UNIVERSITY OF CENTRAL MISSOURI

Department of Computer Science & Cybersecurity

CHPATER 11. GAN, Generative AI

CS 5720: Neural Network & Deep Learning

Dr. I Hua Tsai, Assistant Professor

"This (GANS), and the variations that are now being proposed is the most interestingidea in the last 10 years in ML, in my opinion"

–Yann LeCun

Video:
https://www.youtube.com/watch?v=QiiSAvKJIHo

WHAT ARE GANS?

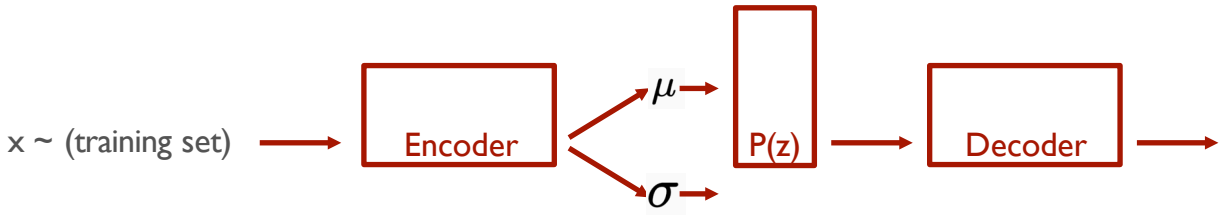
Generative Adversarial Networks

WHAT ARE GANS?

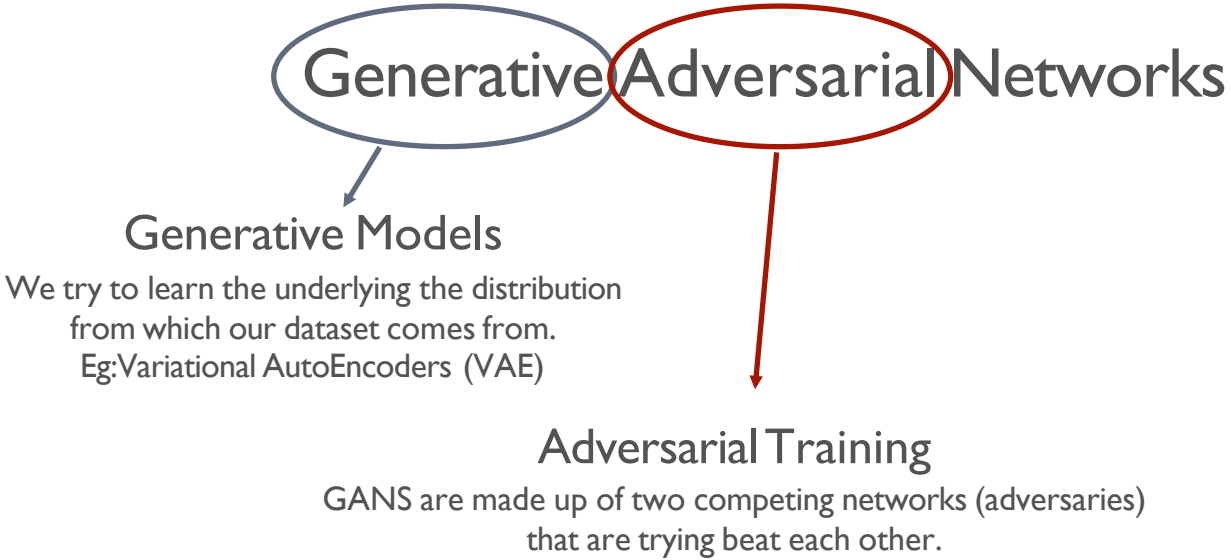
Generative Adversarial Networks

Generative Models

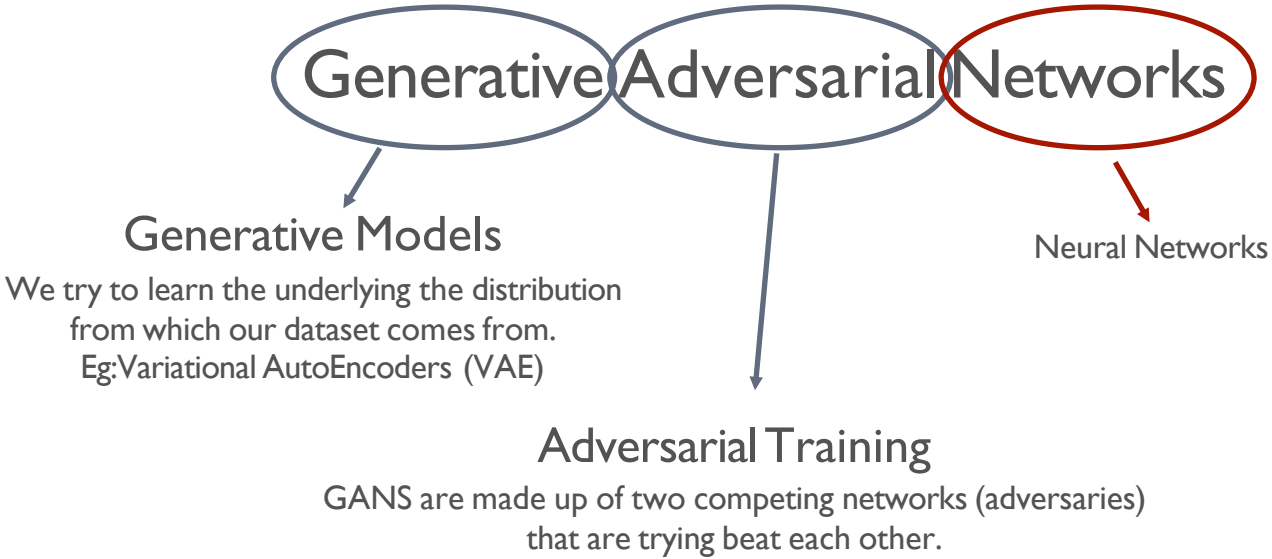
We try to learn the underlying the distribution from which our dataset comes from.
Eg:Variational AutoEncoders (VAE)



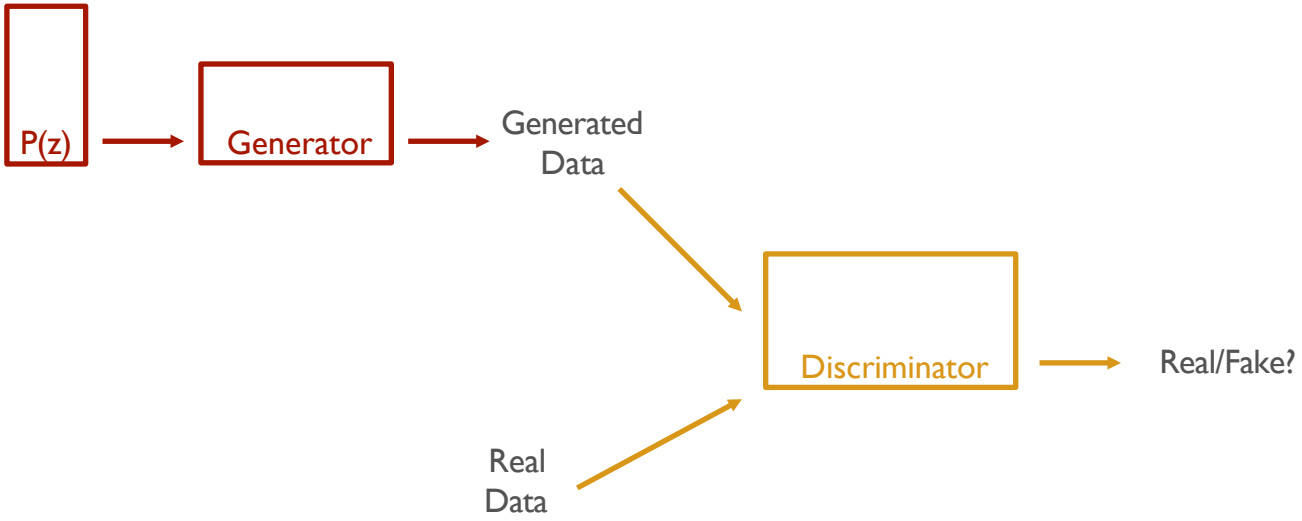
WHAT ARE GANS?



WHAT ARE GANS?



WHAT ARE GANS?



Why Generative Models?

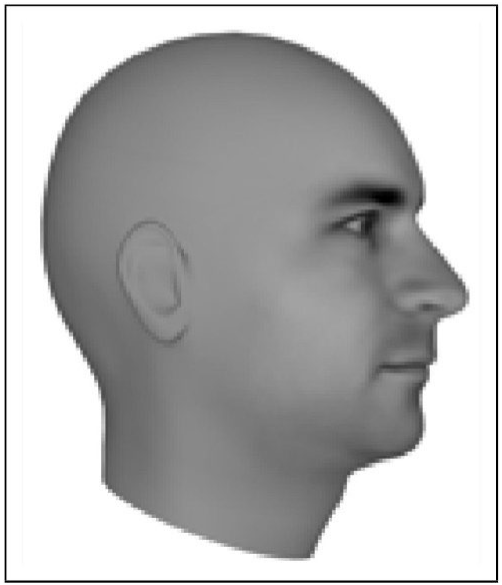
- **We've only seen discriminative models so far**
  - Given an image **X**, predict a label **Y**
  - Estimates **P(Y|X)**

- **Discriminative models have several key limitations**
  - Can't model **P(X)**, i.e. the probability of seeing a certain image
  - Thus, can't sample from **P(X)**, i.e. **can't generate new images**

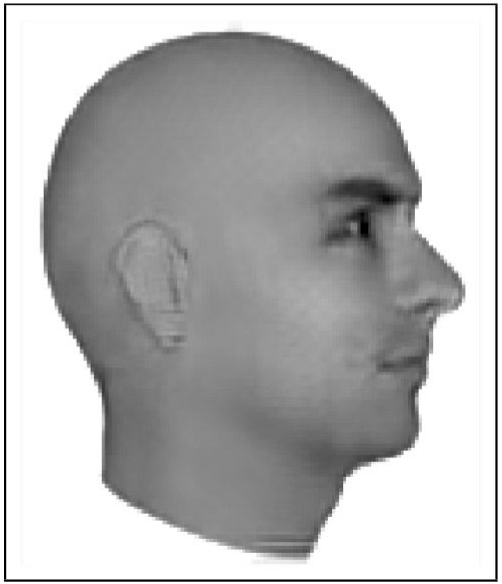
- **Generative models (in general) cope with all of above**
  - Can model **P(X)**
  - Can generate new images

