Generative Modelling with GANs

https://arxiv.org/pdf/1701.00160.pdf

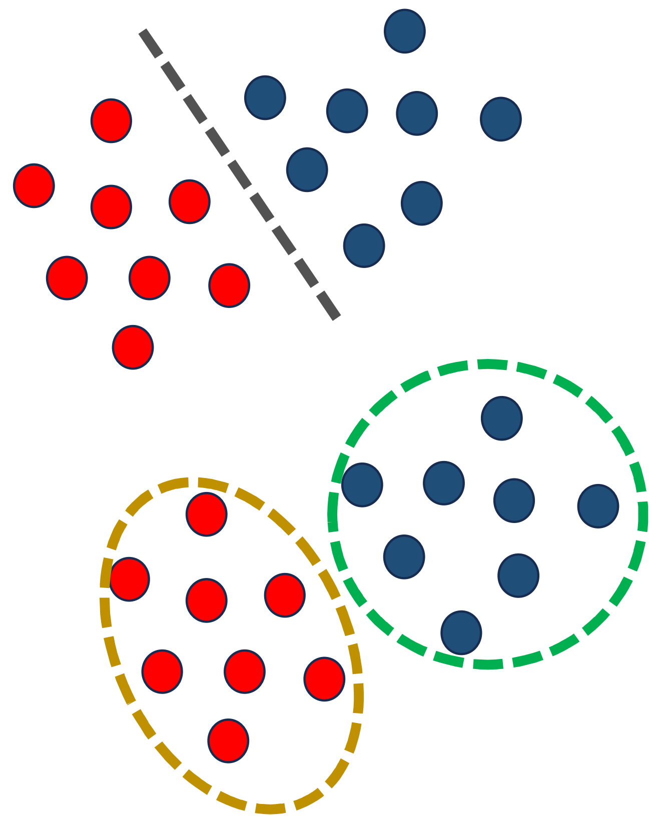
Real vs Fake

Two imaginary celebrities that were dreamed up by a random number generator

Image Source: [Nvidia Research](https://research.nvidia.com/publication/2018-04_progressive-growing-gans-improved-quality-stability-and-variation)

https://research.nvidia.com/publication/2018-04_progressive-growing-gans-improved-quality-stability-and-variation

Generative vs. Discriminative Model



Discriminative Models:

- **Goal**: Learns to distinguish between different classes of data
- **Applications**: Models the decision boundary between the classes. Learns $P(Y|X)$

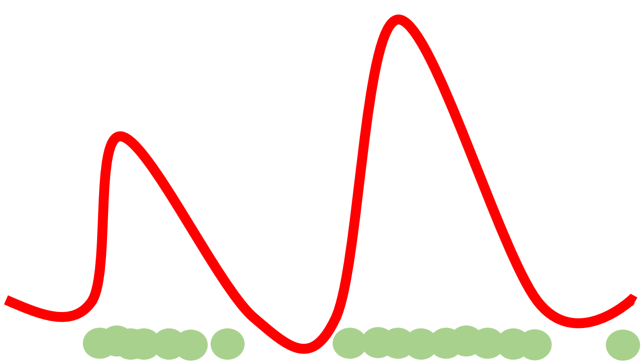
Generative Models:

- **Goal**: Learns the distribution of data and generate new data from the distribution
- **Applications**: Models the actual distribution of each class. Learns $P(X,Y)$

Generative Modelling

- Takes as input training samples from some distribution and learn a model that represents that distribution
- Output is some representation of probability distribution which defines the sample space

Density Estimation



Samples

Sample Generation

Input Samples

Training data ~ $P_{data}(x)$

Generated Samples

Generated ~ $P_{model}(x)$

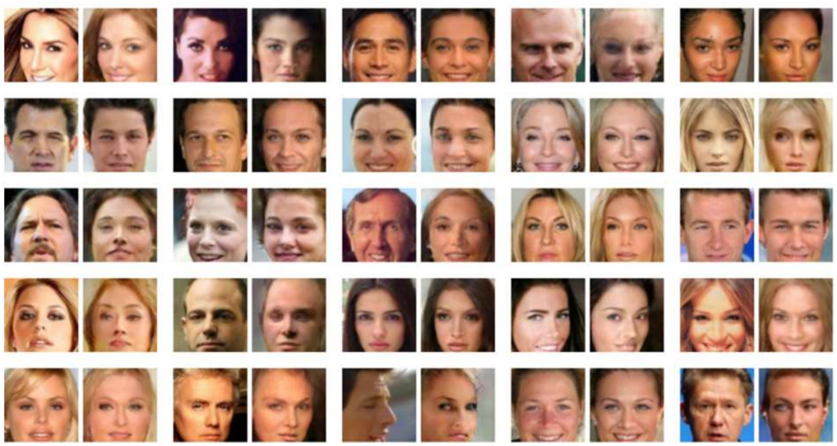
How can we learn $P_{model}(x)$ similar to $P_{data}(x)$?

Why we care about Generative modelling?

❖ **Data Augmentation:** Capable of uncovering the **underlying features** in a dataset and generate synthetic data

Why we care about Generative modelling?

- **Debiasing:** Using the **underlying features** in a dataset, to build more fair and representative datasets



Homogenous skin colour, pose

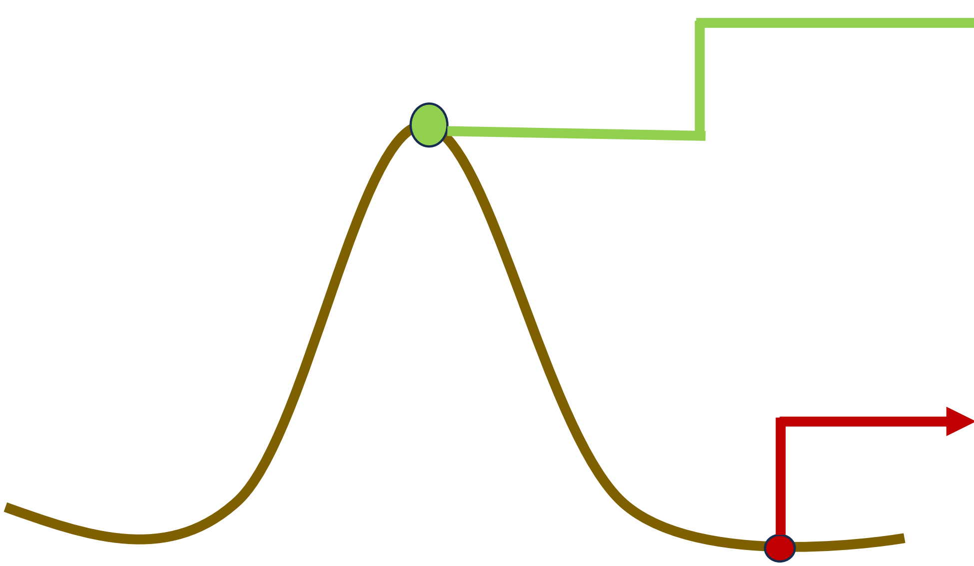
vs.

Diverse skin colour, pose, illumination

Why we care about Generative modelling?

