AdTrans project, 2023-1-PL01-KA220-HED-000158917

ADTRANS
ADVANCED DATA ANALYSIS FOR TRANSPORTATION AND MOBILITY ACADEMIC TEACHERS

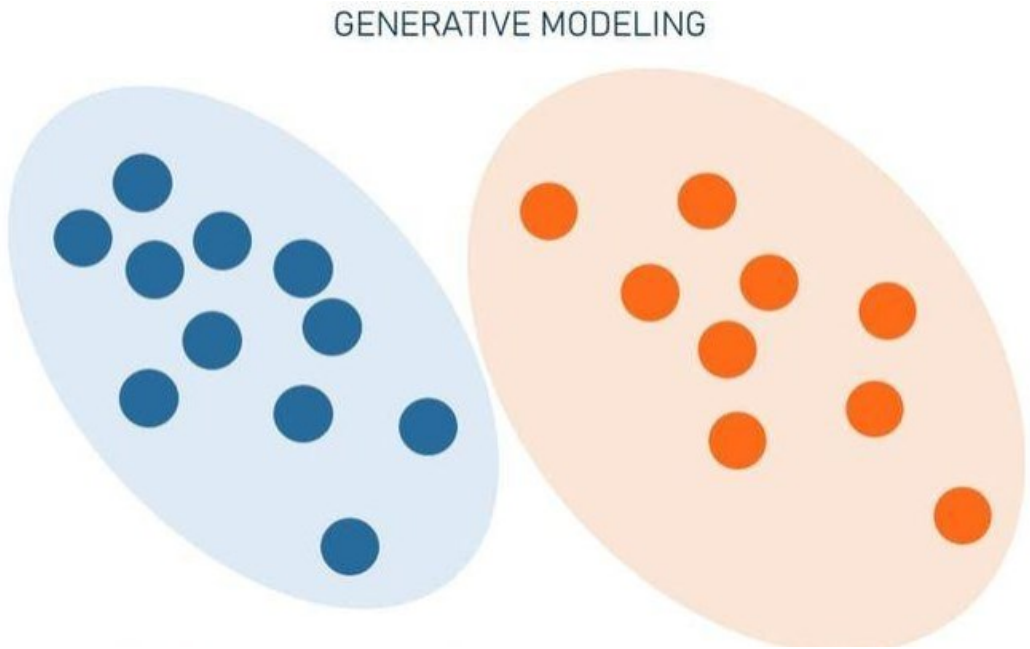
Session 2

Generative Models

Paraskevi Fasouli

Co-funded by the European Union

Generative Models

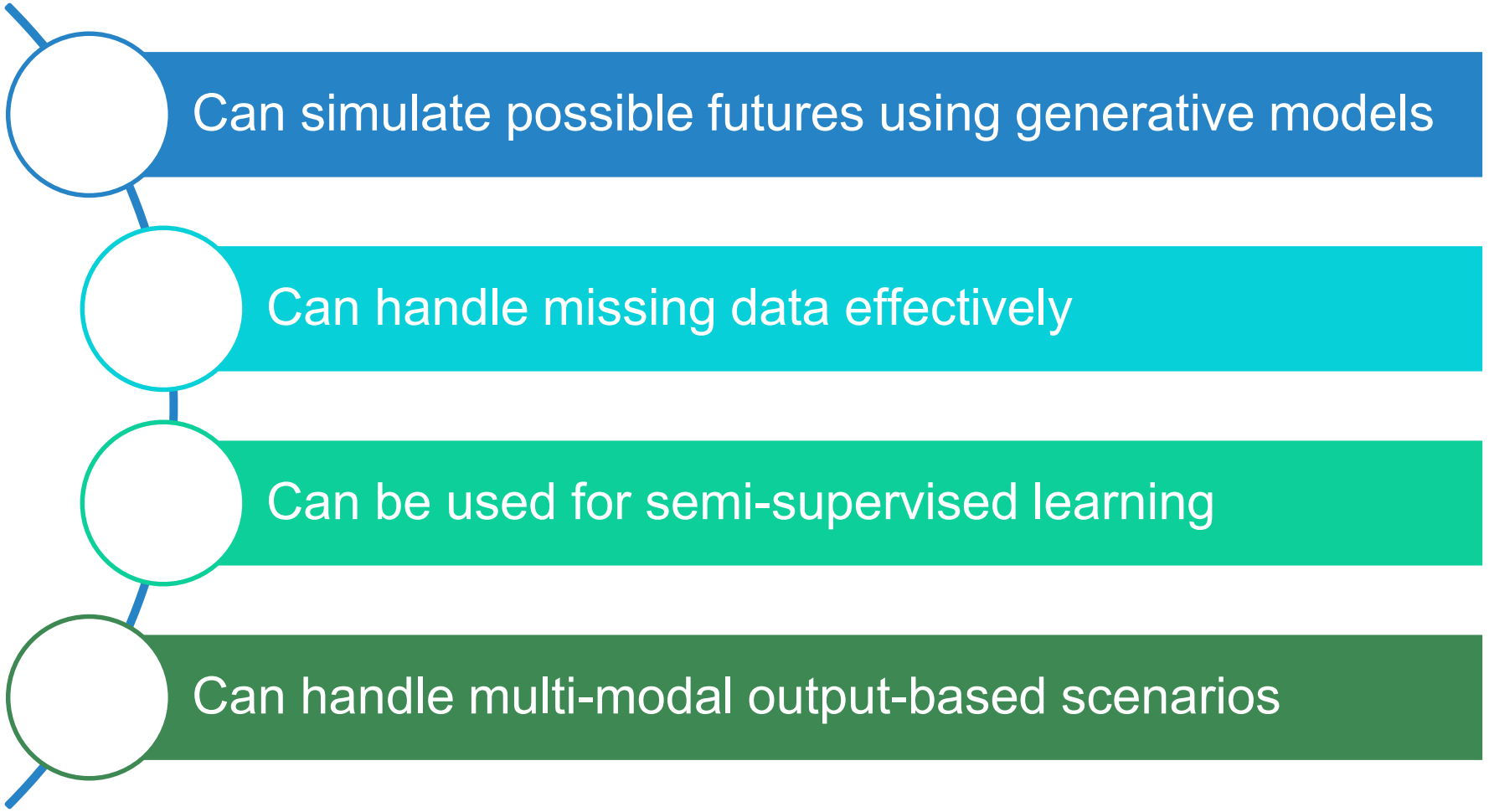


- A generative model tries to learn the joint probability of the input data and labels simultaneously i.e. P(x,y)
- Potential to understand and explain the underlying structure of the input data even when there are no labels

Generative Models

- ✔ Ability of the vehicle to perceive, predict, and interact with their environment in complex and dynamic driving scenarios
- ✔ Generation of realistic **synthetic data** of **trajectories** and **environments** for **training** and **simulations**
- ✔ Facilitation of the development, training, and evaluation of autonomous driving systems
- ✔ Enhancement of vehicle's safety, efficiency, and reliability on the road