Magic of GANs…

Ground Truth



Adversarial



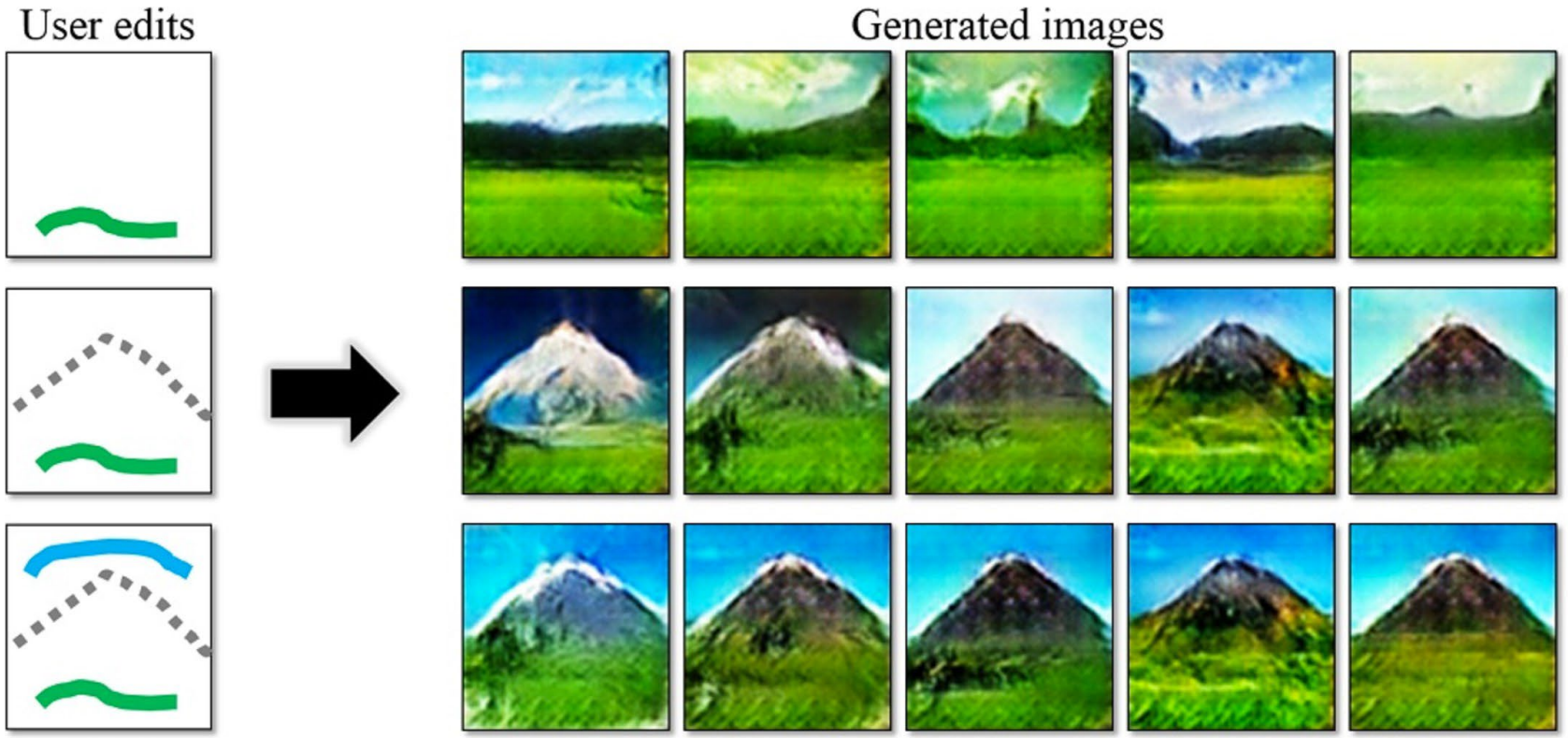
Lotter, William, Gabriel Kreiman, and David Cox. "Unsupervised learning of visual structure using predictive generative networks." *arXiv preprint arXiv:1511.06380* (2015).

Magic of GANs…

Which one is Computer generated?

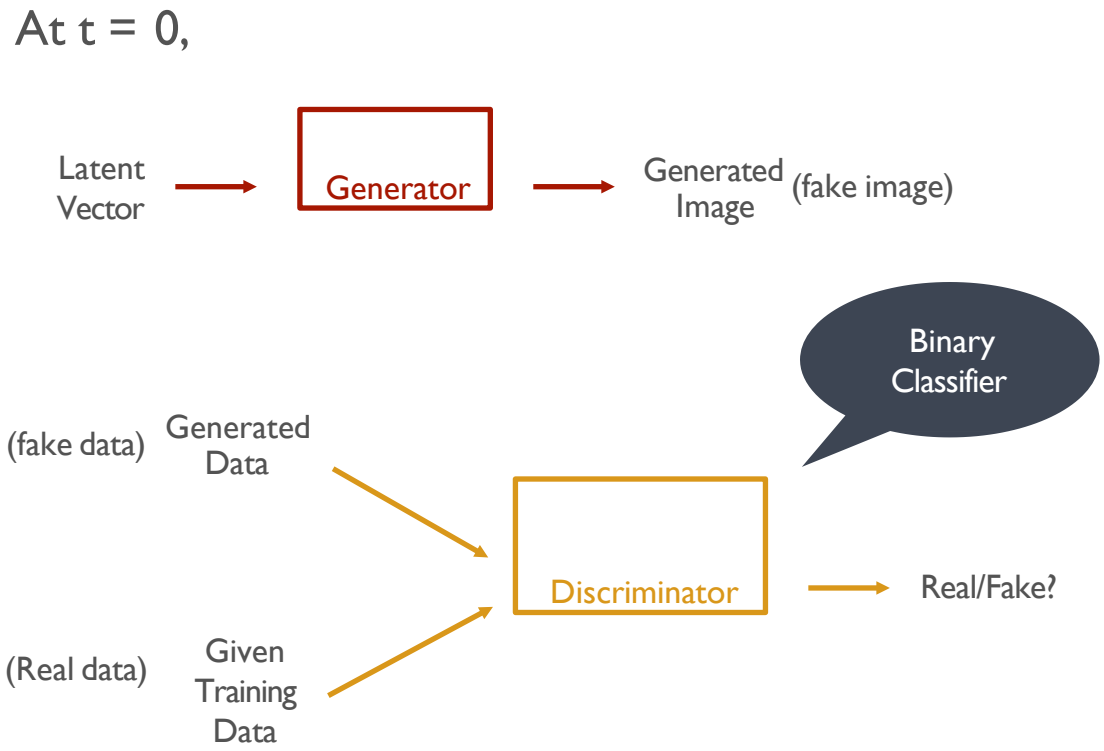
Ledig, Christian, et al. "Photo-realistic single image super-resolution using a generative adversarial network." *arXiv preprint arXiv:1609.04802* (2016).

Magic of GANs…



http://people.eecs.berkeley.edu/~junyanz/projects/gvm/

HOW TO TRAIN A GAN?

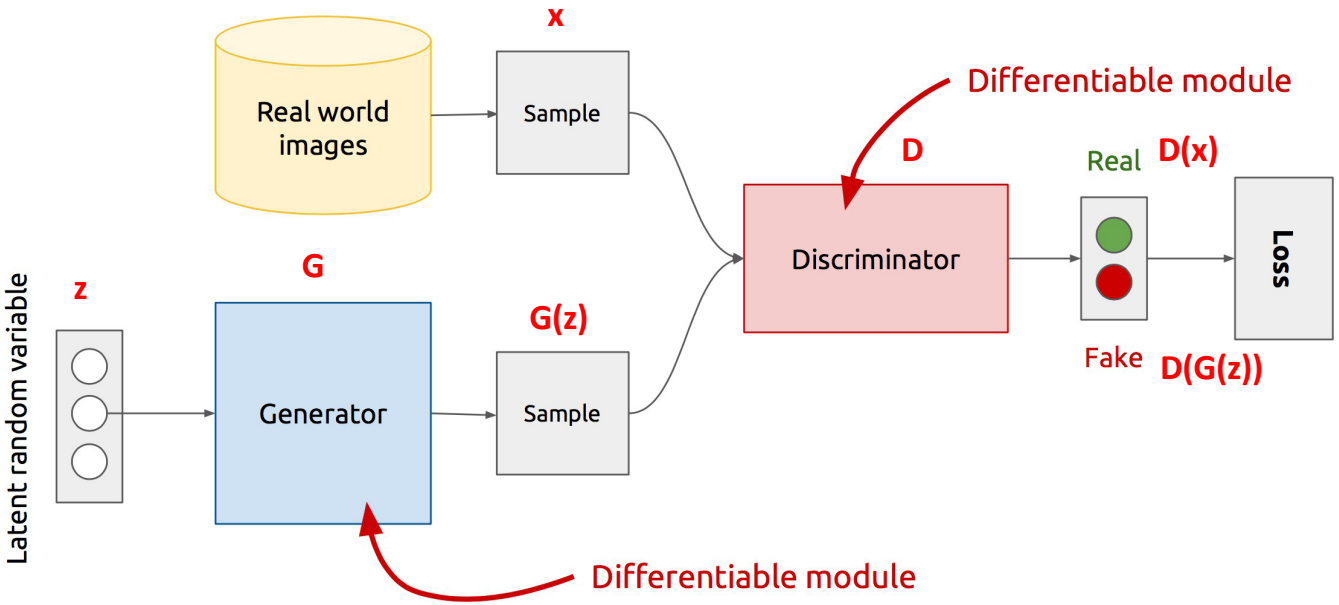


HOW TO TRAIN A GAN?

Step 1:
Train the Discriminator
using the current ability
of the Generator.

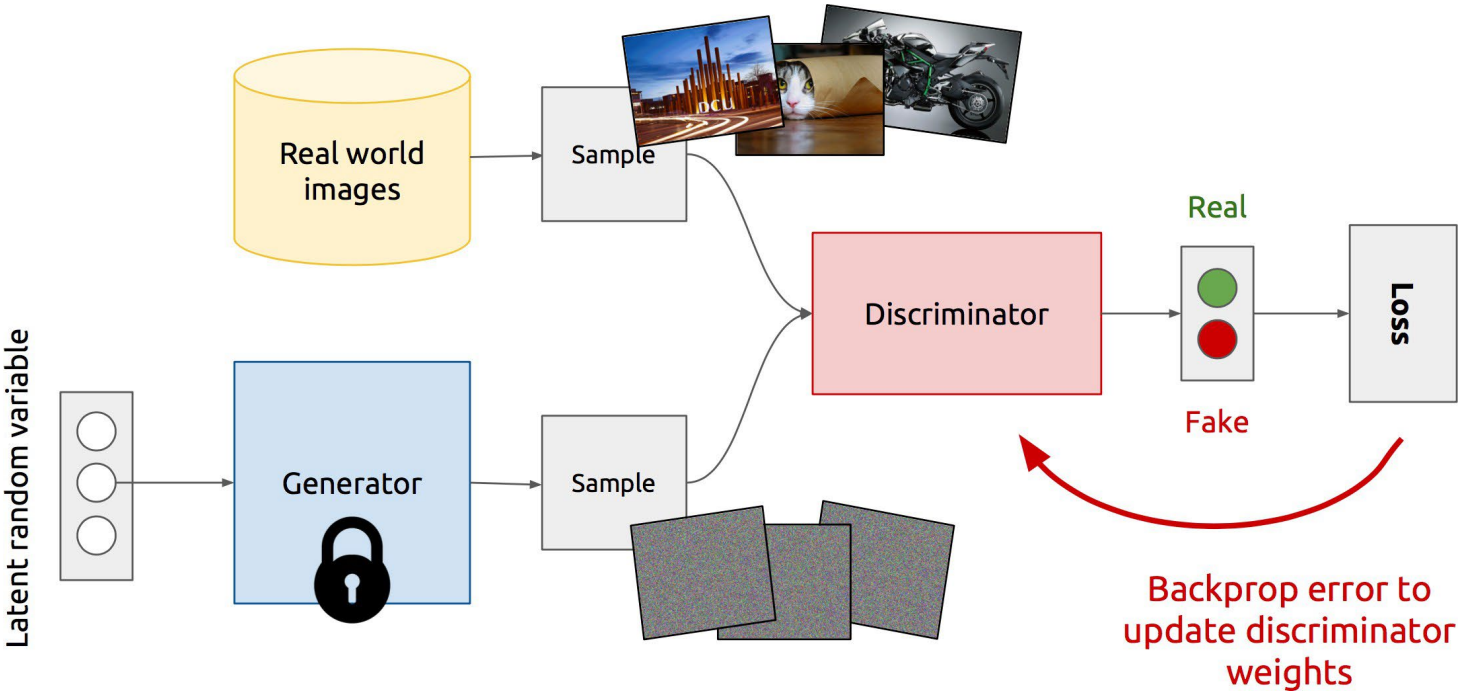
Step 2:
Train the Generator
to beat
the Discriminator.