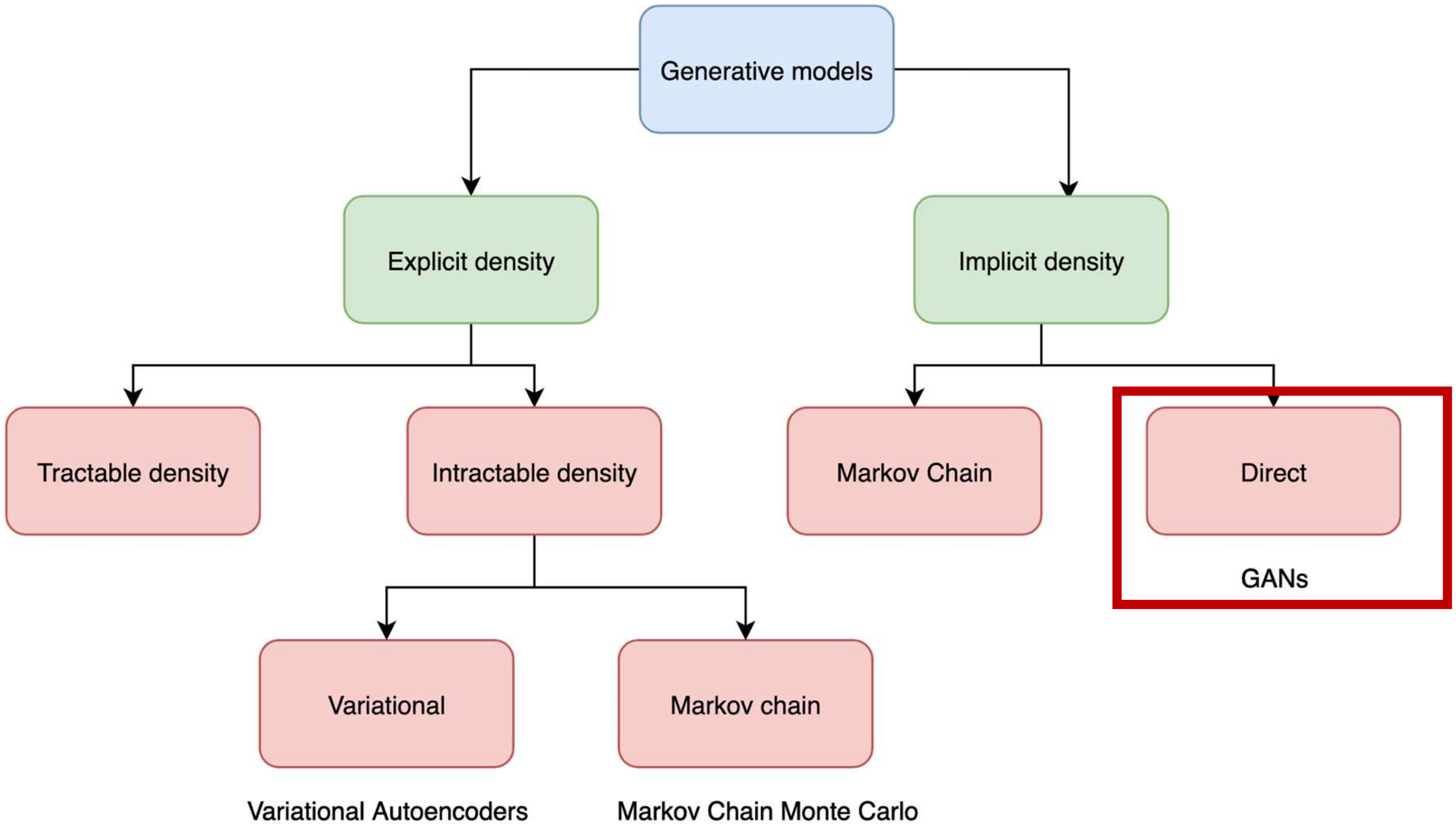
- **Outlier Detection:** Can detect outliers in the distribution. The outliers can be used during training to improve more.

95% of Driving Data:
highway, straight road, fair weather



Detecting outliers to avoid unpredictable behaviour when training

Generative Model - Prominent Types



Why GANs?

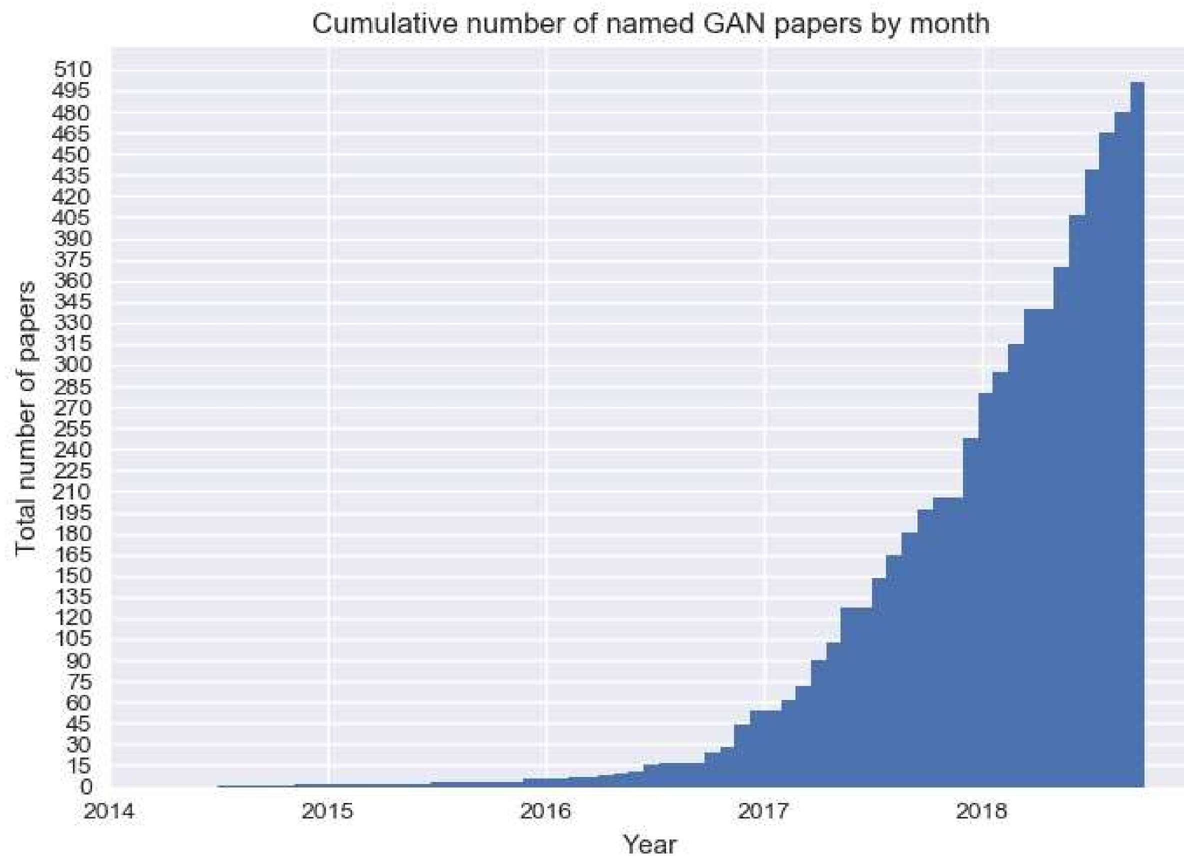


Image Source: https://github.com/hindupuravinash/the-gan-zoo

Generative Adversarial Networks (GANs)

Generative Adversarial Networks

- ❑ Generative Models: We try to learn the underlying distribution from which our dataset comes from.
- ❑ Adversarial Training: GANs are made up of two competing networks (adversaries) that are trying to beat each other
- ❑ Neural Networks

- Generative Adversarial Networks (GANs) are a way to make a generative model by having two neural networks compete with each other