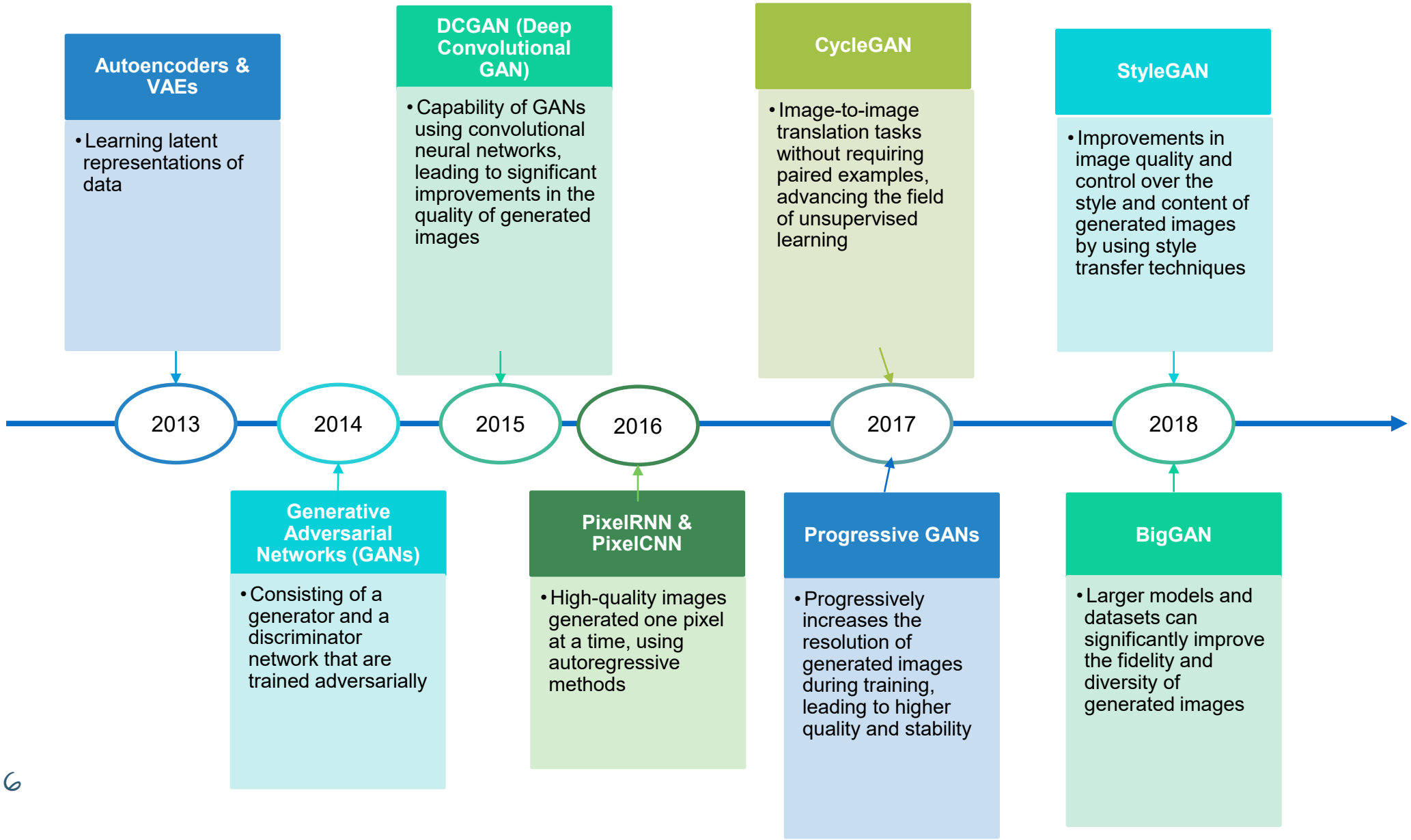
Need for Generative Models



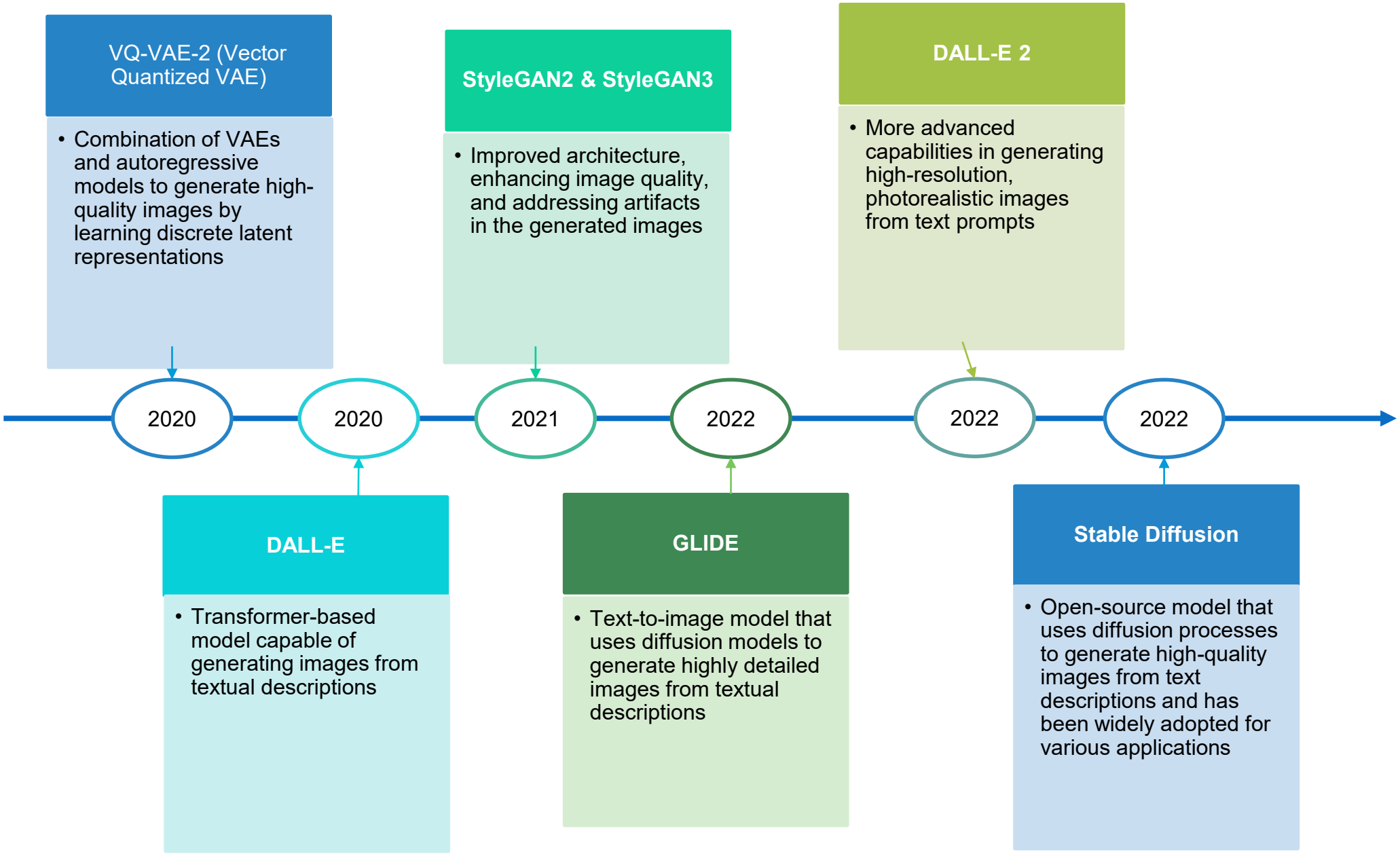
Applications

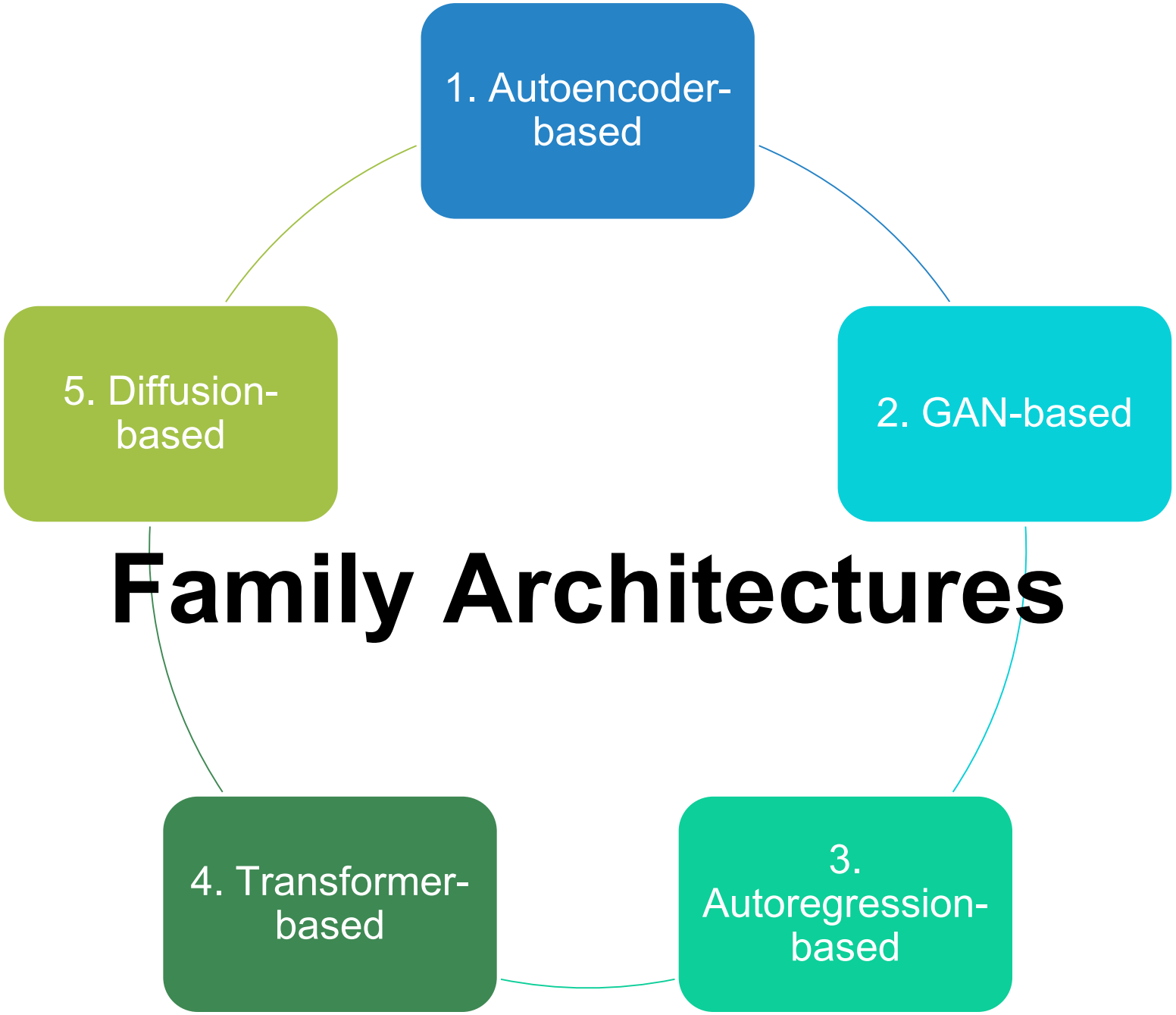
- Image Generation
- LiDAR Point Cloud Generation
- Semantic Segmentation
- Trajectory Prediction
- Behavioral Cloning
- Anomaly Detection
- Environment Simulation