GAN's Architecture

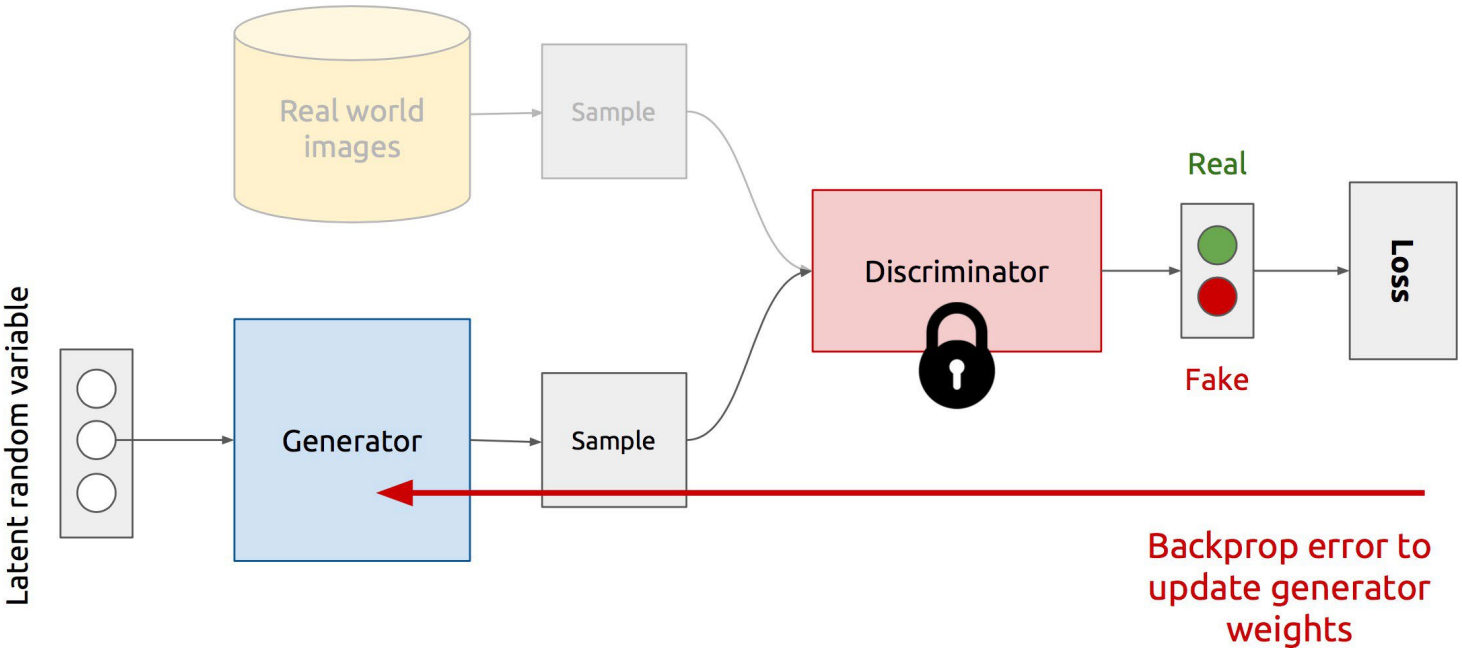


- **Z** is some random noise (Gaussian/Uniform).
- **Z** can be thought as the latent representation of the image.

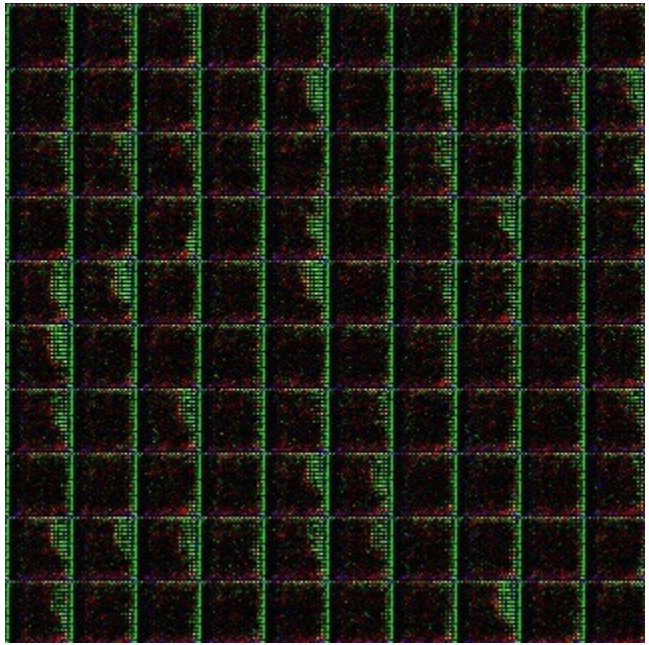
Training Discriminator



Training Generator



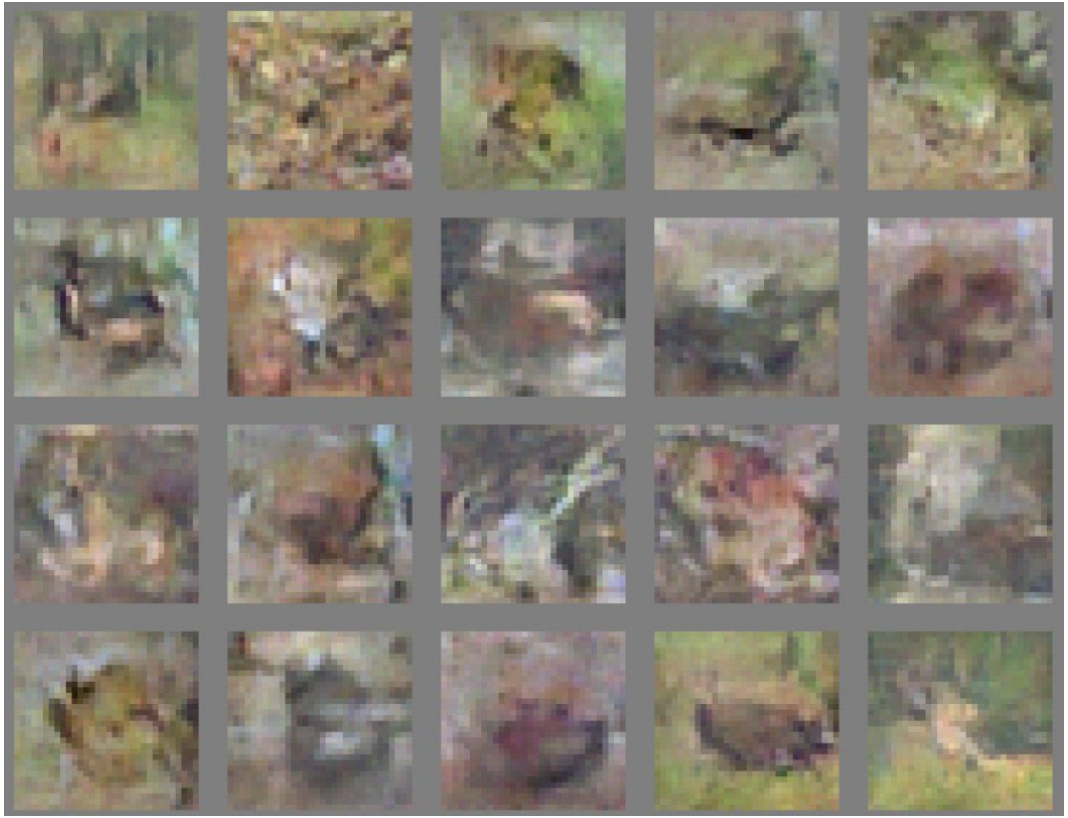
Generator in action



Faces

Goodfellow, Ian, et al. **"Generative adversarial nets."** *Advances in neural information processing systems*. 2014.

CIFAR

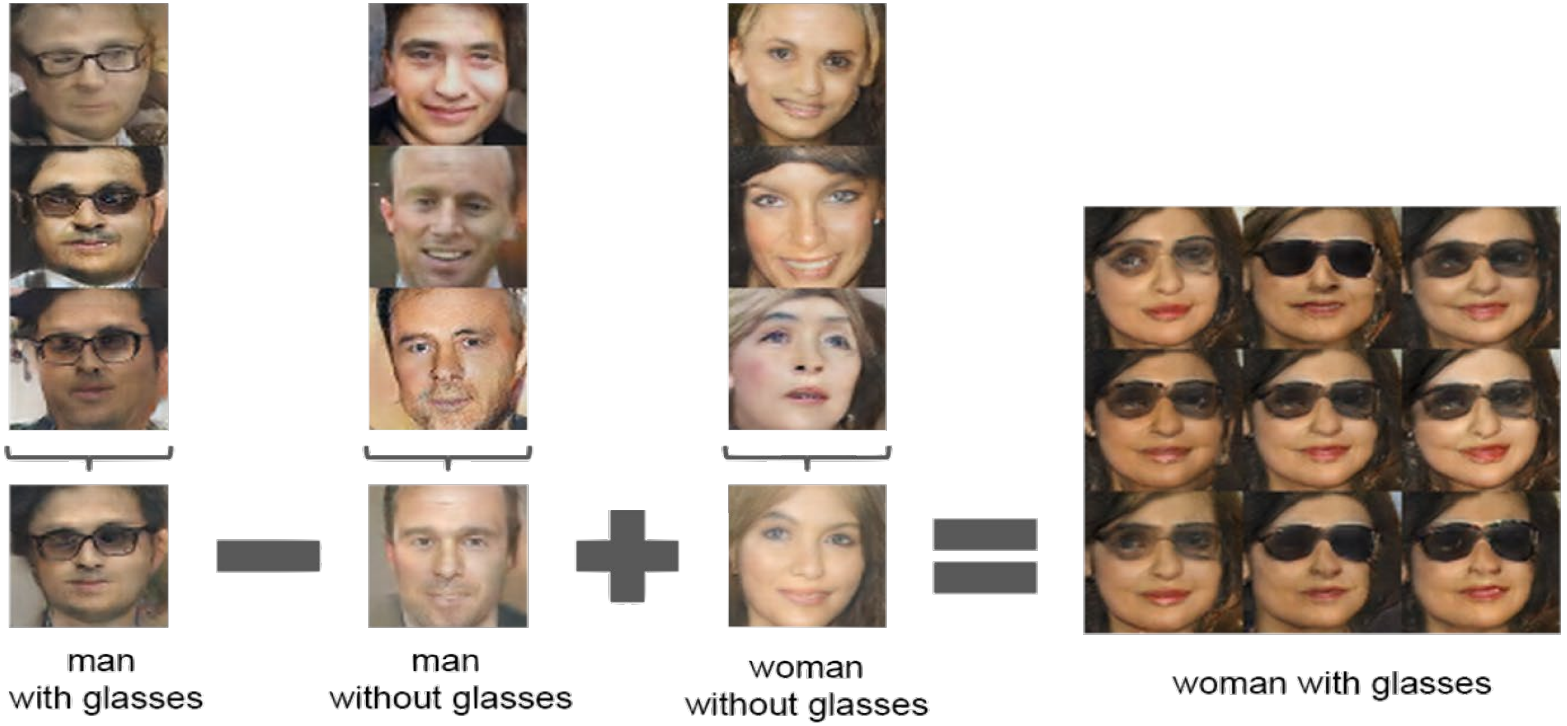


Goodfellow, Ian, et al. **"Generative adversarial nets."** *Advances in neural information processing systems*. 2014.

DCGAN: Bedroom images

Radford, Alec, Luke Metz, and Soumith Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks." arXiv:1511.06434 (2015).