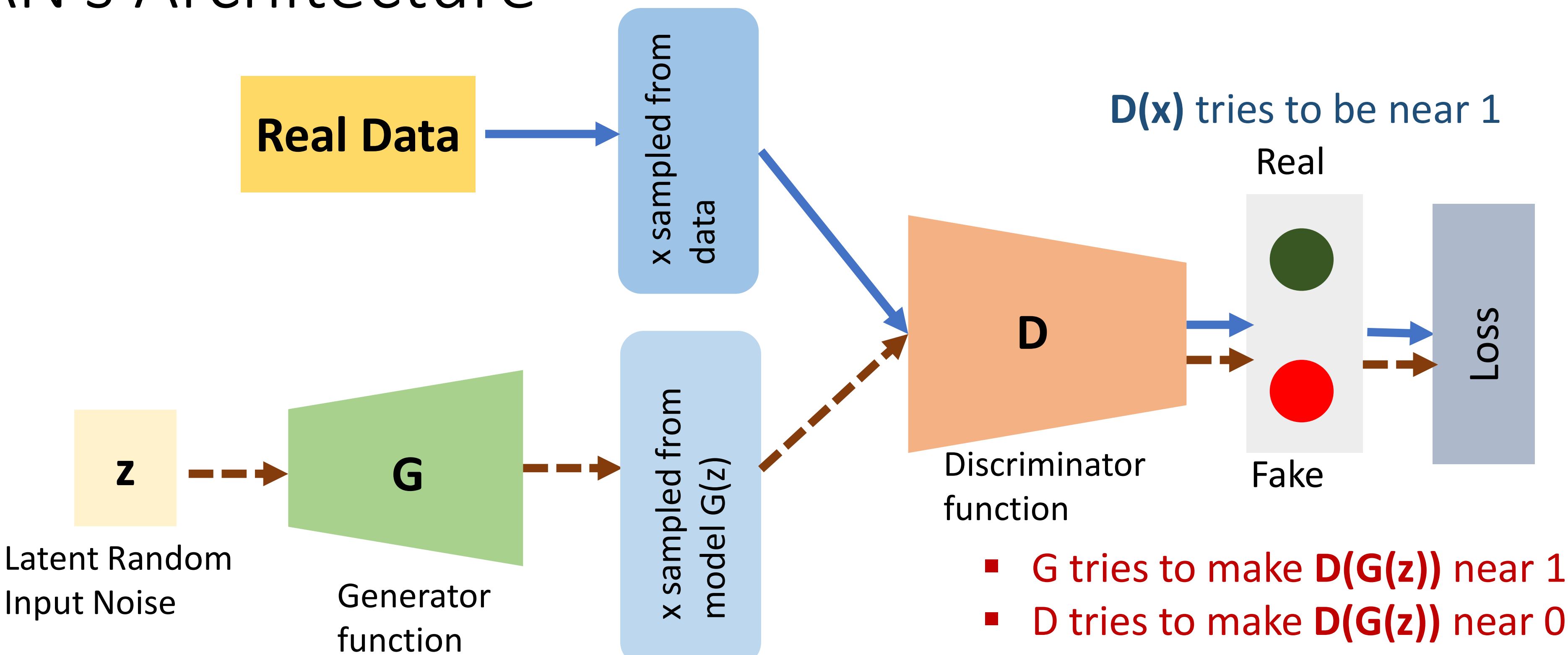
GANs

Quote from the original paper on GANs:

"The generative model can be thought of as analogous to a team of counterfeiters, trying to produce fake currency and use it without detection, while the discriminative model is analogous to the police, trying to detect the counterfeit currency. Competition in this game drives both teams to improve their methods until the counterfeits are indistinguishable from the genuine articles."

- Goodfellow et. al., "Generative Adversarial Networks" (2014)

GAN's Architecture



Note: Backprop error to update discriminator and generator weights

What is a latent variable?

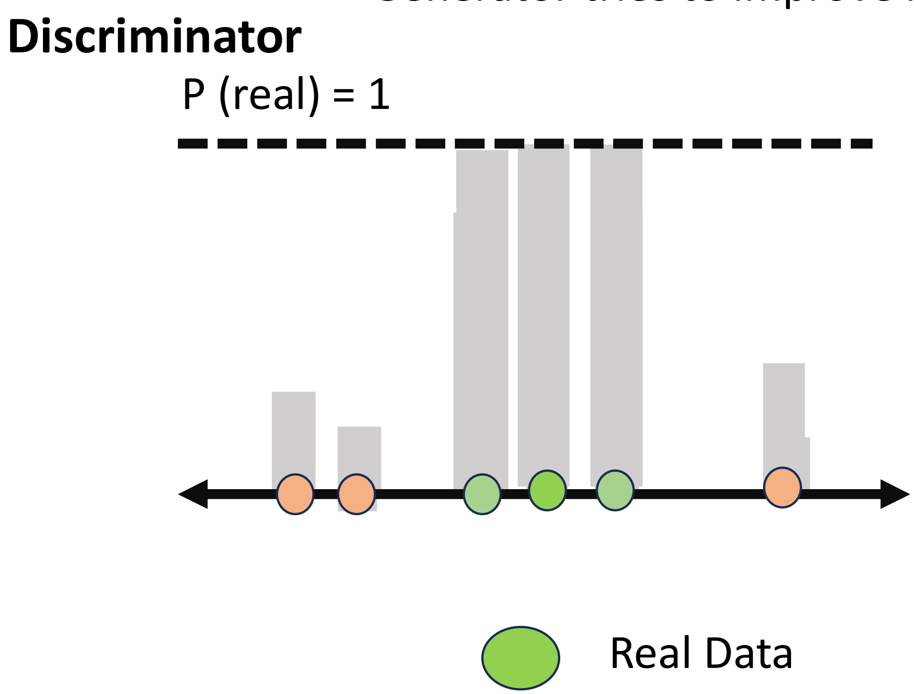
- It is a variable that is not directly observed or measured but is instead inferred from other observed variables.

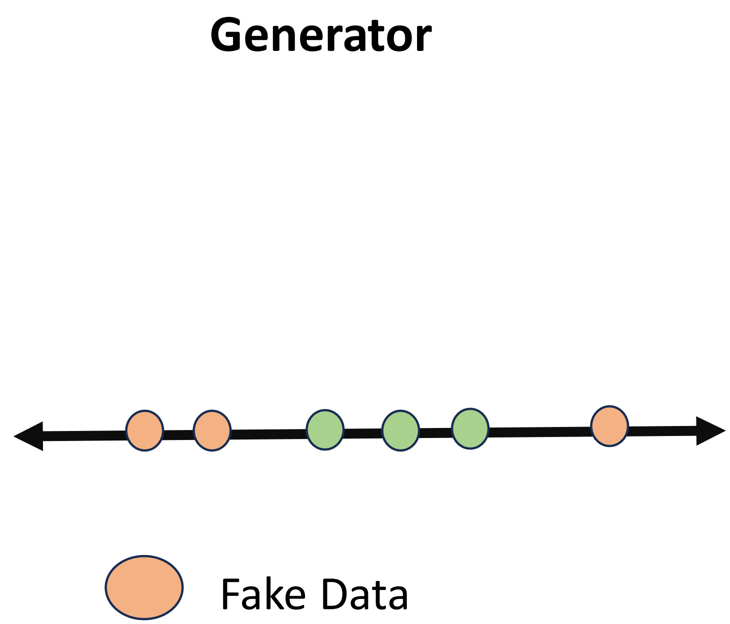
- Latent variables are a transformation of the data points into a continuous lower-dimensional space

Person	Liked	Age	Movie	Genre
X	Y	16	Spiderman	Action
Y	N	9	Hangover	Comedy
Y	Y	9	Clueless	Comedy
X	Y	16	Black Panther	Action
Y	Y	9	Terminal	Comedy
Z	Y	27	Annabelle	Horror
X	Y	16	Star wars	Action
Z	N	27	The Nun	Horror
Z	Y	27	Conjuring	Horror
Y	Y	9	Ted	Comedy

Intuition behind GANs

- Generator starts from noise to try to create an imitation of the data
- Discriminator looks at both real data and fake data created by the generator
- Generator tries to improve its imitation of the data



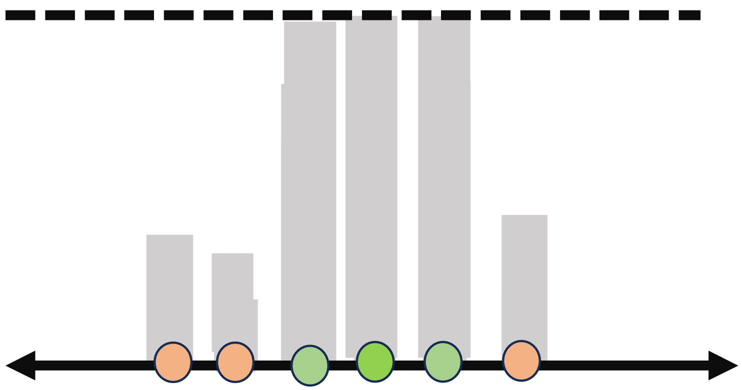


Intuition behind GANs

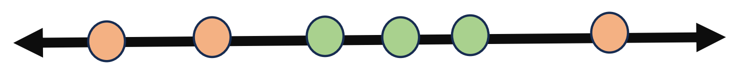
- Generator tries to improve its imitation of the data
- Discriminator tries to predict what's real and what's fake