Timeline in 2010s



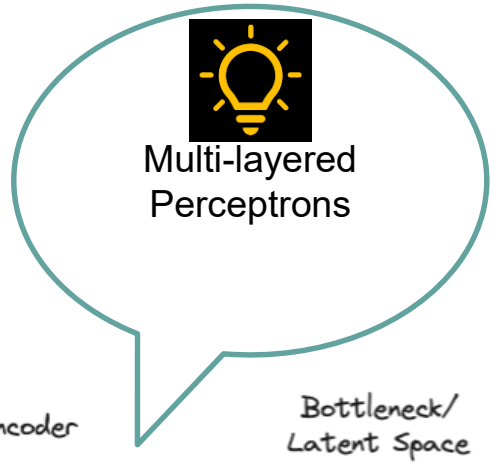
Timeline in 2020s

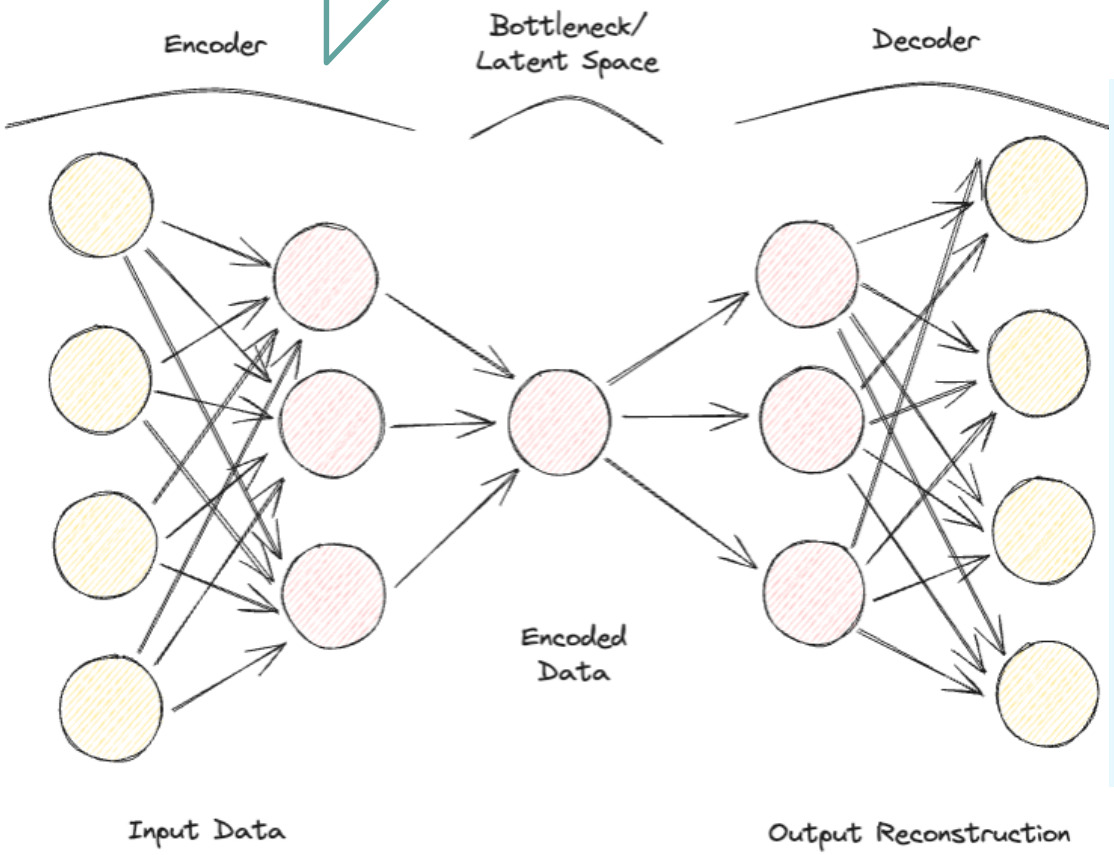


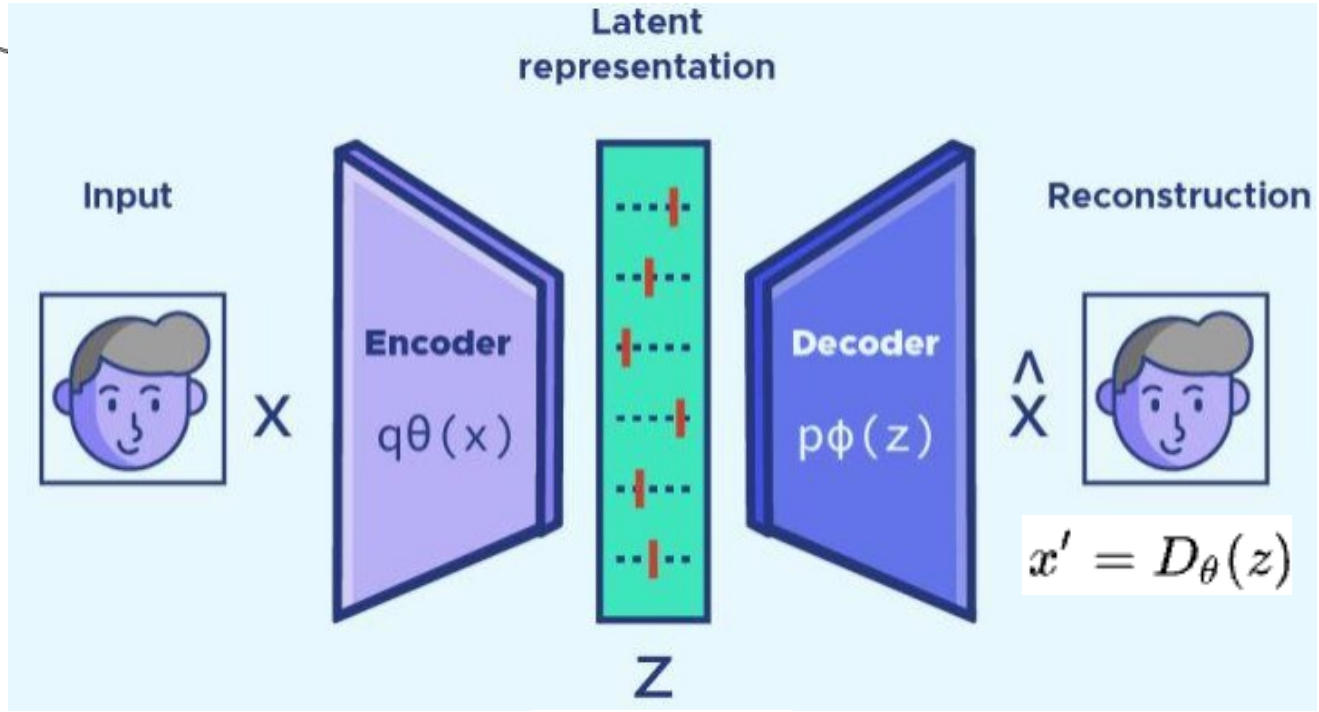


Family Architectures

Autoencoders

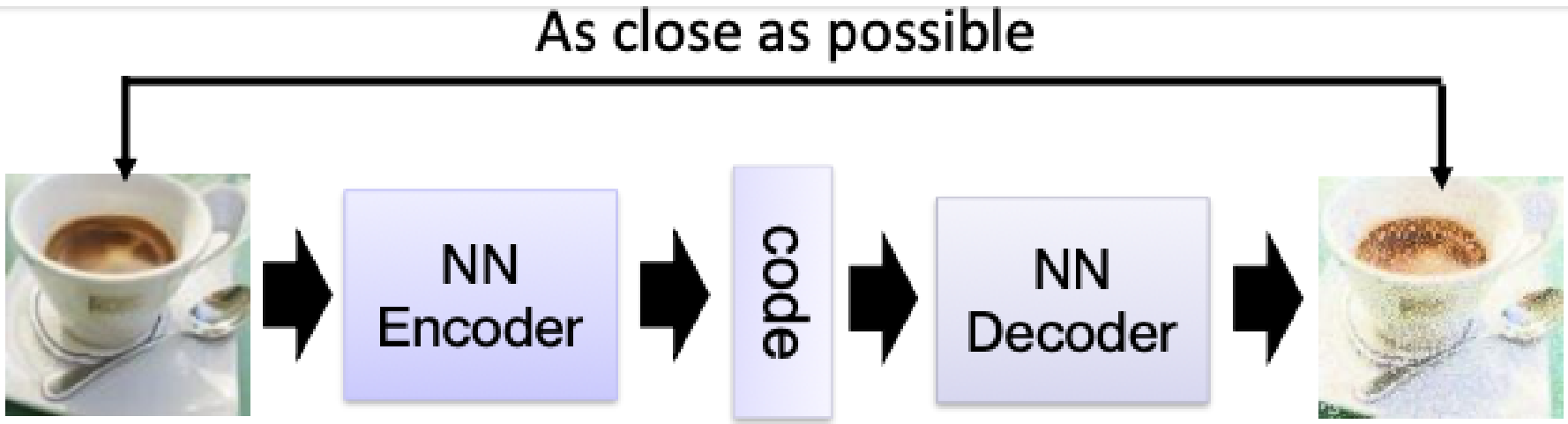






$$z = E_\phi(x)$$

Autoencoders

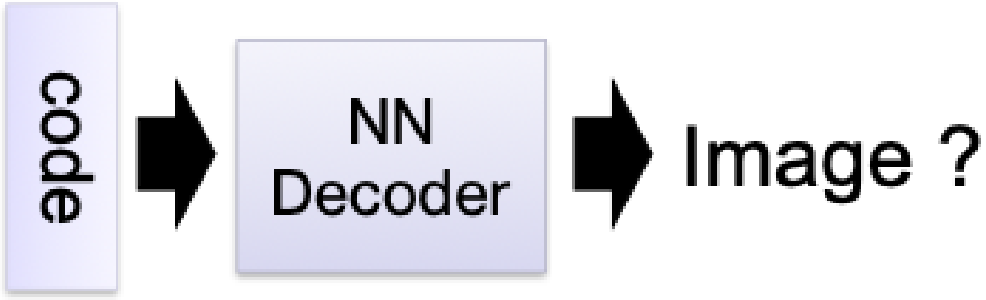


Original Image

Encoded Image

Reconstructed Image

Randomly generate a vector as code

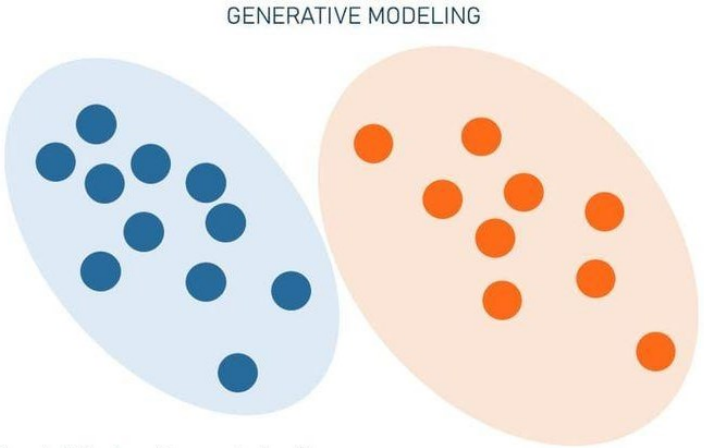


GANs

Generative Model

- A generative model tries to learn the joint probability of the input data and labels simultaneously i.e. P(x,y)

