Images
Text

GRAPH NEURAL NETWORKS

- You have to make the graph first
- Scikit-learn has many ways to do this – see [here]

GNNS

Most traditional application is supervised classification

Each node has a label (i.e. cell type)

Use info from that node and nearby nodes to predict the label

Learn a model that can be used to predict new labels

$$\mathrm{h}_v = f\left(\mathrm{x}_v, \mathrm{x}_{co[v]}, \mathrm{h}_{ne[v]}, \mathrm{x}_{ne[v]}\right)$$

$$\mathbf{o}_v = g(\mathbf{h}_v, \mathbf{x}_v)$$

BIOMED APPLICATIONS?