Discriminator

P (real) = 1



Generator

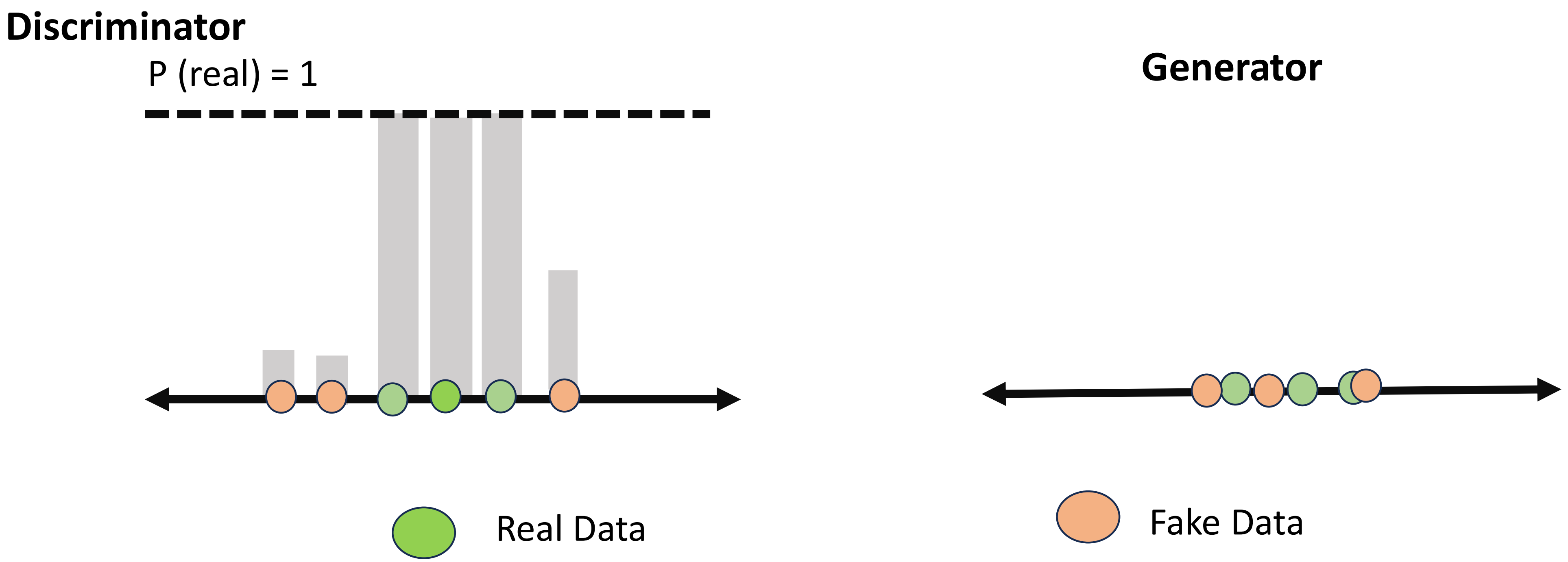


Real Data

Fake Data

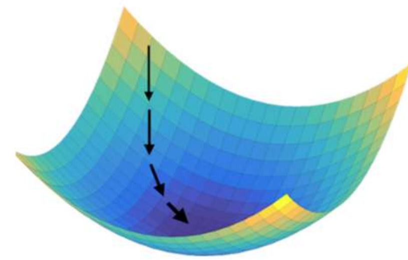
Intuition behind GANs

- Generator tries to create imitations of data to trick the discriminator
- Discriminator tries to identify real data from fakes created by the generator



Training GAN's: Discriminator's cost ($\boldsymbol{J^D}$)

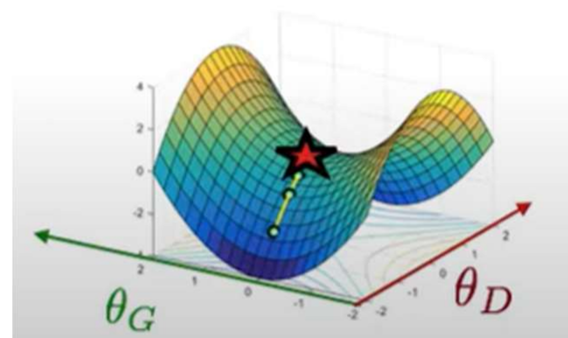
$$\min_{G}\max_{D} V(D,G) = \max_{G}\min_{D} -J^D$$



$$J^{(D)}\left(\theta^{(D)},\theta^{(G)}\right) = -\frac{1}{2}\mathbb{E}_{x\sim p_{data}}\underbrace{\left[\log D(x)\right]}_{\text{Discriminator output for real data x}} - \frac{1}{2}\mathbb{E}_{z\sim q(z)}\underbrace{\left[log(1-D(G(z)))\right]}_{\text{Discriminator output for fake data G(z)}}$$

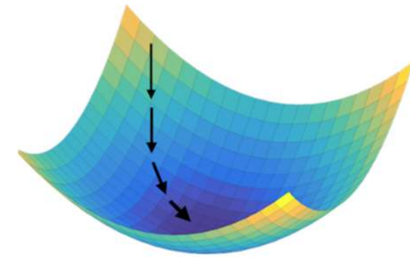
$$J^{G}\left(\theta^{(D)},\theta^{(G)}\right) = -J^D$$

- Generator minimizes the probability of the discriminator being correct
- Saddle Point of discriminators loss



Training GAN's: Discriminator's cost $(\boldsymbol{J^D})$

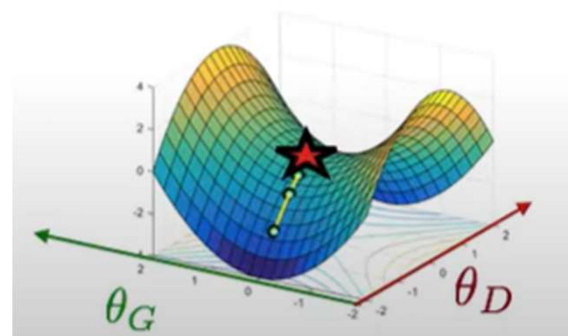
$$\min_{G} \max_{D} \boldsymbol{V}(\boldsymbol{D}, \boldsymbol{G}) = \max_{G} \min_{D} -\boldsymbol{J^D}$$



OPEN QUESTION:

- Is the equilibrium "locally (exponentially) stable"?
- When it is not, how do we make stable?