Potential to understand and

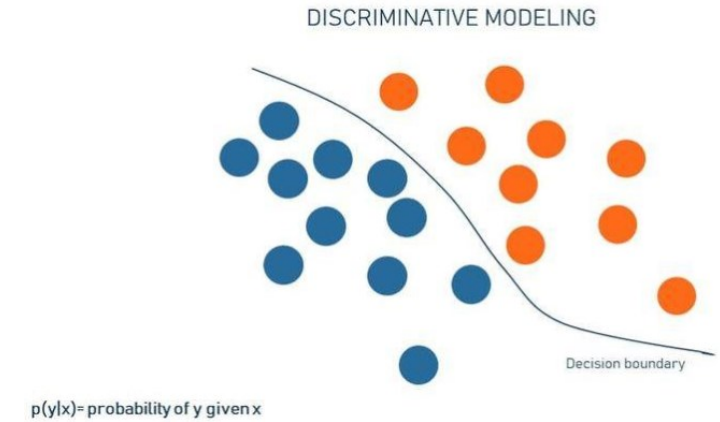


of

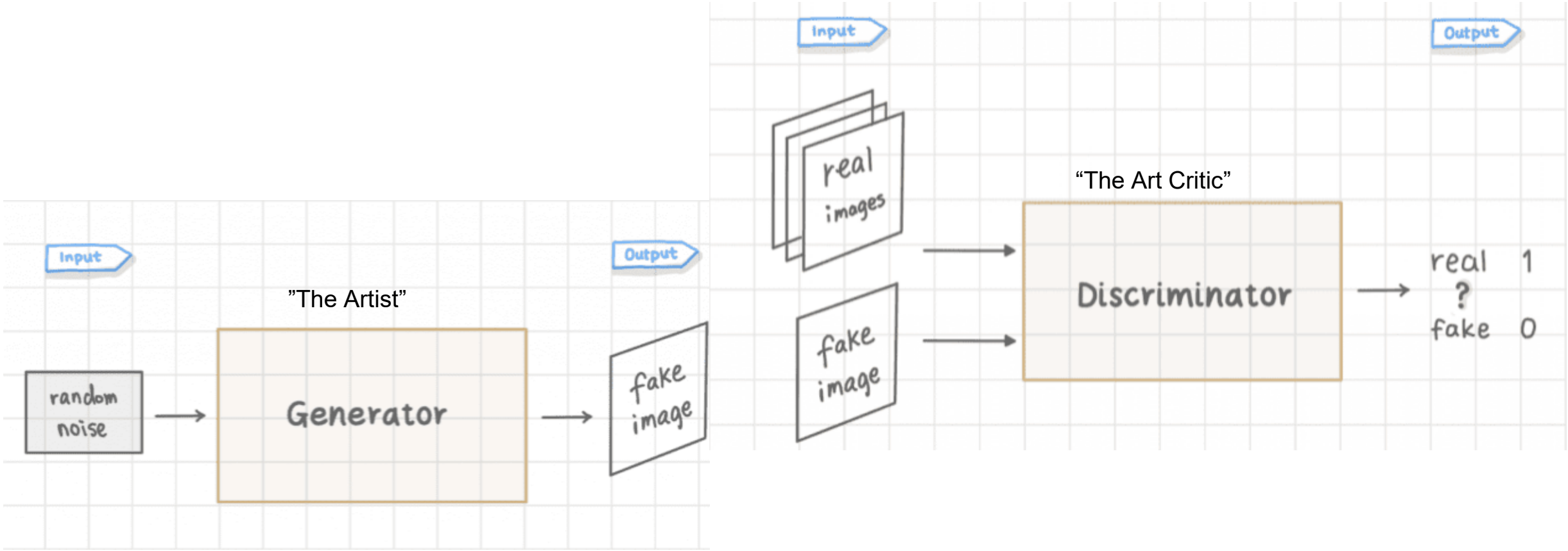
labels.

Discriminative Model

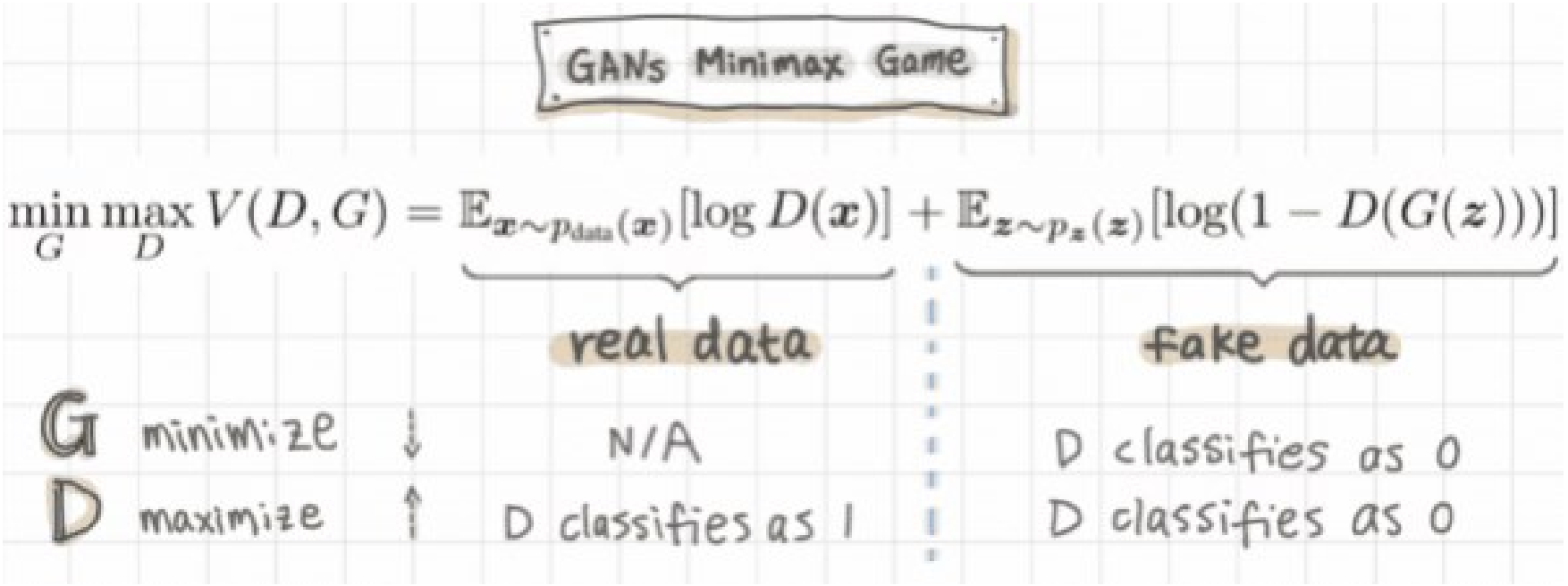
- A discriminative model learns a function that maps the input data (x) to some desired output class label (y).
- In probabilistic terms, they directly learn the conditional distribution P(y|x)



GANs


Input
random noise
”The Artist”
Generator
Output
fake image
Input
real images
fake image
“The Art Critic”
Discriminator
Output
real 1
?
fake 0

How to train GANs



- The generator tries to fool the discriminator into thinking the fake images as real.
- The discriminator tries to classify real and fake images correctly.

