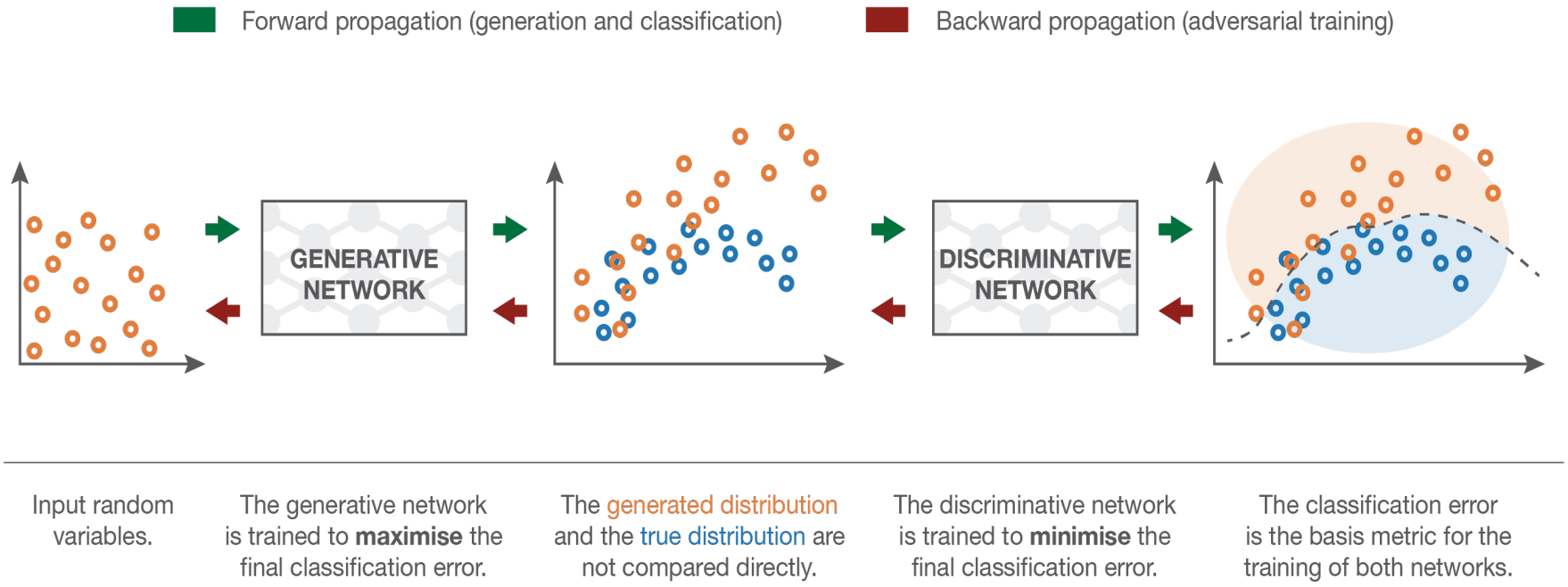
Lecture 13:GANS Part 2

CPSC/AMTH 452/552

CBB 663

GAN

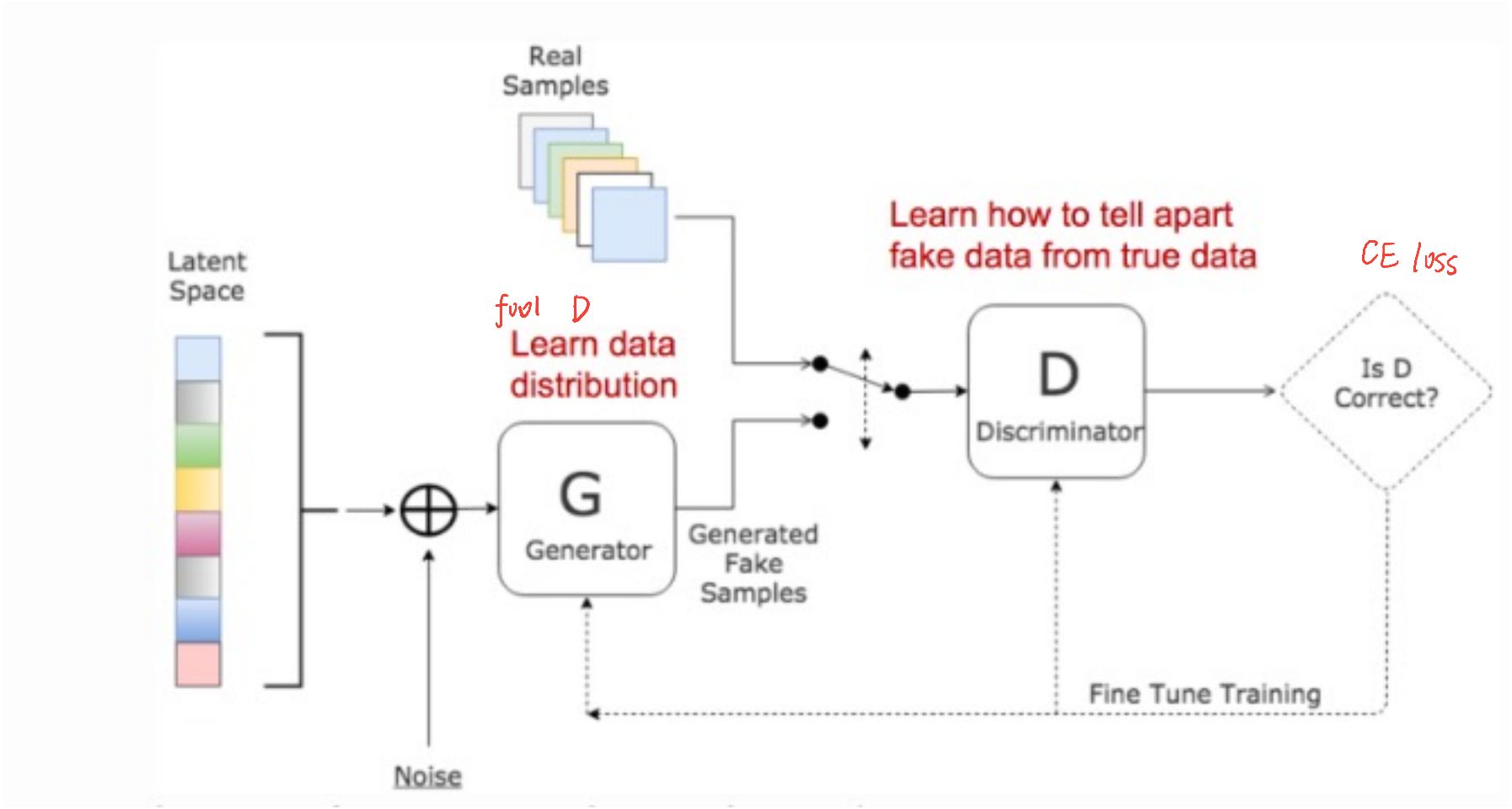


Discriminator classifies between real and fake examples

Generated Examples

Generative Adversarial Network

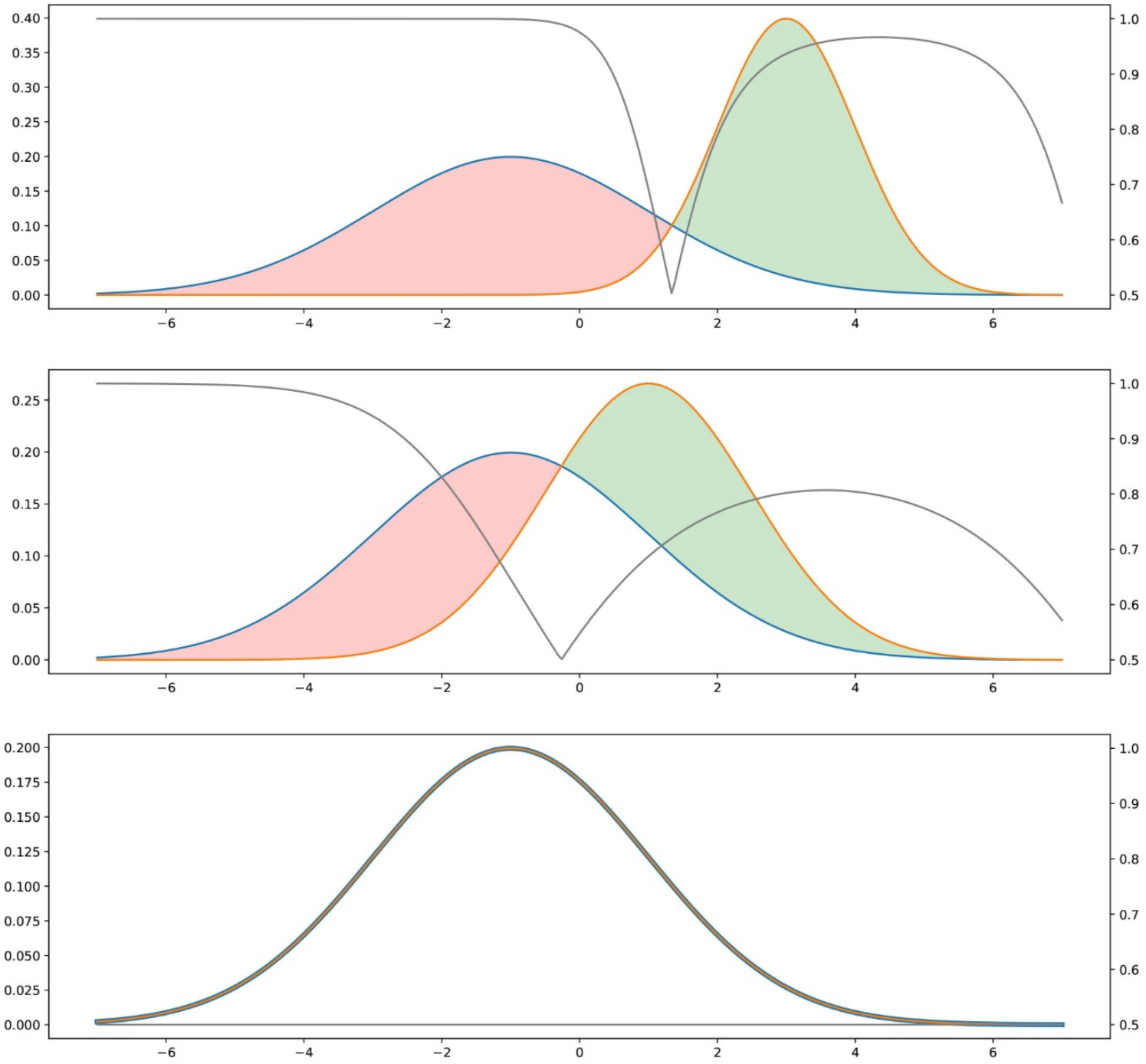
generate data from noise



Goodfellow et al. 2014

2 Networks

- **Generator**: takes random noise and transforms into sample from target distribution
- **Discriminator:** discriminates between generated (“fake”) examples and real examples
- The two networks play a minimax game
- Indirect way of matching distributions


0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
1.0
0.9
0.8
0.7
0.6
0.5
−6
−4
−2
0
2
4
6
0.25
0.20
0.15
0.10
0.05
0.00
1.0
0.9
0.8
0.7
0.6
0.5
−6
−4
−2
0
2
4
6
0.200
0.175
0.150