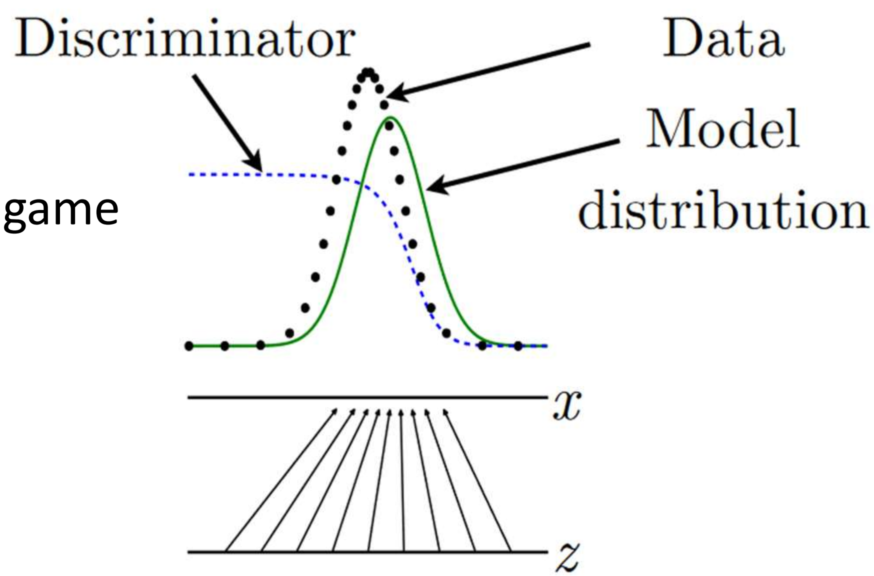
- Generator minimizes the probability of the discriminator being correct
- Saddle Point of discriminators loss



Min-max Game Approach

$$J^{(D)}\left(\theta^{(D)},\theta^{(G)}\right) = -\frac{1}{2}\mathbb{E}_{x\sim p_{data}}\left[\log D(x)\right] - \frac{1}{2}\mathbb{E}_{z\sim q(z)}\left[log(1-D(G(z)))\right]$$

$$\frac{\partial J^{D}}{\partial\, D(x)} = 0 \;\; \rightarrow \;\; D^{*}(x) = \frac{P_{data}(x)}{P_{data}(x) + P_{model}(x)}$$



- The Nash Equilibrium/Saddle point of this particular game is achieved at:
  - $P_{data}(x) = P_{model}(x)\ \forall\ x$
  - $D(x) = \frac{1}{2}\ \forall\ x$

❑ **What happens with D(G(z)) -> 0?**

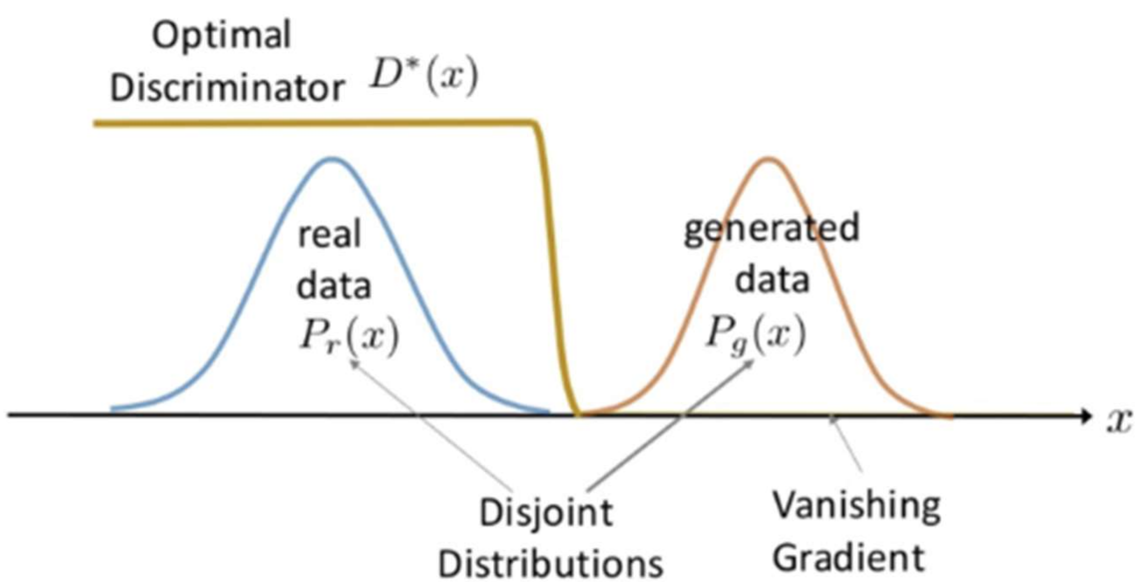
Vanishing Gradient Problem with Generator

$$J^{(D)}(\theta^{(D)}, \theta^{(G)}) = -\frac{1}{2}\mathbb{E}_{x\sim p_{data}}[\log D(x)] - \frac{1}{2}\mathbb{E}_{z\sim q(z)}[log(1 - D(G(z)))]$$

- Gradient goes to 0 if D is confident, i.e., D(G(z)) -> 0

✓ As can be seen that whenever the discriminator becomes very confident, the loss value will be zero

✓ Nothing to improve for the generator



Heuristic, non-saturating game

$$J^{(D)} = -\frac{1}{2}\mathbb{E}_{x\sim p_{data}}\left[\log D(x)\right] - \frac{1}{2}\mathbb{E}_{z\sim q(z)}\left[log(1 - D(G(z)))\right]$$

$$J^{G} = -\frac{1}{2}E_{z\sim q(z)}\log D(G(z))$$

- Generator maximizes the log probability of the discriminator’s mistake

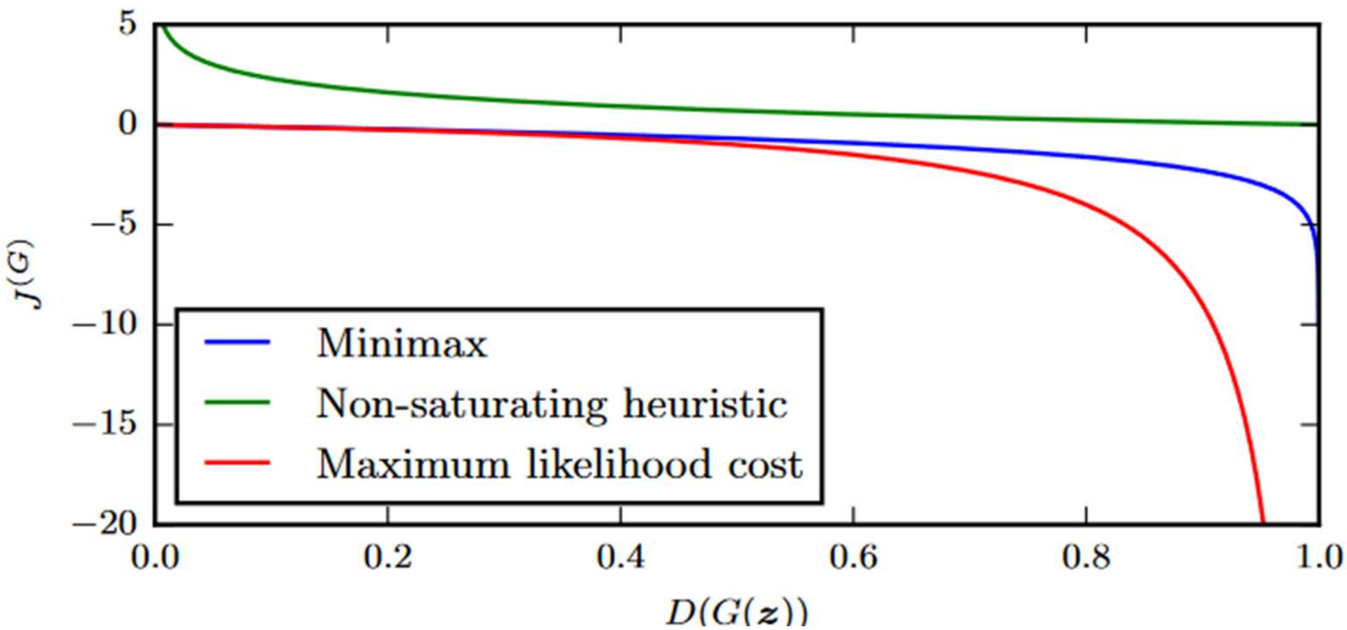
Comparison of Generator Losses

The cost that the generator receives for generating a samples G(z) depends only on how the discriminator responds to that sample

$$J^G = -J^D$$

$$J^G = -\frac{1}{2} E_z \log D(G(z))$$

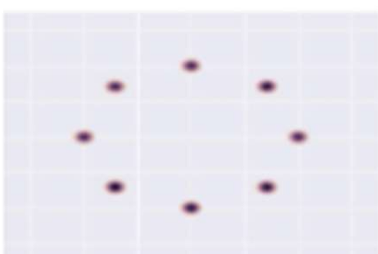
$$J^G = -\frac{1}{2} E_z \, e^{(\sigma^{-1} D(G(z)))}$$



Why GANs are hard to train?