Latent vectors capture interesting patterns…



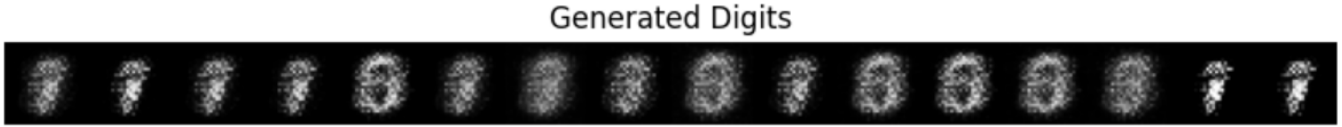
Radford, Alec, Luke Metz, and Soumith Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks." arXiv:1511.06434 (2015).

Example

```python
# Part 1: Import Libraries
import torch
import torch.nn as nn
import torch.optim as optim
from torchvision import datasets, transforms
from torch.utils.data import DataLoader
import matplotlib.pyplot as plt

# Part 2: Hyperparameters
batch_size = 64
lr = 0.0002
z_dim = 100
epochs = 10
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")

# Part 3: Data Loader
transform = transforms.Compose([
    transforms.ToTensor(),
    transforms.Normalize([0.5], [0.5])
])
dataloader = DataLoader(
    datasets.MNIST('.', download=True, transform=transform),
    batch_size=batch_size, shuffle=True
)

# Part 4: Generator Network
class Generator(nn.Module):
    def __init__(self, z_dim):
        super().__init__()
        self.model = nn.Sequential(
            nn.Linear(z_dim, 256),
            nn.ReLU(),
            nn.Linear(256, 784),
            nn.Tanh()  # Outputs between -1 and 1
        )

    def forward(self, z):
        return self.model(z).view(-1, 1, 28, 28)

# Part 5: Discriminator Network
class Discriminator(nn.Module):
    def __init__(self):
        super().__init__()
        self.model = nn.Sequential(
            nn.Flatten(),
            nn.Linear(784, 256),
            nn.LeakyReLU(0.2),
            nn.Linear(256, 1),
            nn.Sigmoid()
        )

    def forward(self, x):
        return self.model(x)
```

```python
# Part 6: Initialize Networks
G = Generator(z_dim).to(device)
D = Discriminator().to(device)

# Optimizers and Loss
criterion = nn.BCELoss()
opt_G = optim.Adam(G.parameters(), lr=lr)
opt_D = optim.Adam(D.parameters(), lr=lr)

# Part 7: Training Loop
for epoch in range(epochs):
    for real_imgs, _ in dataloader:
        real_imgs = real_imgs.to(device)
        batch_size = real_imgs.size(0)

        # Real and Fake labels
        real_labels = torch.ones(batch_size, 1).to(device)
        fake_labels = torch.zeros(batch_size, 1).to(device)

        # --- Train Discriminator ---
        z = torch.randn(batch_size, z_dim).to(device)
        fake_imgs = G(z)

        D_real = D(real_imgs)
        D_fake = D(fake_imgs.detach())

        loss_D = criterion(D_real, real_labels) + criterion(D_fake, fake_labels)

        opt_D.zero_grad()
        loss_D.backward()
        opt_D.step()

        # --- Train Generator ---
        z = torch.randn(batch_size, z_dim).to(device)
        fake_imgs = G(z)
        D_fake = D(fake_imgs)

        loss_G = criterion(D_fake, real_labels)

        opt_G.zero_grad()
        loss_G.backward()
        opt_G.step()

    print(f"Epoch [{epoch+1}/{epochs}] Loss D: {loss_D.item():.4f}, Loss G: {loss_G.item():.4f}")
```

```
100%|██████████| 9.91M/9.91M [00:00<00:00, 56.3MB/s]
100%|██████████| 28.9k/28.9k [00:00<00:00, 1.70MB/s]
100%|██████████| 1.65M/1.65M [00:00<00:00, 14.2MB/s]
100%|██████████| 4.54k/4.54k [00:00<00:00, 5.30MB/s]
Epoch [1/10] Loss D: 1.0971, Loss G: 0.8736
Epoch [2/10] Loss D: 0.4397, Loss G: 2.0745
Epoch [3/10] Loss D: 1.3601, Loss G: 0.8748
Epoch [4/10] Loss D: 0.5082, Loss G: 1.6825
Epoch [5/10] Loss D: 1.0124, Loss G: 1.0934
Epoch [6/10] Loss D: 0.9316, Loss G: 1.0692
Epoch [7/10] Loss D: 0.9583, Loss G: 1.2154
Epoch [8/10] Loss D: 1.0815, Loss G: 1.2540
Epoch [9/10] Loss D: 0.9804, Loss G: 1.1316
Epoch [10/10] Loss D: 0.8182, Loss G: 1.0831
```



1. **Early Epochs**: Images look like noise—GAN hasn't learned much yet.
2. **Mid Epochs**: Some digit-like shapes begin to emerge.
3. **Later Epochs**: Many images start to resemble MNIST digits, but there are occasional artifacts.

In class assignment

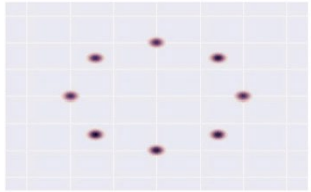
Problems with GANs

- **Mode-Collapse**

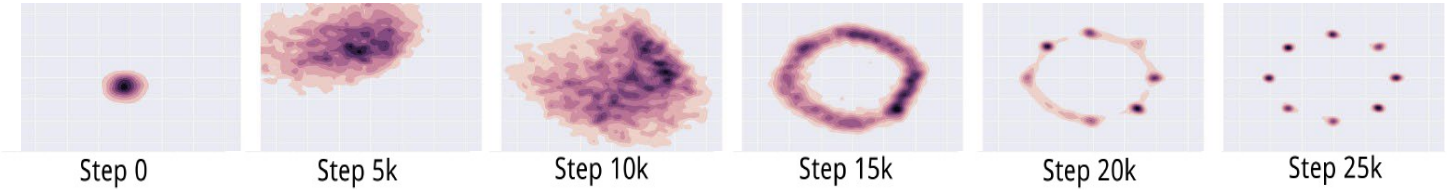
Mode-Collapse

- **Generator fails to output diverse samples**

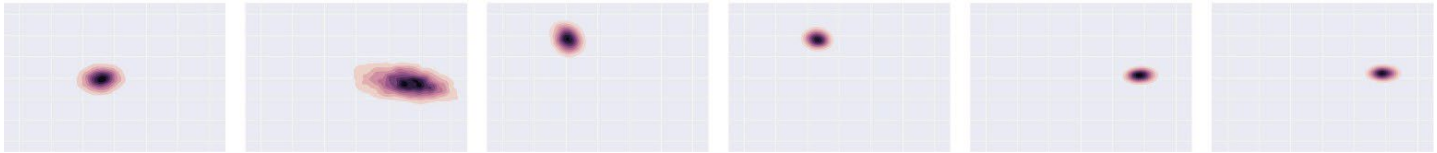
Target



Expected

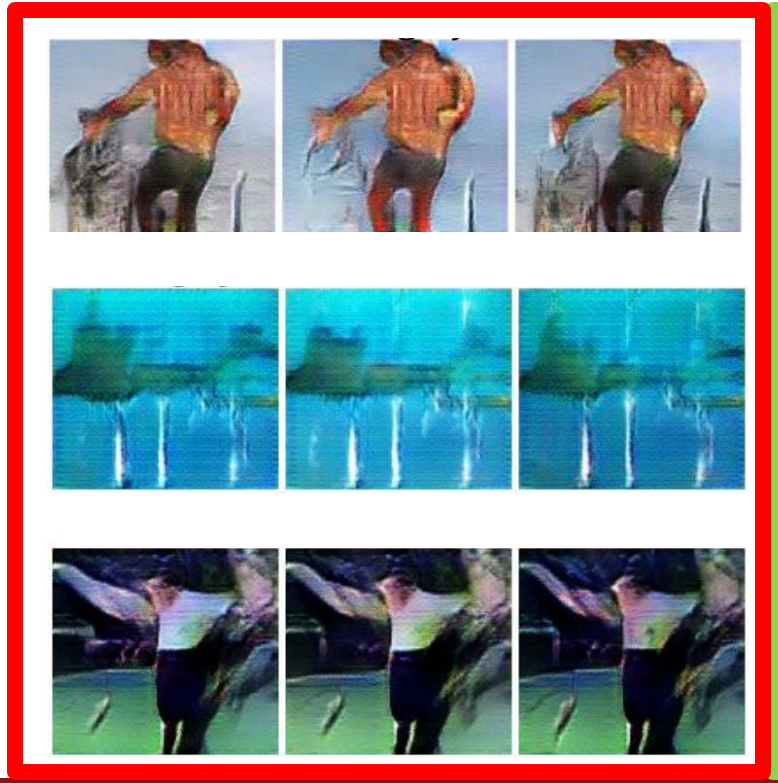


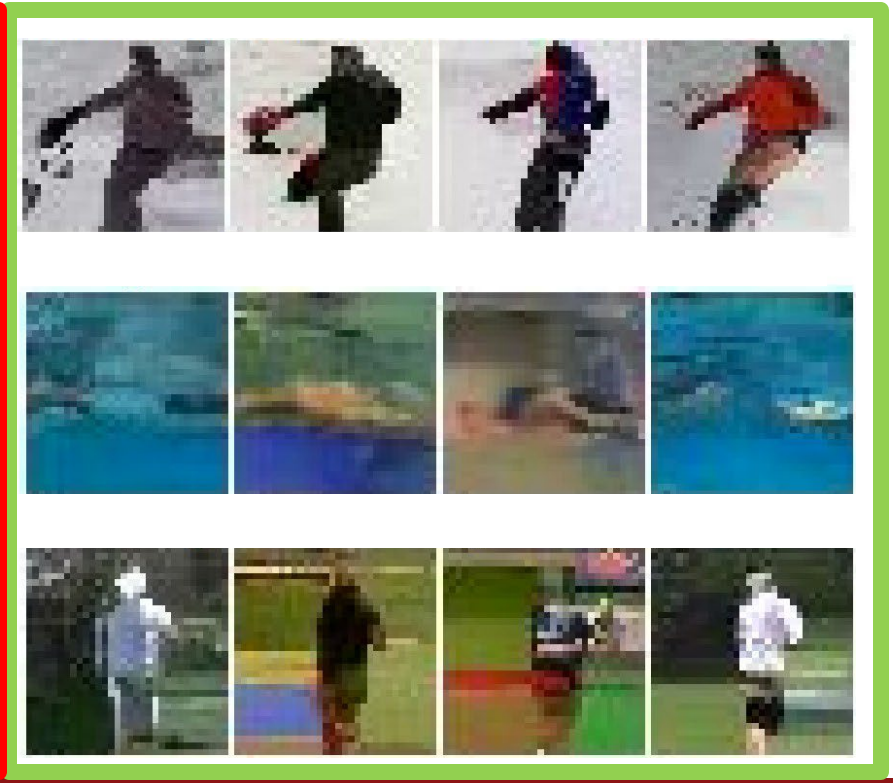
Output



Metz, Luke, et al. **"Unrolled Generative Adversarial Networks."** arXiv preprint arXiv:1611.02163 (2016).