Training GANs is to find an equilibrium in the game when:

- The generator makes data that looks almost identical to the training data.
- The discriminator can no longer tell the difference between the fake images from the real images.

Evaluating GANs

A good GAN model should have good image **quality.** To evaluate the GAN model, you can visually inspect the generated images during training or by inference with the generator model. For quantitative evaluation, there are 2 popular metrics:

1. **Inception Score,** which captures both the ***quality*** and ***diversity*** of the generated images
2. **Fréchet Inception Distance** which compares the real vs. fake images and doesn't just evaluate the generated images in isolation

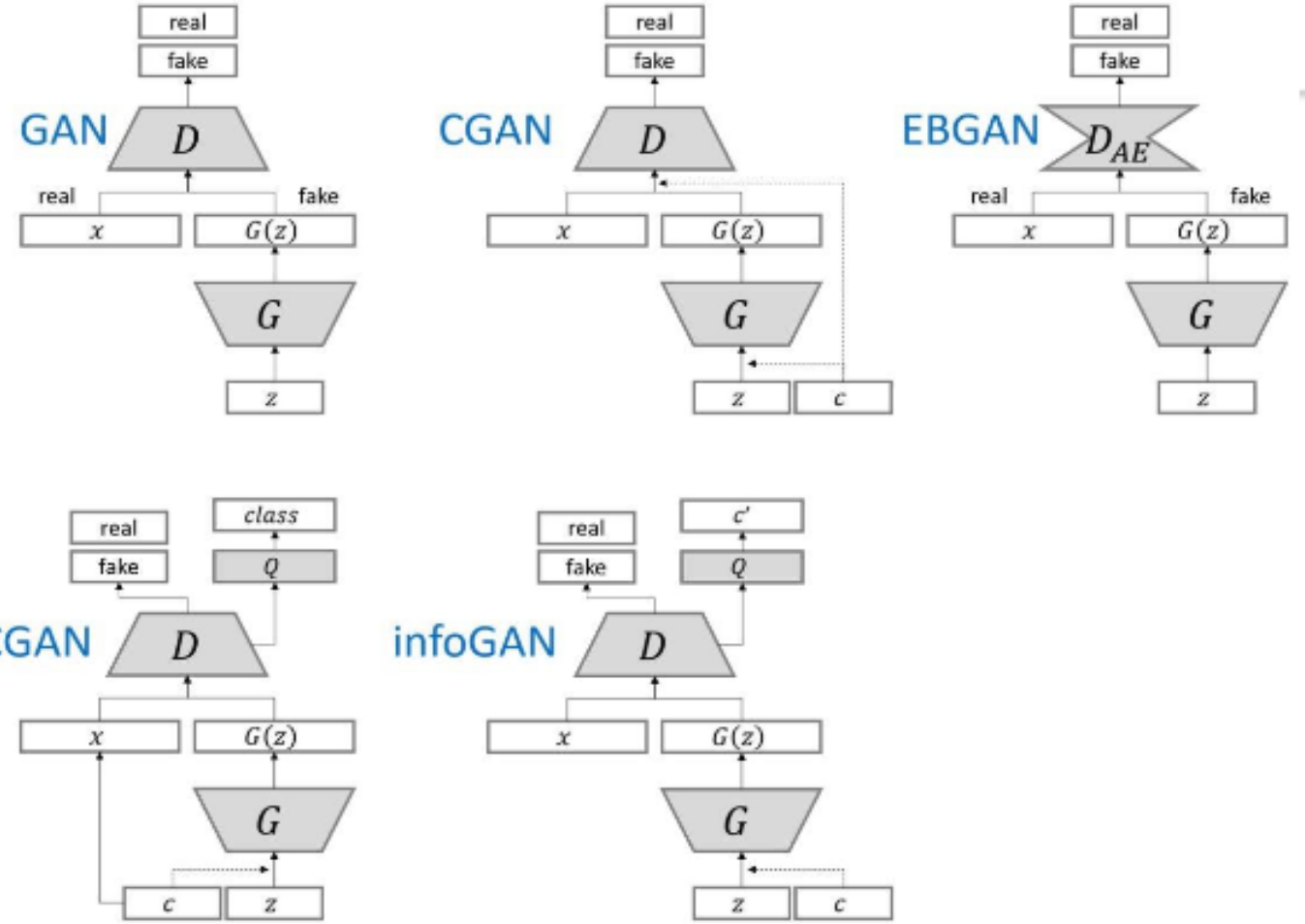
How GANs are being used

- ✔ Applied for modelling natural images.
- ✔ Performance is good in comparison to other generative models.
- ✔ Useful for unsupervised learning tasks.

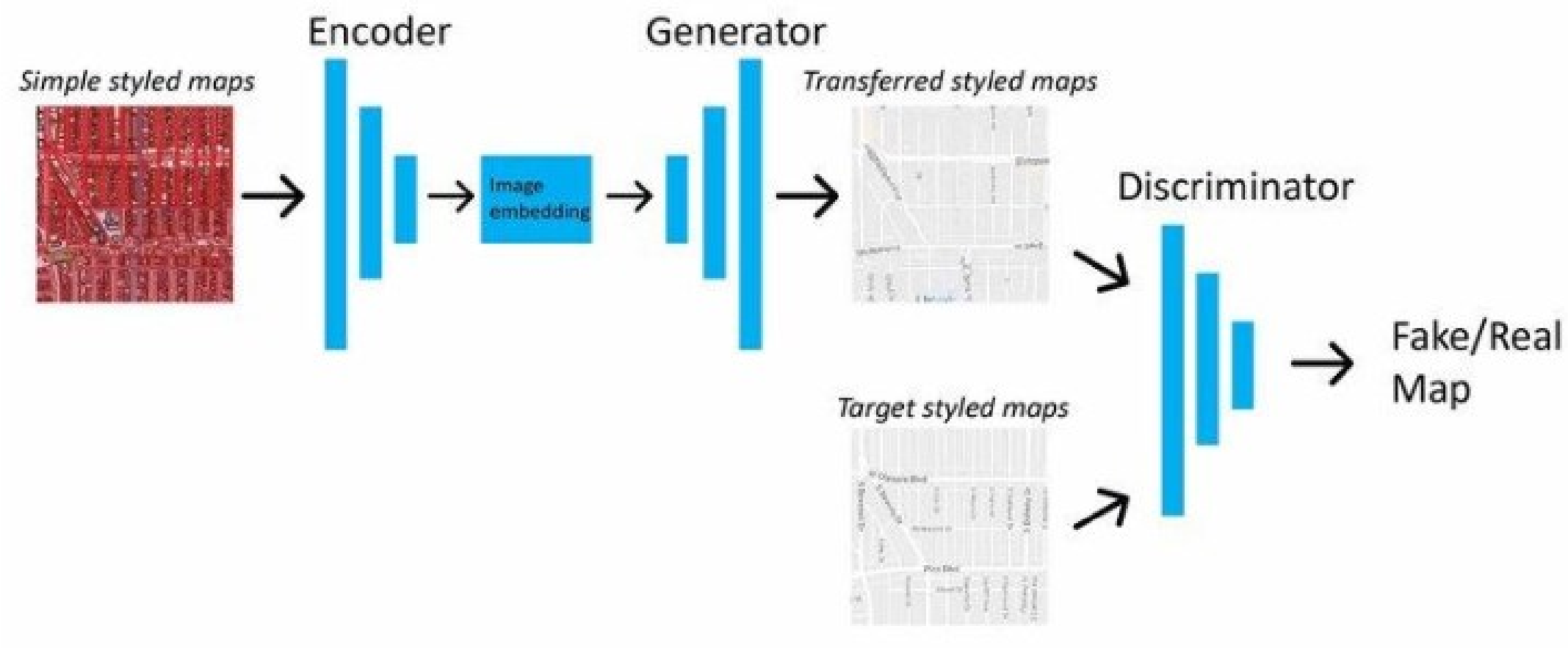
Variations of GANs

- ✔ X is real image from dataset
- ✔ Z is noise vector
- ✔ C is conditional vector *(1. a feature vector derived from an image that encodes the object*