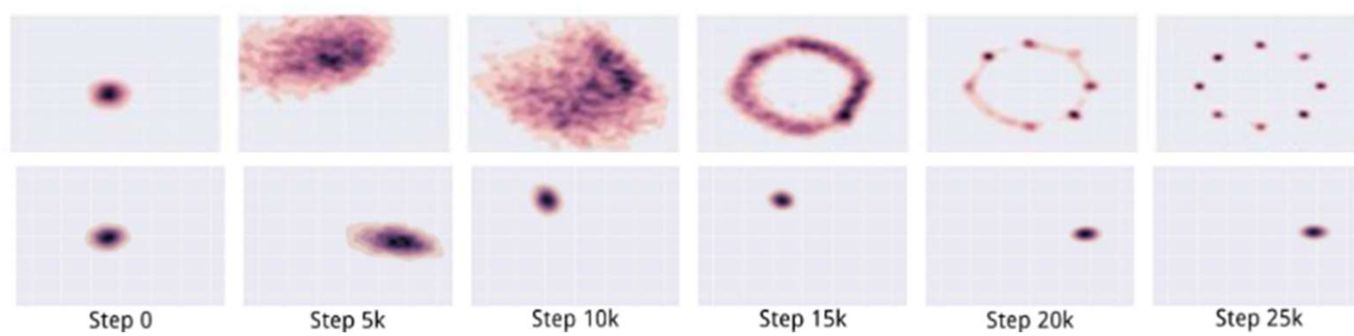
- Non-Convergence: D & G nullifies each other's learning in every iteration. The two learning tasks need to have balance to achieve stability

- Mode Collapse: Maintain trade-off of generating more accurate vs high coverage samples. Generator excels in a subspace but does-not cover entire real distribution



Target

Expected



Output

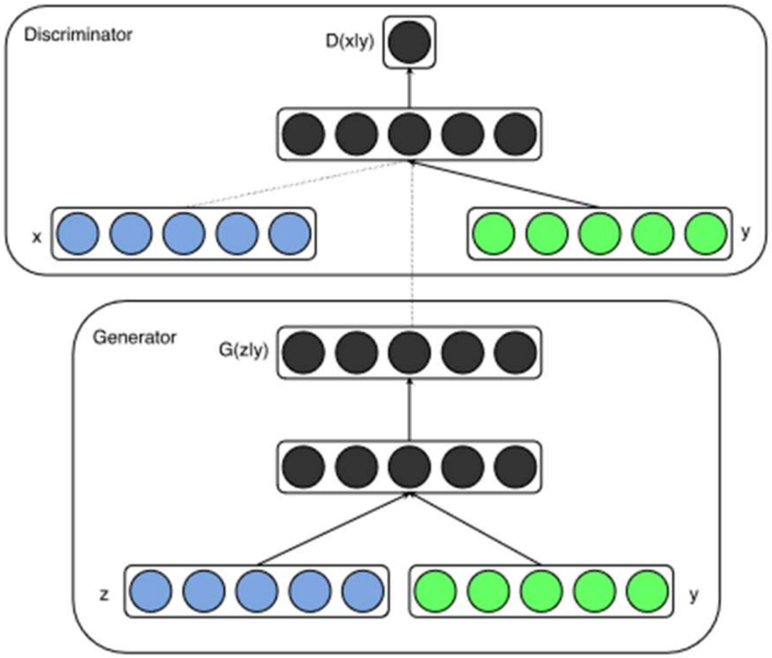
Tricks to train GANs

- Historical generated batches: Helps stabilize discriminator training at early stages. [Shrivastava, Ashish, et al. "Learning from Simulated and Unsupervised Images through Adversarial Training." CVPR. Vol. 2. No. 4. 2017]

- One Side Label Smoothening: Involves modifying the labels used during the training of the discriminator. [Salimans, Tim, et al. "Improved techniques for training gans." Advances in Neural Information Processing Systems. 2016]

- Feature Matching: Generator is trained such that the expected value of statistics matches the expected value of real statistics

Few variations of GAN – Conditional GANs

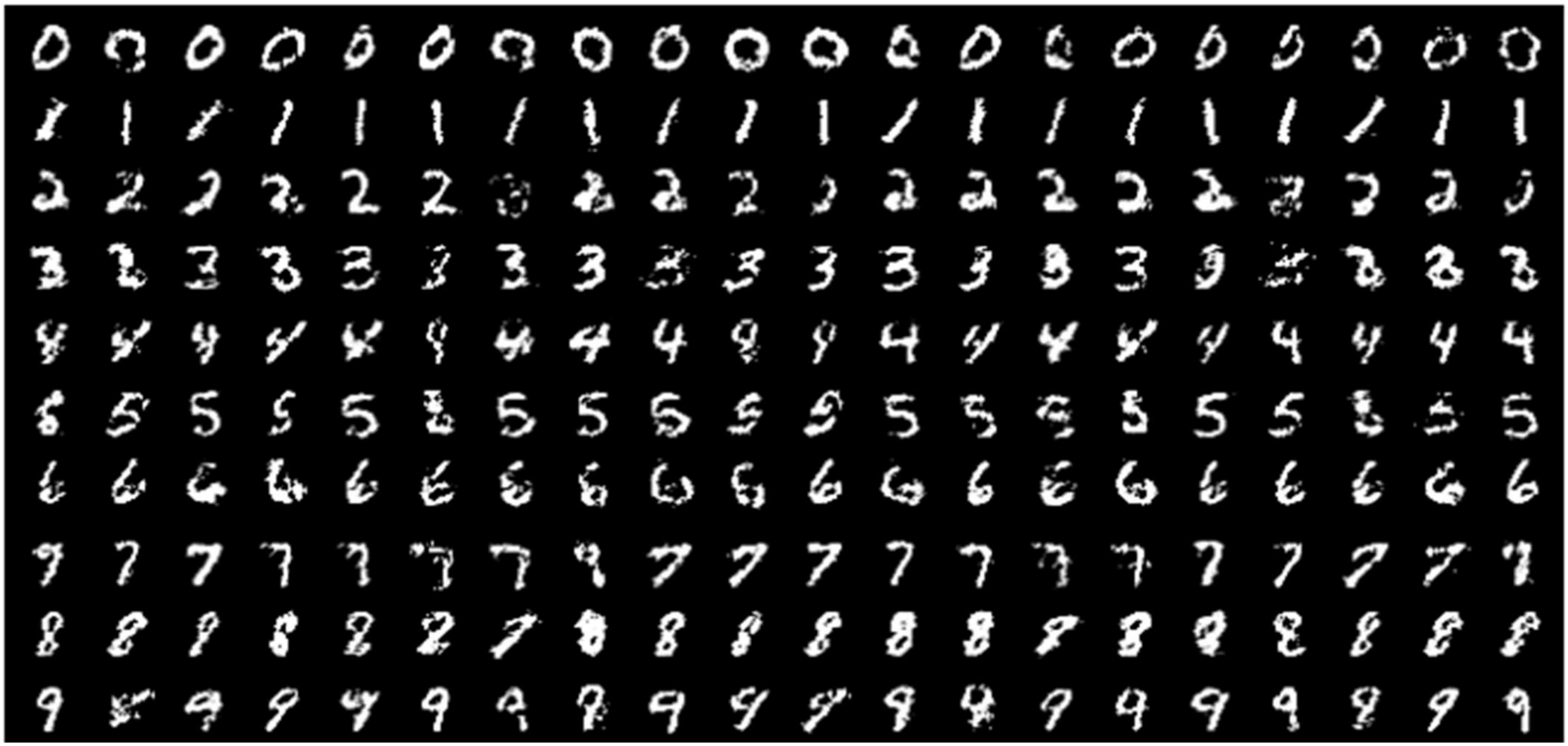


- Generator Learns P(x|z,y)
- Discriminator Learns P(x,y)