Some real examples





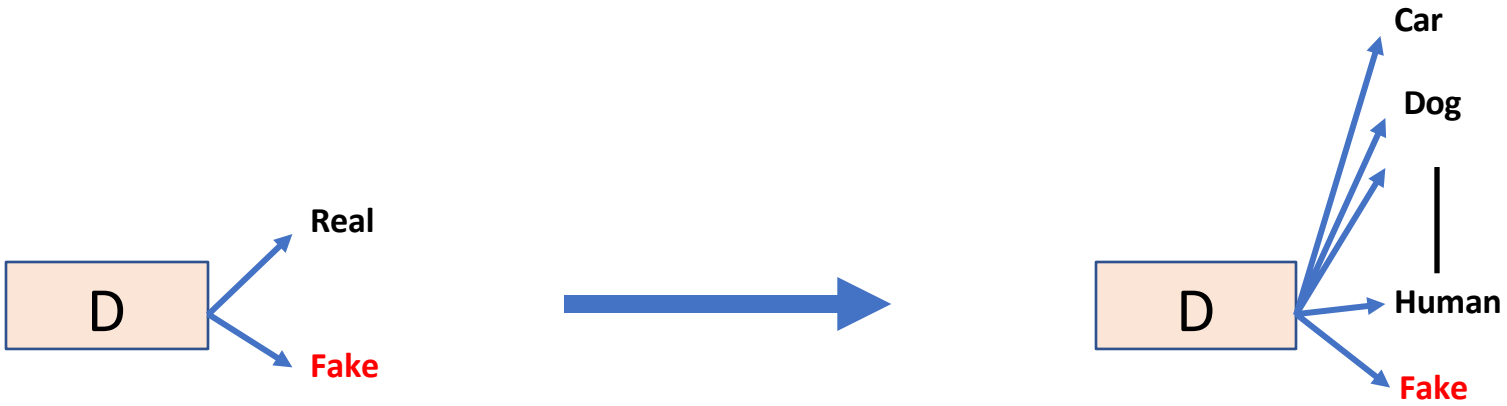
Reed, S., et al. ***Generating interpretable images with controllable structure.*** Technical report, 2016. 2, 2016.

Basic (Heuristic) Solutions

- **Supervision with labels**

Supervision with Labels

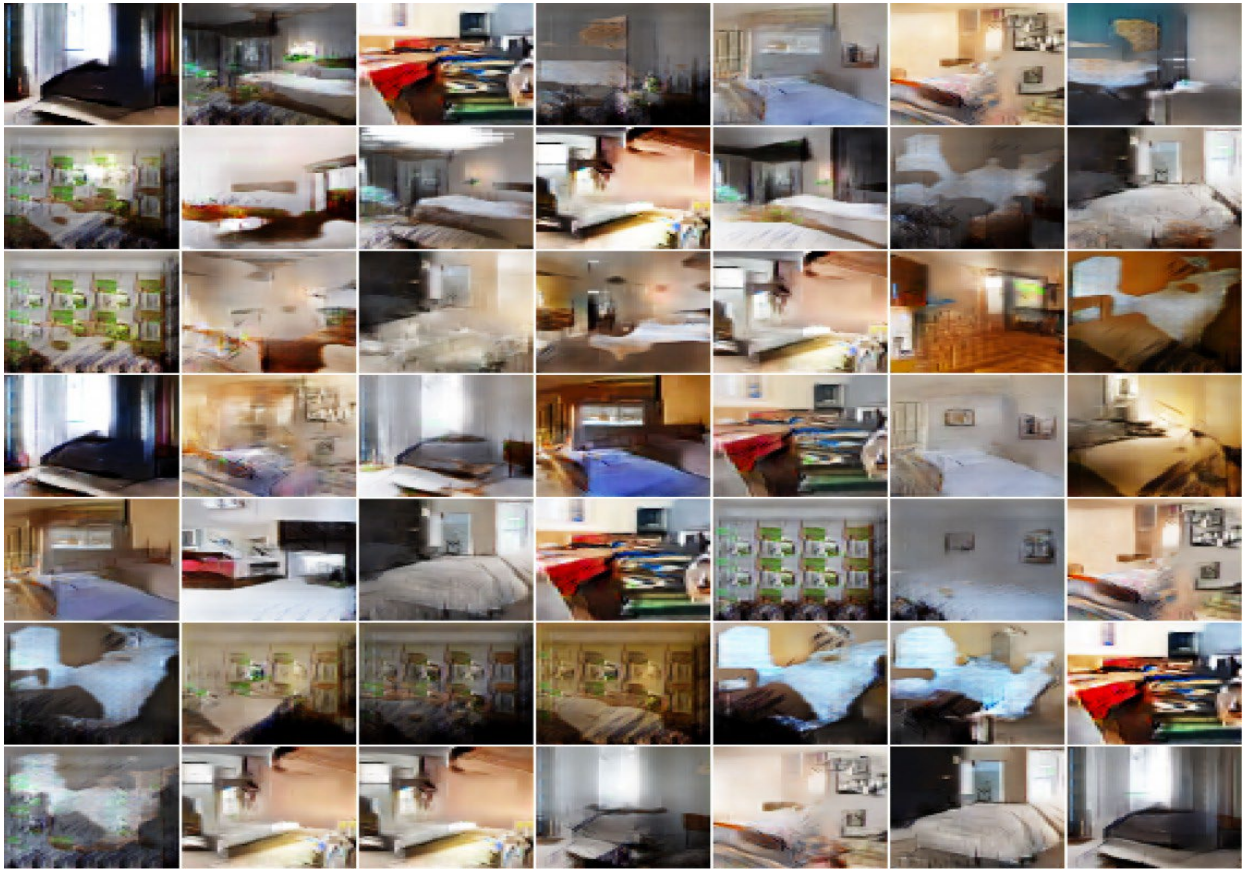
- Label information of the real data might help



- Empirically generates much better samples

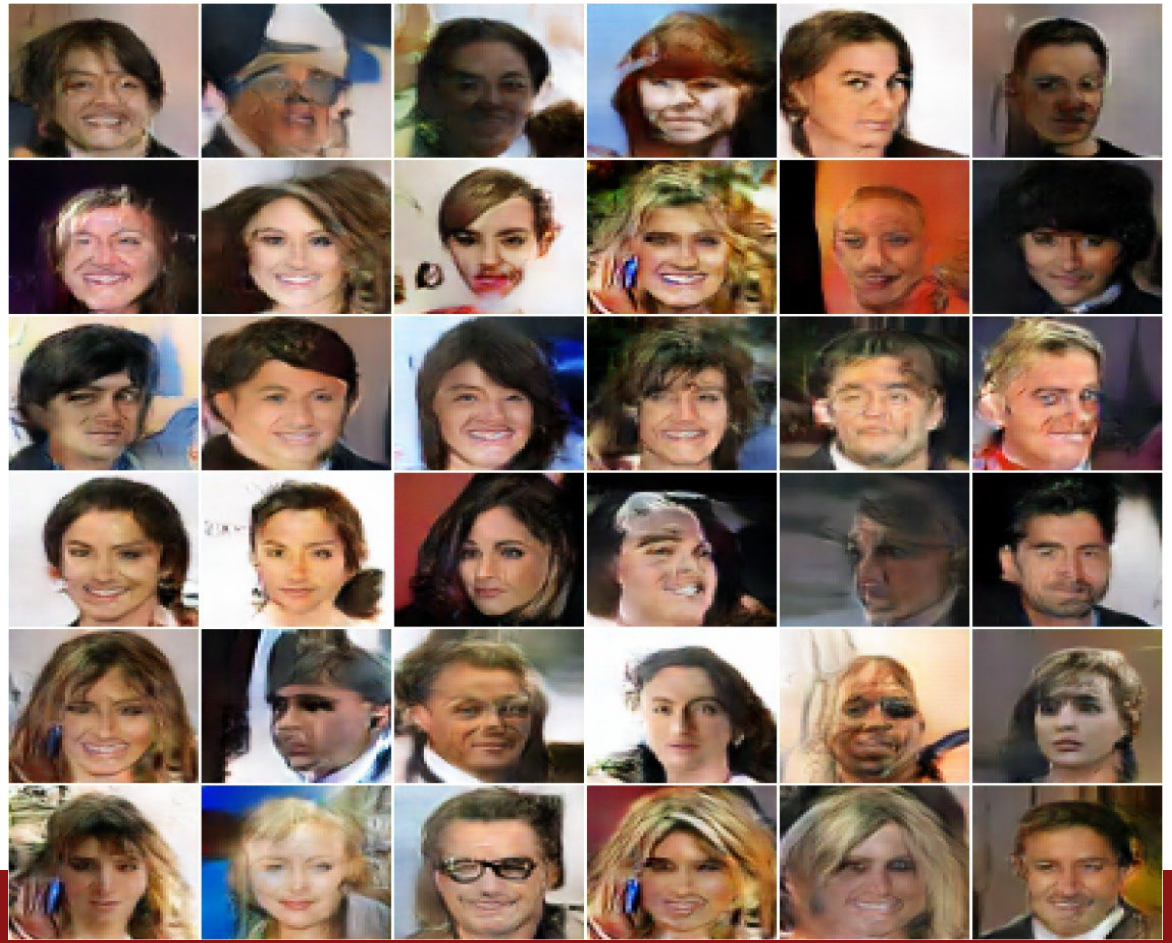
Salimans, Tim, et al. "Improved techniques for training gans." *Advances in Neural Information Processing Systems*. 2016.

More Bedrooms…



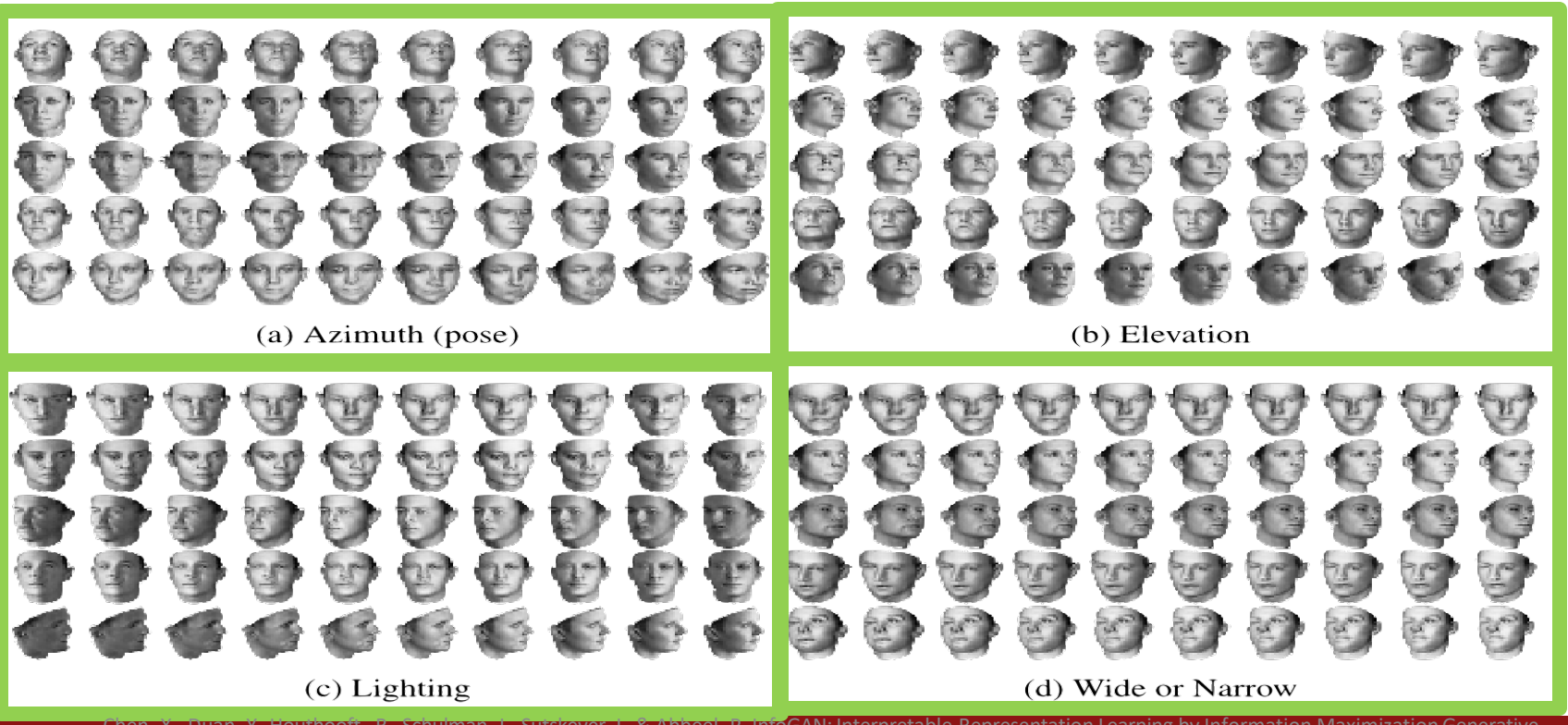
Zhao, Junbo, Michael Mathieu, and Yann LeCun. **"Energy-based generative adversarial network."** arXiv preprint arXiv:1609.03126 (2016)

Celebs…



The Cool Stuff…

3D Faces



Chen, X., Duan, Y., Houthooft, R., Schulman, J., Sutskever, I., & Abbeel, P. InfoGAN: Interpretable Representation Learning by Information Maximization Generative Adversarial Nets, NIPS (2016).