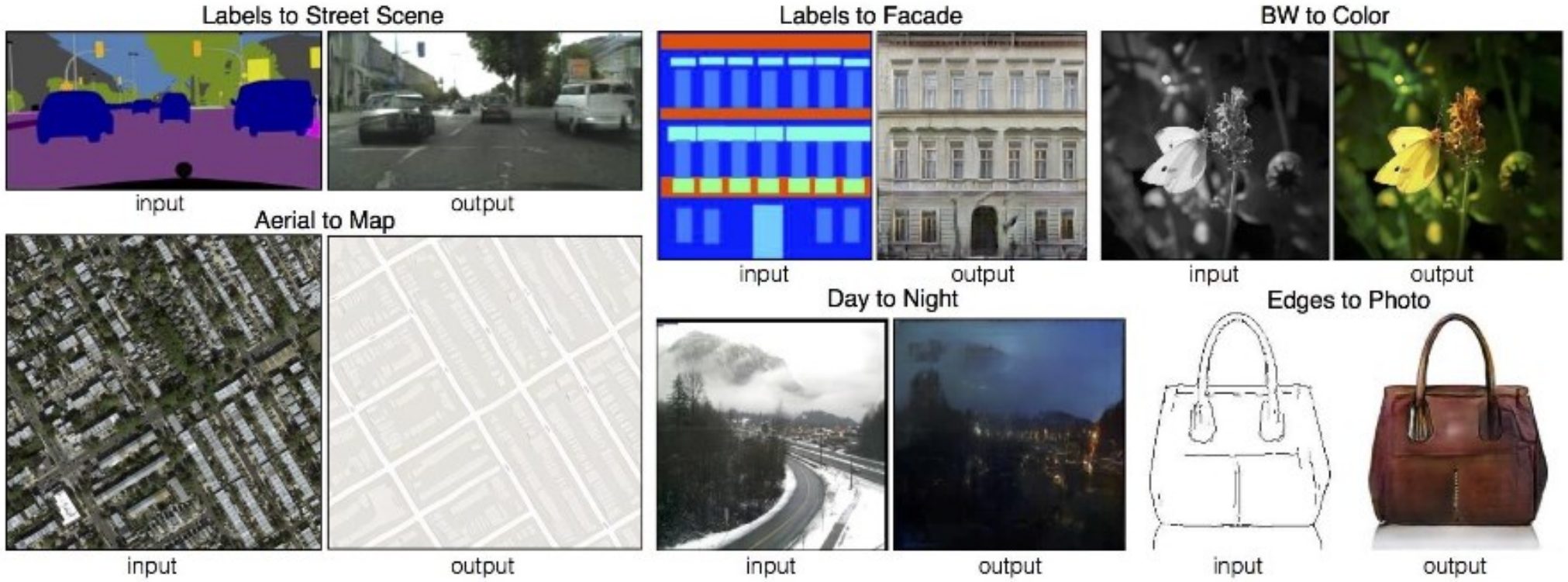
  or 2. a set of specific characteristics we expect from the image.)



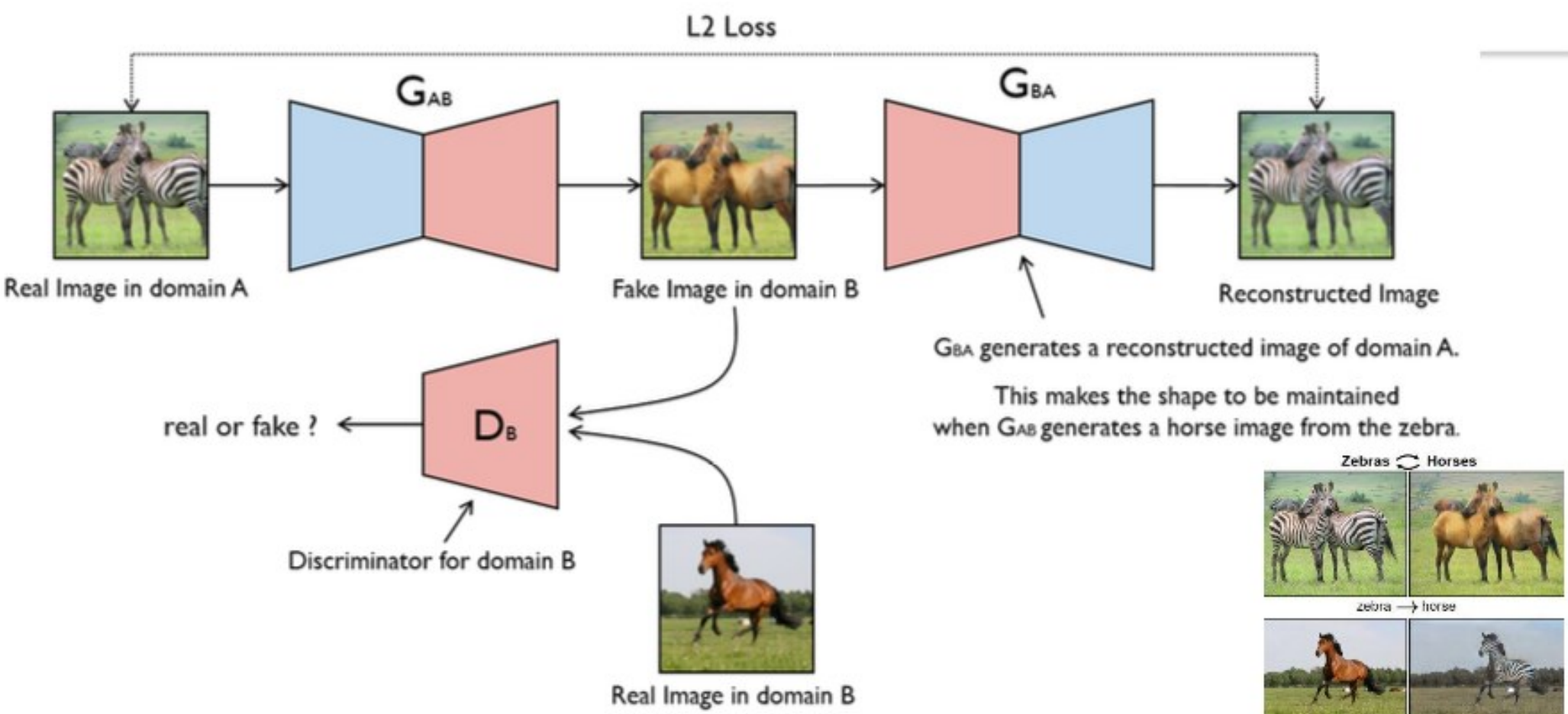
Pix2Pix


Simple styled maps
Encoder
Image embedding
Generator
Transferred styled maps
Discriminator
Target styled maps
Fake/Real Map