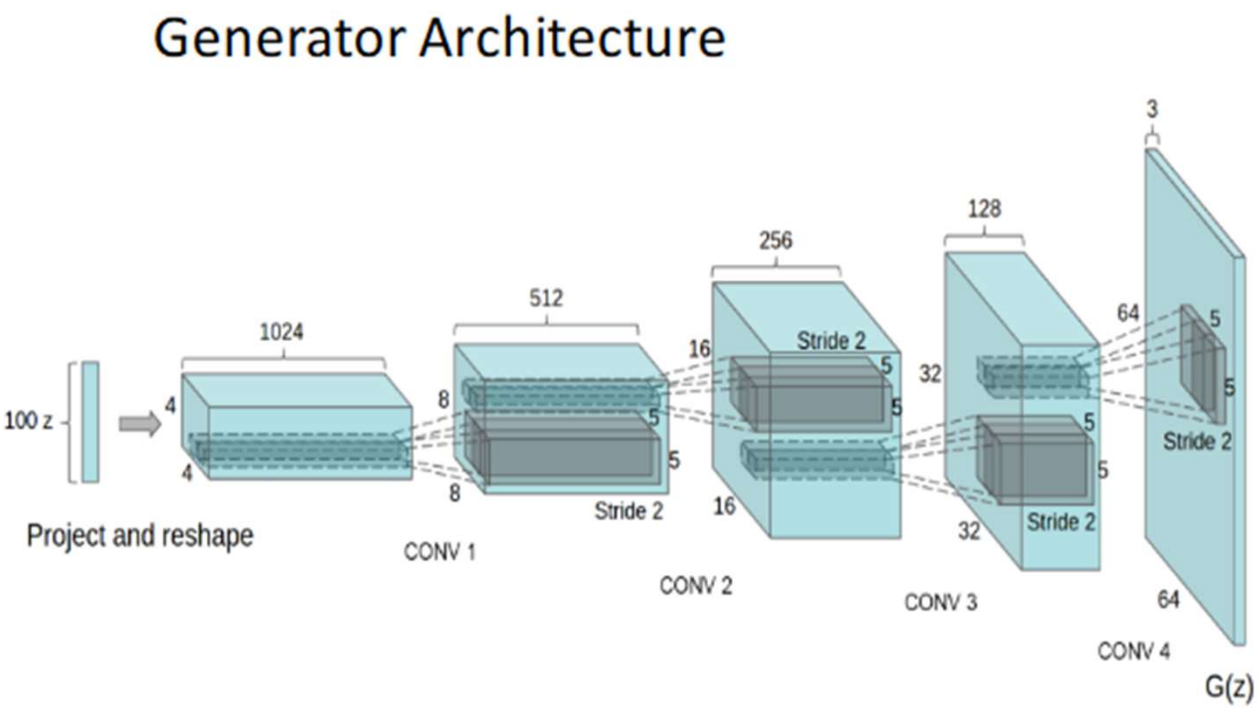
$$J^D = \mathbb{E}_{x \sim p_{data}(x)}[\log D(x|y)] + \mathbb{E}_{z \sim p_x(z)}[\log(1 - D(G(z|y)))]$$

Mirza, M. and Osindero, S., 2014. Conditional generative adversarial nets. arXiv:1411.1784.

Each row is conditioned on a different label. You can use a single neural network to generate all 10 digits by telling it what digit to generate



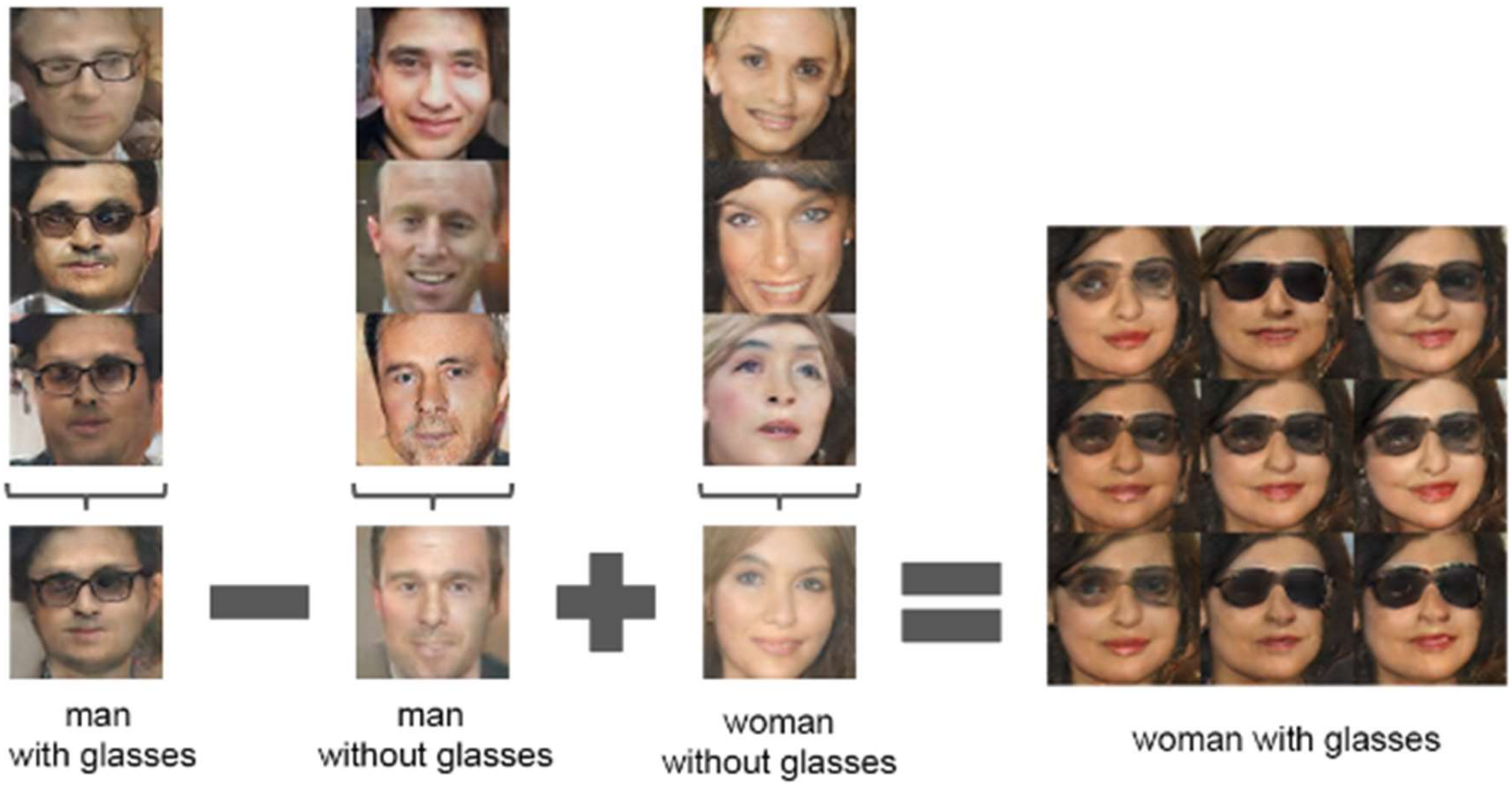
Deep Convolutional GANs (DCGANs)



- Replace FC hidden layers with Convolutions
  - Generator: Fractional-Strided Convolutions
- Use Batch Normalization after each layer
- Inside Generator
  - Use ReLU for hidden layers
  - Use Tanh for the output layer