Image-to-Image Translation

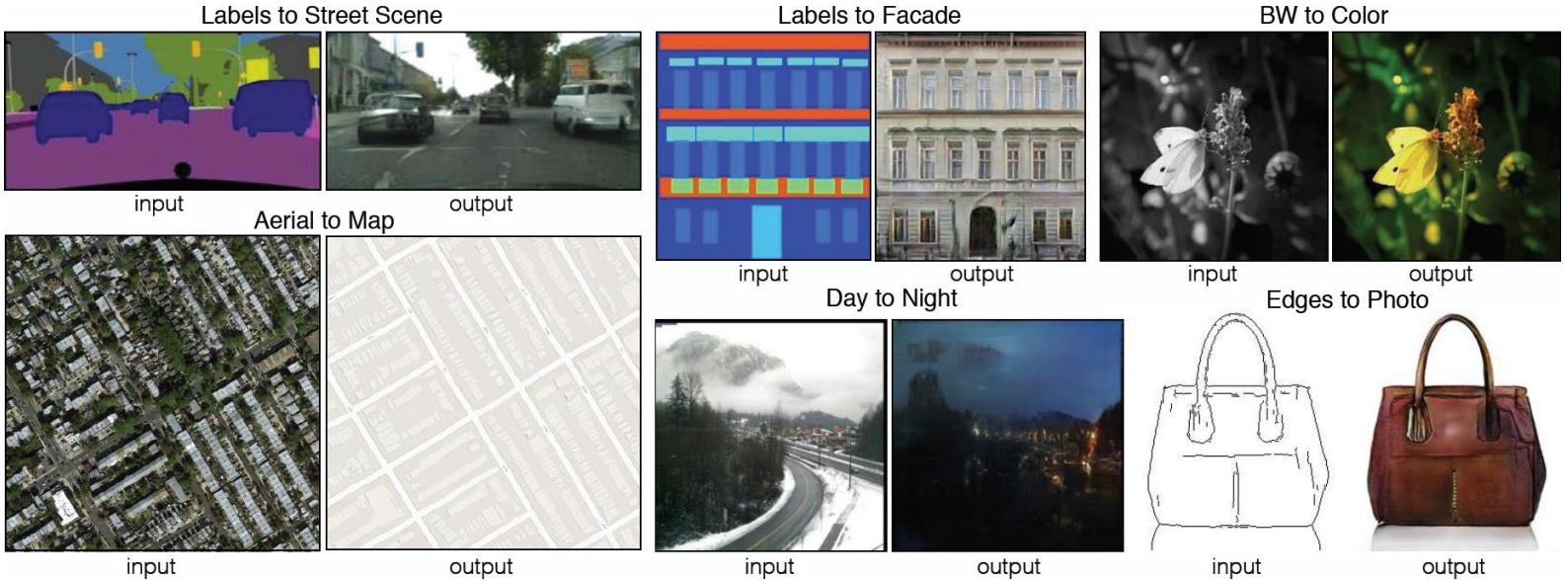


Figure 1 in the original paper.

[Link to an interactive demo of this paper]

Isola, P., Zhu, J. Y., Zhou, T., & Efros, A. A. "**Image-to-image translation with conditional adversarial networks**". arXiv preprint arXiv:1611.07004. (2016).

Image-to-Image Translation

- Architecture: *DCGAN*-based architecture
- Training is conditioned on the images from the source domain.
- Conditional GANs provide an effective way to handle many complex domains without worrying about designing *structured loss* functions explicitly.

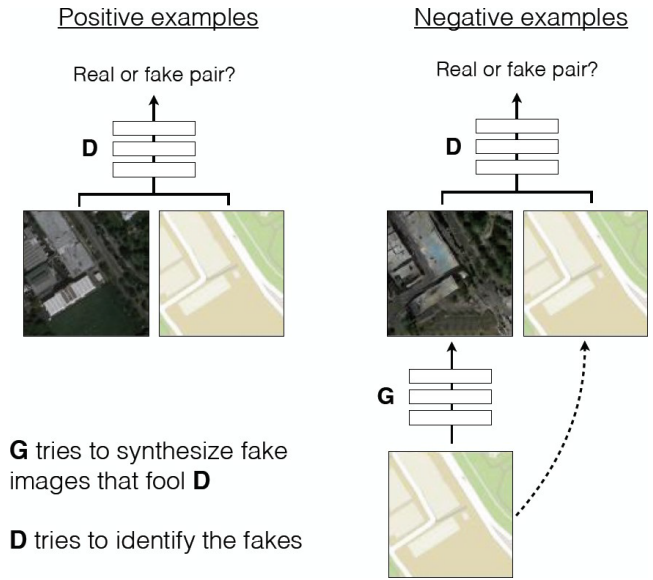


Figure 2 in the original paper.

Isola, P., Zhu, J. Y., Zhou, T., & Efros, A. A. "**Image-to-image translation with conditional adversarial networks**". arXiv preprint arXiv:1611.07004. (2016).

Text-to-Image Synthesis

Motivation

Given a text description, generate images closely associated.

Uses a conditional GAN with the generator and discriminator being condition on “dense” text embedding.

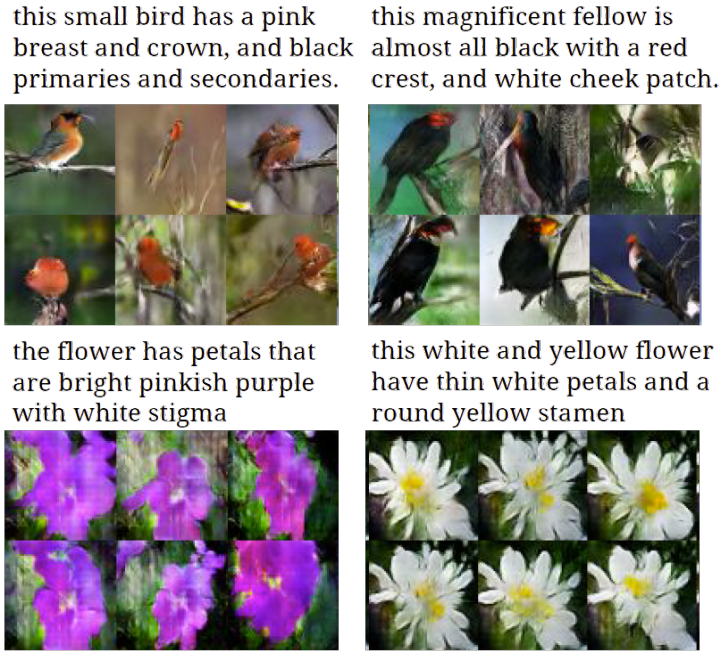


Figure 1 in the original paper.

Text-to-Image Synthesis

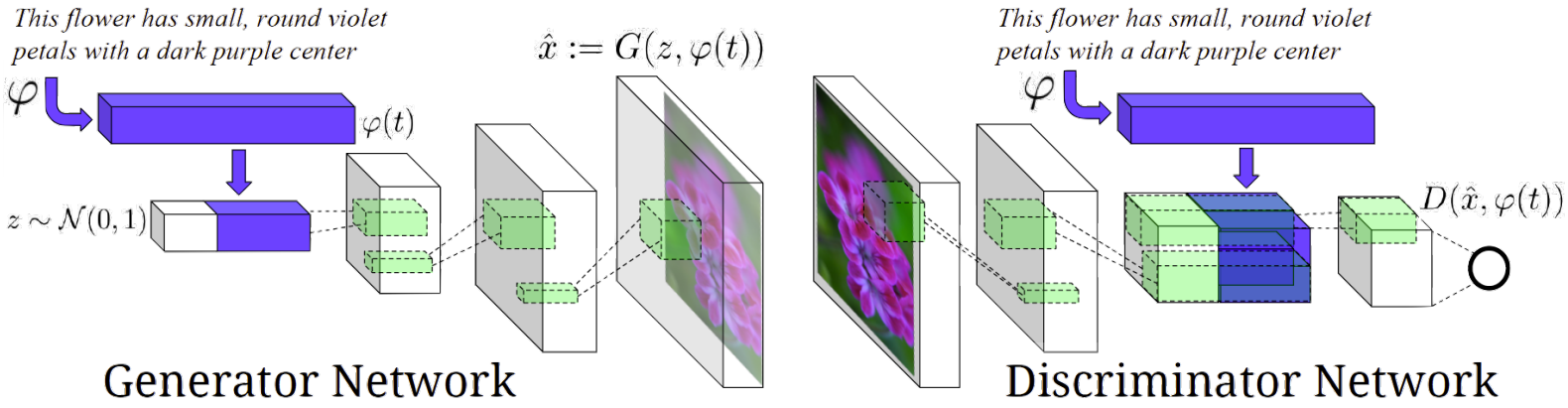


Figure 2 in the original paper.

Positive Example:

Real Image, Right Text

Negative Examples:

Real Image, Wrong Text

Fake Image, Right Text

Reed, S., Akata, Z., Yan, X., Logeswaran, L., Schiele, B., & Lee, H. "**Generative adversarial text to image synthesis**". ICML (2016).

Face Aging with Conditional GANs

- Differentiating Feature: Uses an *Identity Preservation Optimization* using an auxiliary network to get a better approximation of the latent code (z*) for an input image.
- Latent code is then conditioned on a discrete (one-hot) embedding of age categories.

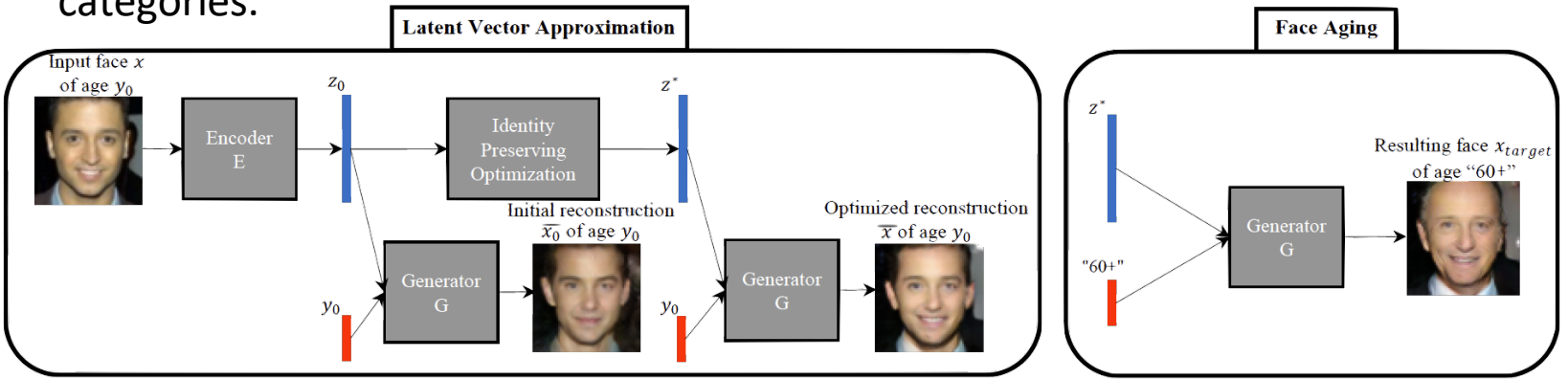


Figure 1 in the original paper.