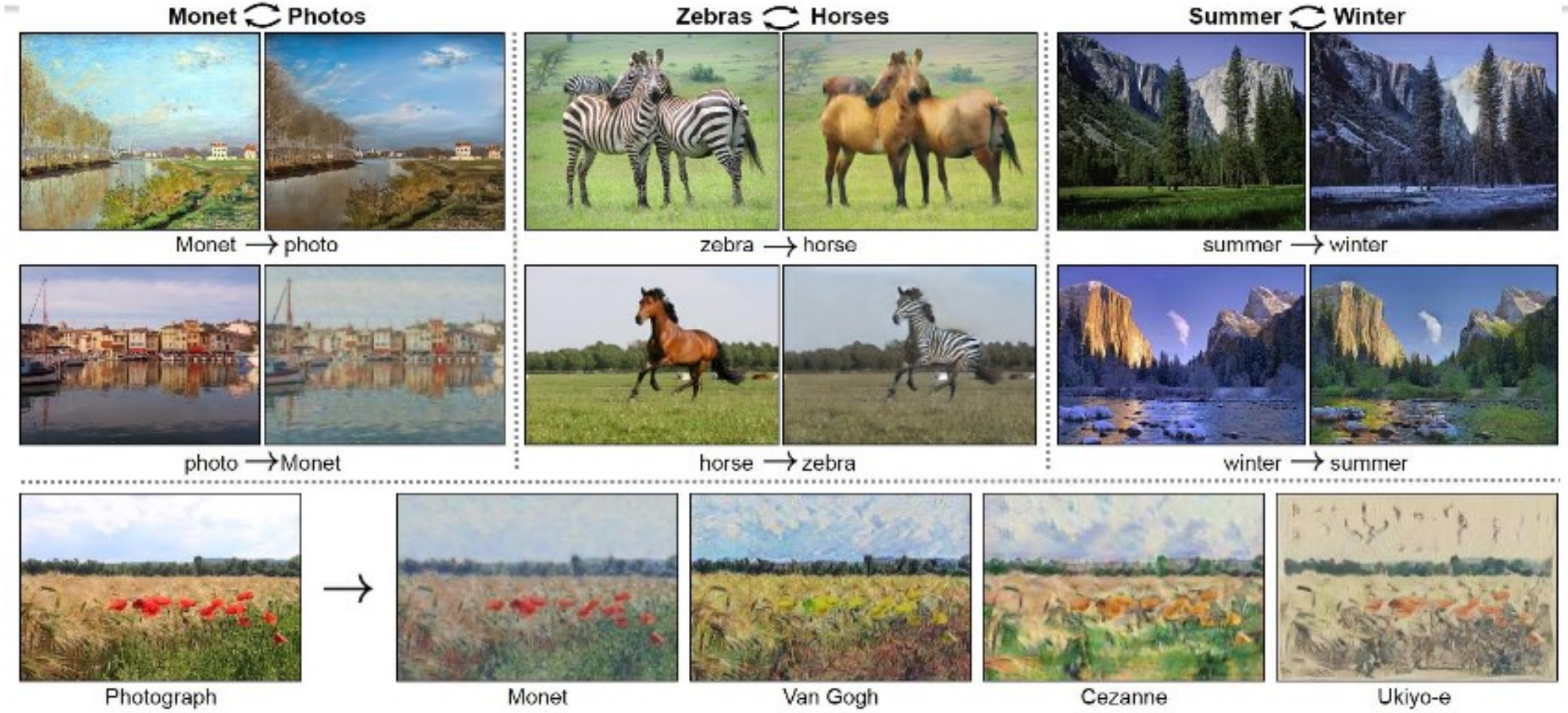
Pix2Pix Use Cases



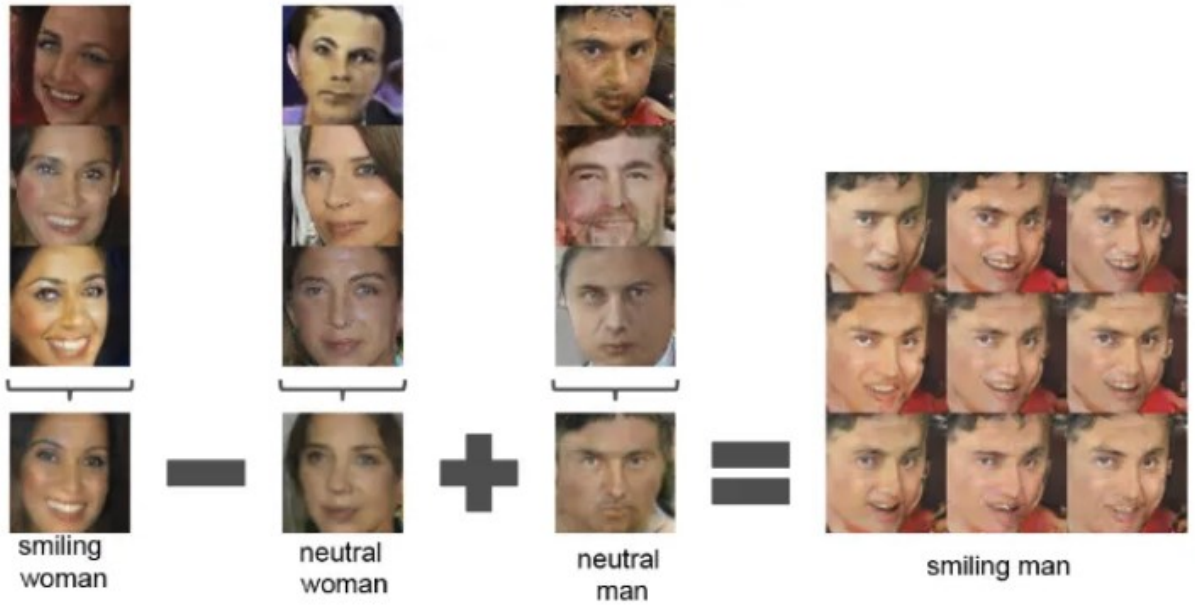
CycleGAN


L2 Loss
G_{AB}
G_{BA}
Real Image in domain A
Fake Image in domain B
Reconstructed Image
G_{BA} generates a reconstructed image of domain A.
This makes the shape to be maintained
when G_{AB} generates a horse image from the zebra.
real or fake ?
D_B
Discriminator for domain B
Real Image in domain B
Zebras
Horses
zebra → horse
horse → zebra

CycleGAN Use Cases



DCGAN



This model implementation will be your homework

The architecture guidelines for stable Deep Convolutional GANs:

- Replace any pooling layers with strided convolutions (discriminator) and fractional-strided convolutions (generator).
- Use batchnorm in both the generator and the discriminator.
- Remove fully connected hidden layers for deeper architecture.
- Use ReLU activation in generator for all layers except for the output, which uses Tanh.
- Use LeakyReLU activation in the discriminator for all layers.

Common Failure Cases

- The discriminator becomes too strong too quickly and the generator ends up not learning anything.
- The generator only learns very specific weaknesses of the discriminator.
- The generator learns only a very small subset of the true data distribution.

Optimization Tips

- Normalize the inputs
- A modified loss function
- Use a spherical Z
- BatchNorm
- Avoid Sparse Gradients: ReLU, MaxPool
- Use Soft and Noisy Labels
- DCGAN / Hybrid Models
- Track failures early (D loss goes to 0: failure mode)
- If you have labels, use them
- Add noise to inputs, decay over time

End of Session 2