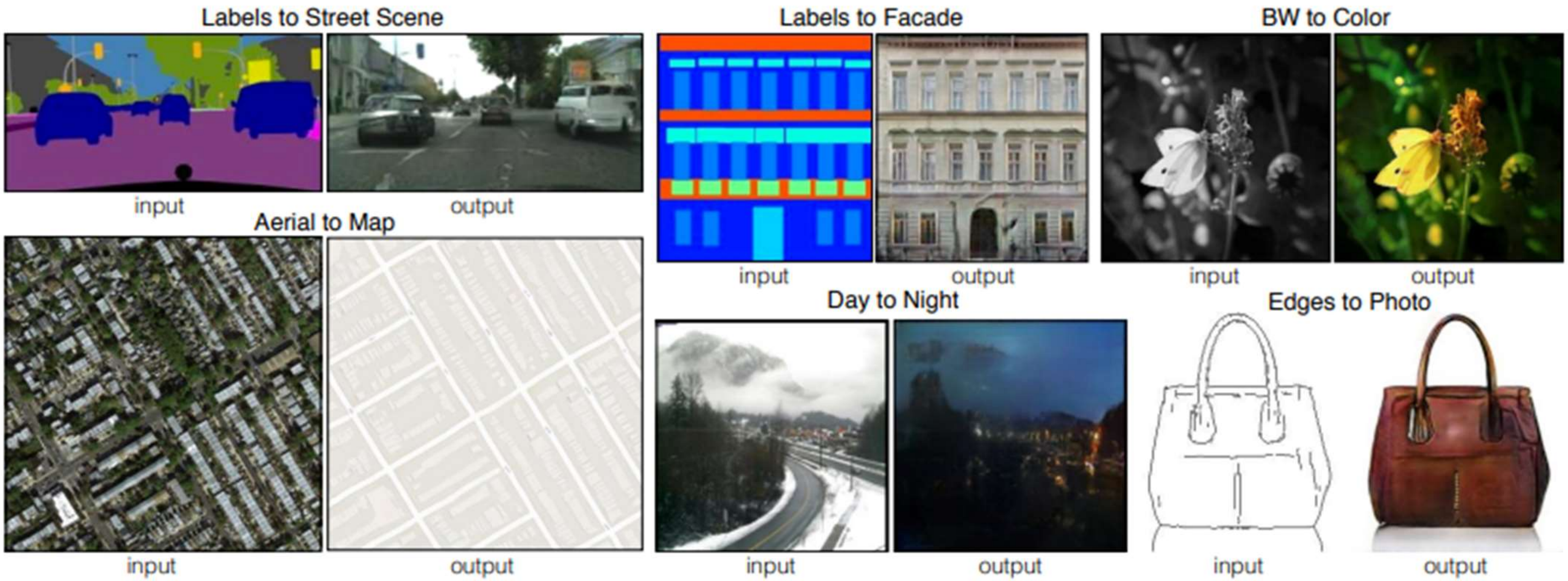
Radford, Alec, Luke Metz, and Soumith Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks." arXiv:1511.06434 (2015).

Latent vectors capture interesting patterns….



vector additions and subtractions are meaningful in this latent space

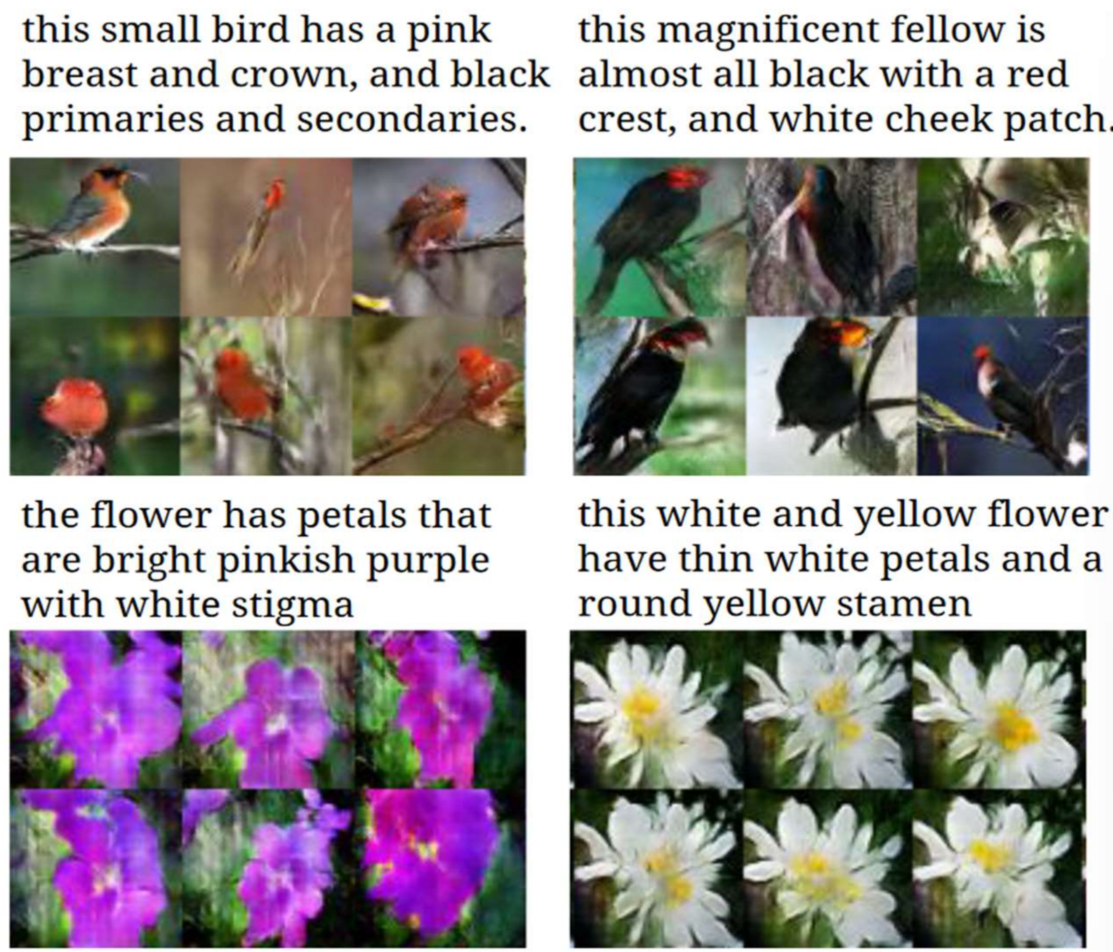
Image-to-Image Translation



[Link to an interactive demo of the 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

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

Summary

- GANs are generative models that are implemented using two stochastic neural network modules: Generator and Discriminator
- The Generator tries to generate samples from random noise as input
- The Discriminator tries to distinguish the samples from Generator and samples from the real data distribution
- Both the networks are trained adversarially to fool the other component making them better at their respective tasks
- GAN is an active area of research with a lot of work currently done in the theoretic foundation of the Network

Important Papers

- NIPS 2016 Tutorial: - Ian Goodfellow, https://arxiv.org/search/cs?searchtype=author&query=Goodfellow,+I
- Arjovsky, Martin, and Léon Bottou. "Towards principled methods for training generative adversarial networks." arXiv preprint arXiv:1701.04862 (2017).
- Roth, Kevin, et al. "Stabilizing training of generative adversarial networks through regularization." Advances in Neural Information Processing Systems. 2017.
- Li, Jerry, et al. "Towards understanding the dynamics of generative adversarial networks.“ arXiv:1706.09884 (2017).
- Kodali, Naveen, et al. "On convergence and stability of GANs." arXiv:1705.07215 (2017).
- Fedus, William, et al. "Many Paths to Equilibrium: GANs Do Not Need to Decrease a Divergence At Every Step. arXiv:1710.08446 (2017).
- https://github.com/soumith/ganhacks#authors
- http://www.inference.vc/instance-noise-a-trick-for-stabilising-gan-training/
- https://www.araya.org/archives/1183

Codes, Tools and Tricks

- https://github.com/soumith/ganhacks#authors
- https://medium.com/@utk.is.here/keep-calm-and-train-a-gan-pitfalls-and-tips-on-training-generative-adversarial-networks-edd529764aa9
- https://jhui.github.io/2017/03/05/Generative-adversarial-models/
- https://www.kaggle.com/code/theblackmamba31/generating-fake-faces-using-gan
- https://realpython.com/generative-adversarial-networks/