Face Aging with Conditional GANs

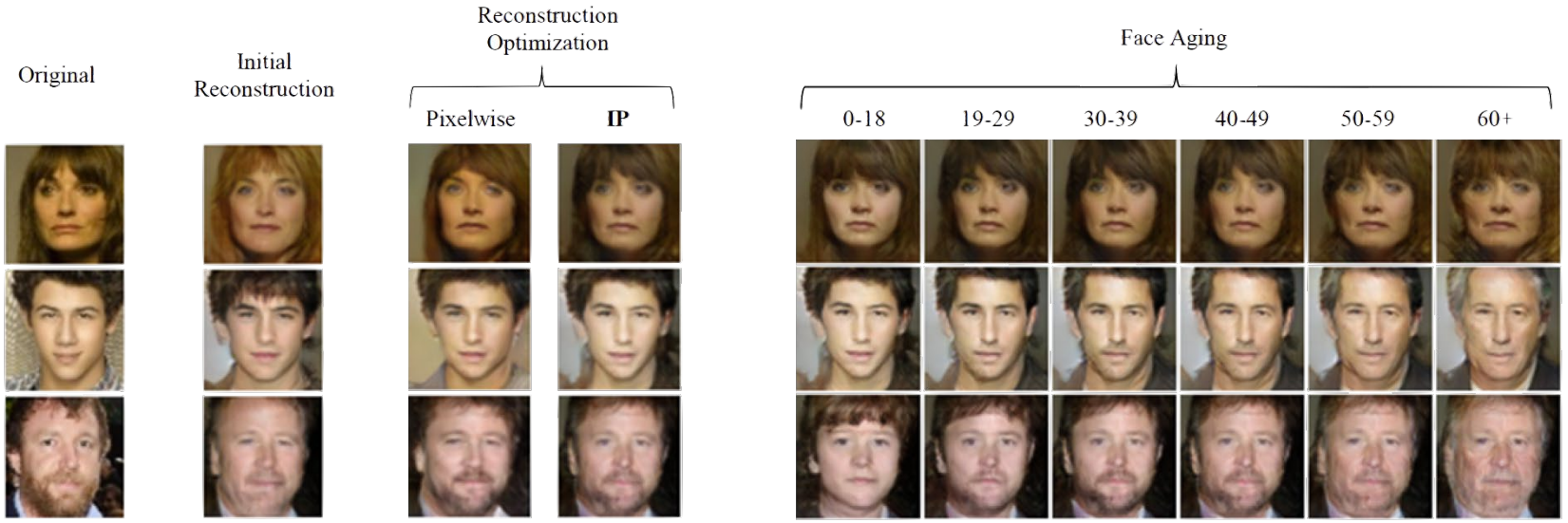


Figure 3 in the original paper.

Antipov, G., Baccouche, M., & Dugelay, J. L. (2017). "**Face Aging With Conditional Generative Adversarial Networks**". arXiv preprint arXiv:1702.01983.

Coupled GAN

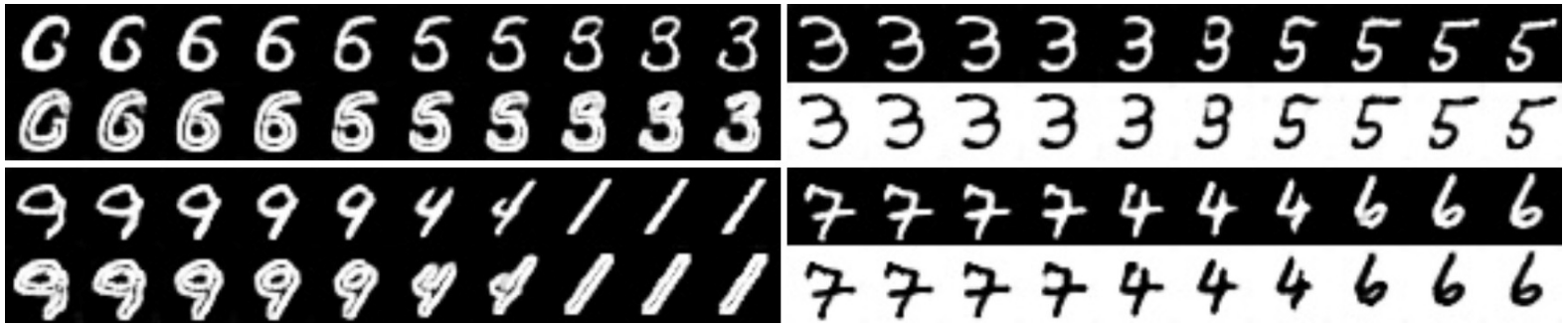


Figure 2 in the original paper.

- Learning a *joint distribution* of *multi-domain* images.
- Using GANs to learn the joint distribution with samples drawn from the marginal distributions.
- Direct applications in domain adaptation and image translation.

Liu, Ming-Yu, and Oncel Tuzel. "**Coupled generative adversarial networks**". NIPS (2016).

Coupled GANs

- Some examples of generating facial images across different feature domains.

- Corresponding images in a column are generate from the same latent code z

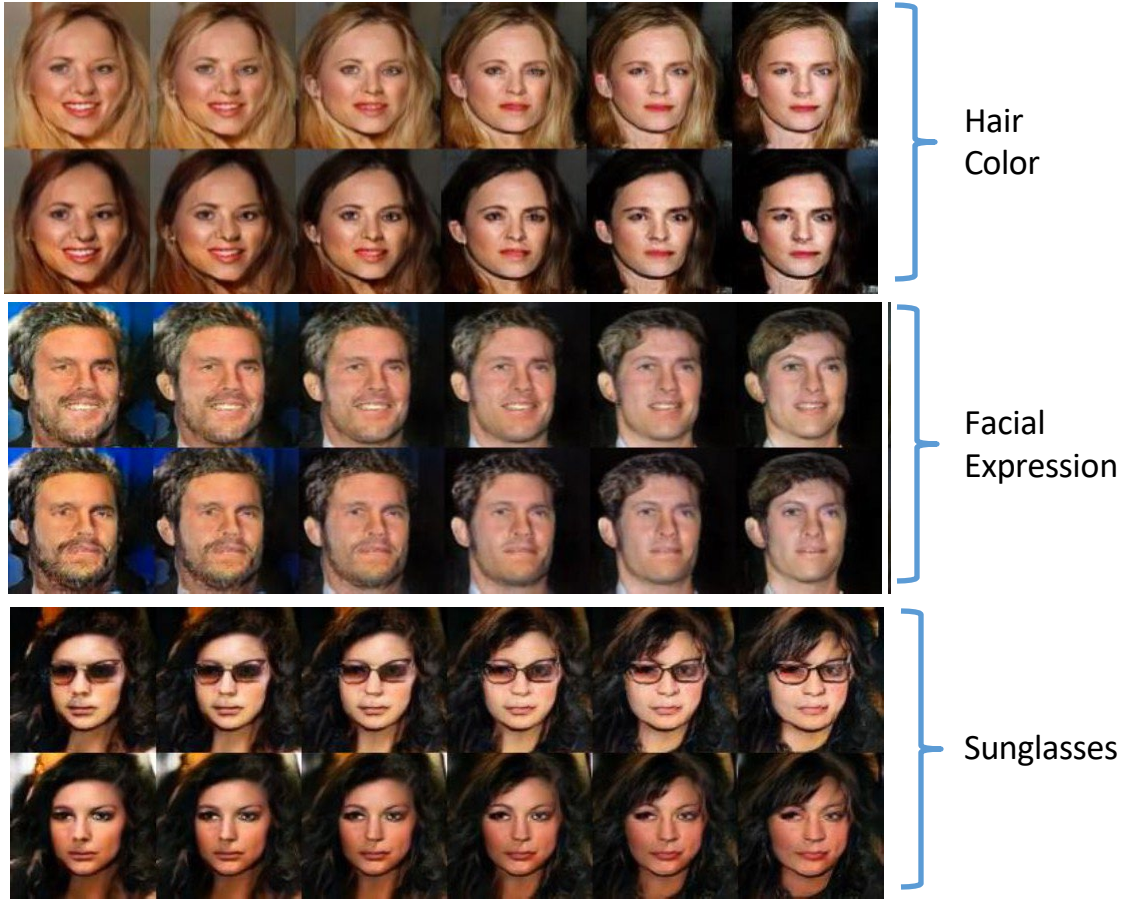


Figure 4 in the original paper.

Laplacian Pyramid of Adversarial Networks

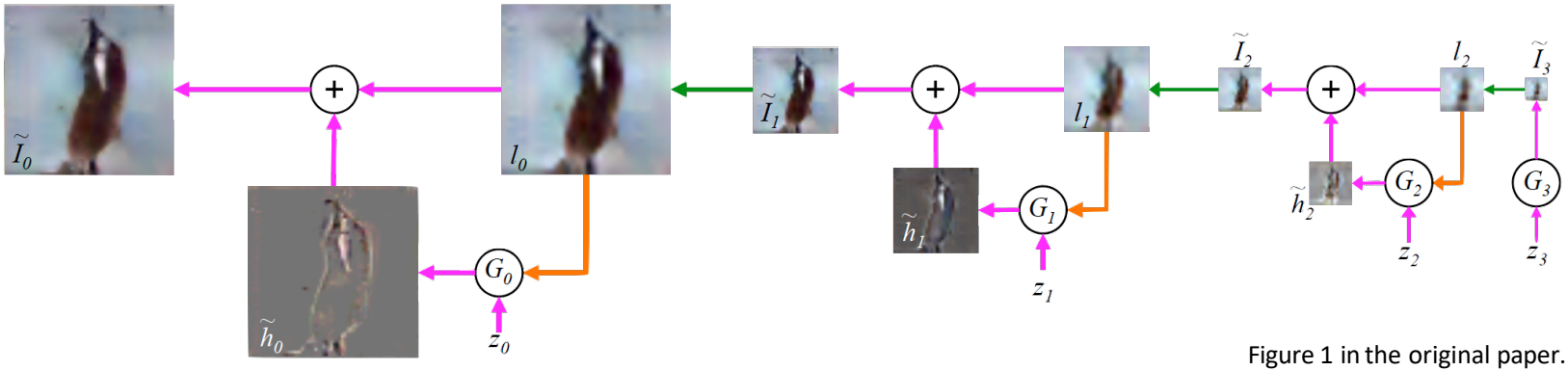


Figure 1 in the original paper.

Image Generation using a LAPGAN

- Generator G_0 generates the base image $\tilde{I}_0$ from random noise input z_0.
- Generators (G_1, G_2, G_3) iteratively generate the *difference image* ($\tilde{h}$) conditioned on previous small image (l).
- This *difference image* is added to an up-scaled version of previous smaller image.

Laplacian Pyramid of Adversarial Networks

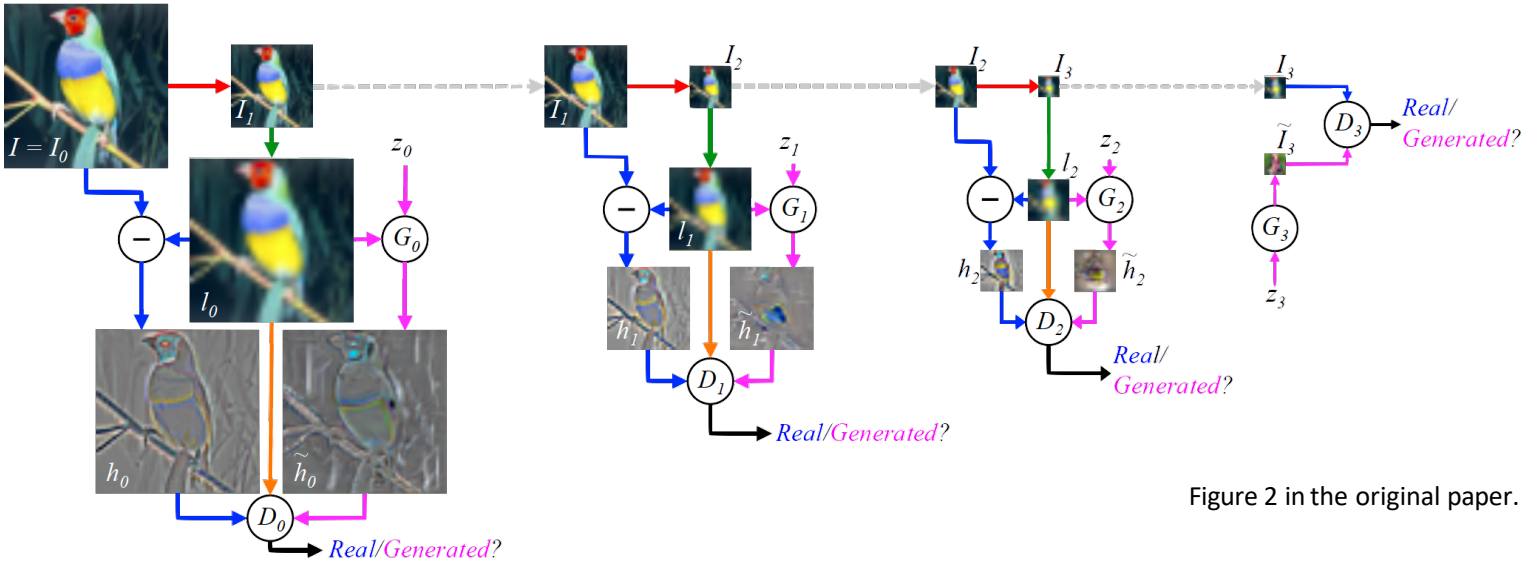


Figure 2 in the original paper.

Training Procedure:

Models at each level are trained independently to learn the required representation.

Summary

- GANs are generative models that are implemented using two stochastic neural network modules: **Generator** and **Discriminator**.
- **Generator** tries to generate samples from random noise as input
- **Discriminator** tries to distinguish the samples from Generator and samples from the real data distribution.
- Both networks are trained adversarially (in tandem) to fool the other component. In this process, both models become better at their respective tasks.

Why use GANs for Generation?

- Can be trained using back-propagation for Neural Network based Generator/Discriminator functions.
- Sharper images can be generated.
- Faster to sample from the model distribution: *single* forward pass generates a *single* sample.

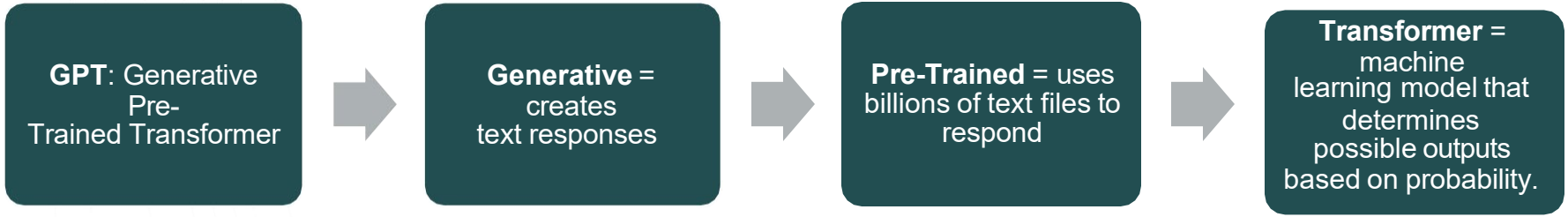
WHAT IS GENERATIVE AI & WHAT CAN IT DO?

GENERATIVE AI TOOLS

ChatGPT	• Nov. 2022 • Developed by OpenAI • https://chat.openai.com/
Bing AI	• Feb. 2023 • Developed by Microsoft • https://www.bing.com
Google Bard	• Mar. 2023 • Developed by Google • https://bard.google.com/
Others	• Notion, Jasper, DALL-E, and many more.

HOW DOES GENERATIVE AI WORK?

- Generative AI (Gen AI): Refers to deep-learning models that can generate high-quality text, images, and other content based on the data they are trained on. (IBM)
- ChatGPT: Large Language Model-based chatbot



HOW DOES GENERATIVE AI WORK?

Step One

- User inputs a prompt or question into ChatGPT
- ***Prompt**: Write a 150-word essay on the Impressionism movement in 19th century France and its characteristics.*

Step Two

- ChatGPT uses a deep learning algorithm to analyze the prompt and generate a response

Step Three

- ChatGPT identifies keywords and phrases in the prompt to understand the context of the question
- ***Command:** Write an essay in 150-words*
- ***Topic:** Impressionism characteristics, 19th century France*

Step Four

- ChatGPT accesses its database of information to find relevant information related to the prompt.
- *What does it know about writing essays? What does it know about the time, country, and characteristics of this artistic movement?*

Step Five

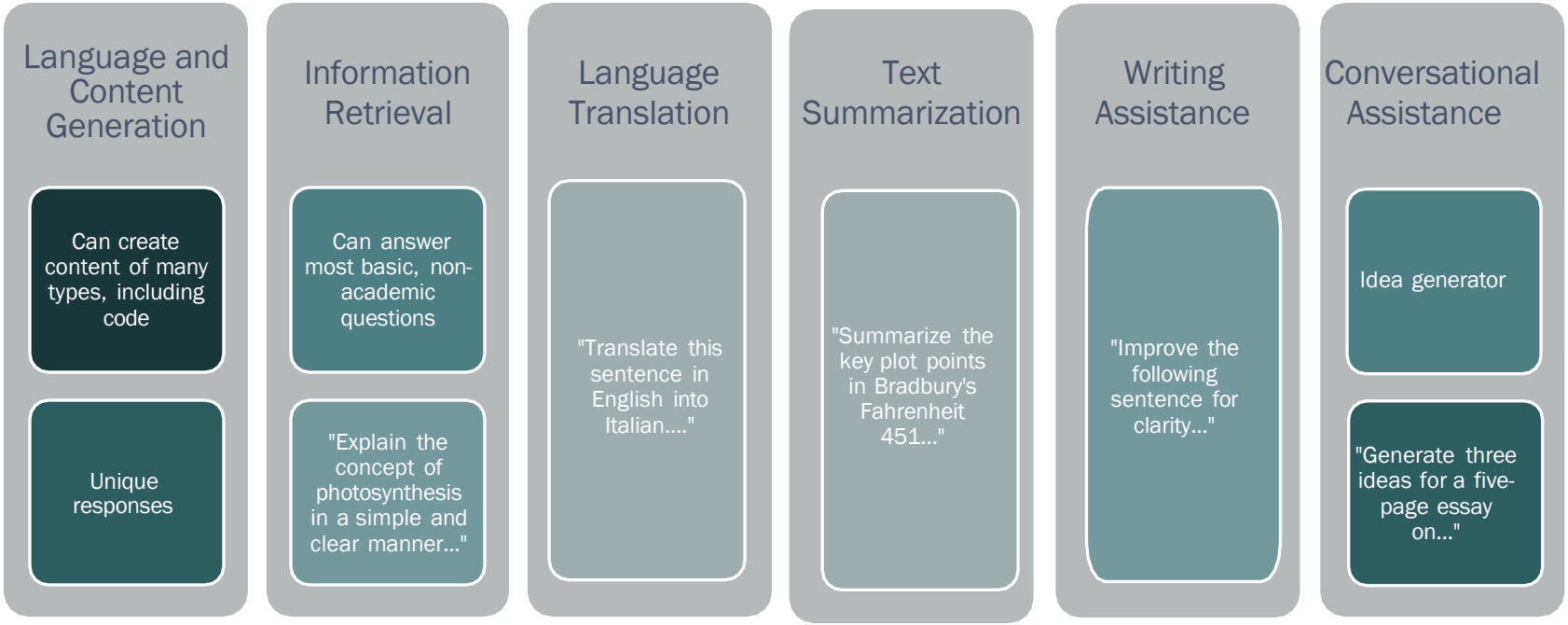
- ChatGPT uses natural language processing to generate a response that is grammatically correct and contextually relevant to its dataset

Step Six

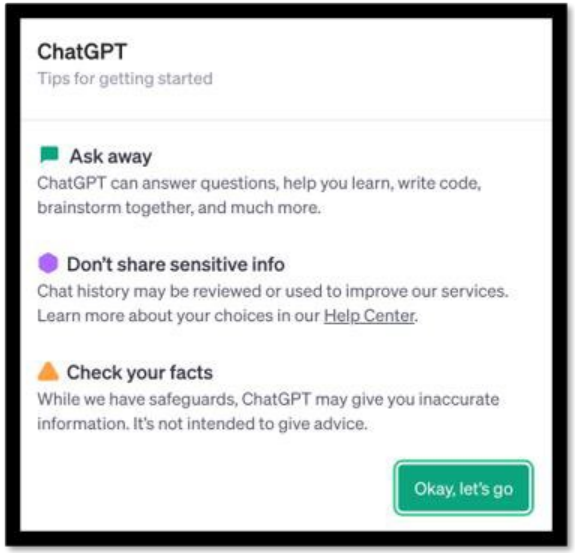
- ChatGPT presents the response to the user, who can choose to continue the conversation or provide additional prompts.

Adapted and taken from Compose & Consult: https://composeandconsult.com/the-edit-effect/ai-platforms-hallucination-effect-content-generation

GENERATIVE AI: FUNCTIONS



GENERATIVE AI: LIMITATIONS



- "ChatGPT may produce inaccurate information about people, places, or facts." - *ChatGPT disclaimer*

Incorrect or nonsensical responses

- Produces hallucinations
- Can not verify the accuracy of information (disinformation)

Over reliant on the data it is trained on: information contains biases

Doesn't have genuine comprehension, generates text based on patterns in data

GENERATIVE AI CONSIDERATIONS

- Lack of regulations on the technology
  - Potential copyright issues & ownership of content
- Admin/Educators are still working on developing policies specific to GenAI across the nation
- When using GenAI, provide clear, concise directions or instructions
  - Rephrase prompts if you aren't getting a desired result
- Use any generative AI tool with caution
  - Avoid sharing sensitive information
  - Get permission before using another's intellectual property for prompts

GENERATIVE AI CONSIDERATIONS

- Encourage dialogue surrounding GenAI in the classroom
- Provide your students with clear communication and expectations
- Consider ethical and legal issues when using AI tools Recognize that AI detection tools are not perfect Understand the inevitability of advancing AI technology

EXPLORING GEN AI IN THE CLASSROOM

Bias
- Discriminatory outputs

Environment
- Mining, energy consumption, and waste

Academic Integrity
- Plagiarism, cheating

Copyright
- Infringement on intellectual property rights

Privacy
- Personal data collection

Datafication
- Commodification of personal data

Human Labor
- Job automation and exploitation

Power
- Global power imbalances, structural inequalities

Adapted and taken from Leon Furze's"Teaching AI Ethics" series: https://leonfurze.com/2023/01/26/teaching-ai-ethics/

WRAP-UP

Key Take-Aways

- This technology is not going away and will continue to grow in its applications
- It isn't "smart" as it does not have the ability to comprehend its output or data
- It is not the end of traditional writing or skills
- Instead, it may help us determine which skills are unique to humans and which can be automated
- Consider GenAI as a potential brainstorming partner, not a fact finder