0.125
0.100
0.075
0.050
0.025
0.000
1.0
0.9
0.8
0.7
0.6
0.5
−6
−4
−2
0
2
4
6

GAN Loss function

p_z	Data distribution over noise input z
p_g	The generator's distribution over data x
p_r	Data distribution over real sample x

Maximize Discriminator's answer. (1) on real samples

$$\mathbb{E}_{x \sim p_r(x)}[\log D(x)]$$

Maximize Discriminator's answer (0) for fake samples

$$\mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))].$$

Minimax game: Generator wants discriminator to perform badly

$$\begin{aligned}\min_G \max_D L(D, G) &= \mathbb{E}_{x \sim p_r(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))] \\ &= \mathbb{E}_{x \sim p_r(x)}[\log D(x)] + \mathbb{E}_{x \sim p_g(x)}[\log(1 - D(x)]\end{aligned}$$

Optimal Discriminator Behavior

$$L(G,D) = \int_x \Big(p_r(x)\log(D(x)) + p_g(x)\log(1-D(x)) \Big) dx$$

$$\tilde{x} = D(x), A = p_r(x), B = p_g(x)$$

$$\begin{aligned} f(\tilde{x}) &= A log\tilde{x} + Blog(1-\tilde{x}) \\ \frac{df(\tilde{x})}{d\tilde{x}} &= A\frac{1}{ln10}\frac{1}{\tilde{x}} - B\frac{1}{ln10}\frac{1}{1-\tilde{x}} \\ &= \frac{1}{ln10}(\frac{A}{\tilde{x}} - \frac{B}{1-\tilde{x}}) \\ &= \frac{1}{ln10}\frac{A-(A+B)\tilde{x}}{\tilde{x}(1-\tilde{x})} \end{aligned}$$

Thus, set $\frac{df(\tilde{x})}{d\tilde{x}} = 0$, we get the best value of the discriminator:

$$D^*(x) = \tilde{x}^* = \frac{A}{A+B} = \frac{p_r(x)}{p_r(x)+p_g(x)} \in [0,1].$$

Discriminator computes JS Divergence

- When generator is optimal $p_g = p_r, D^*(x)$
- When both G and D are optimal D loss is -2 log 2 (otherwise known as ½)
- According to formula for JS Divergence

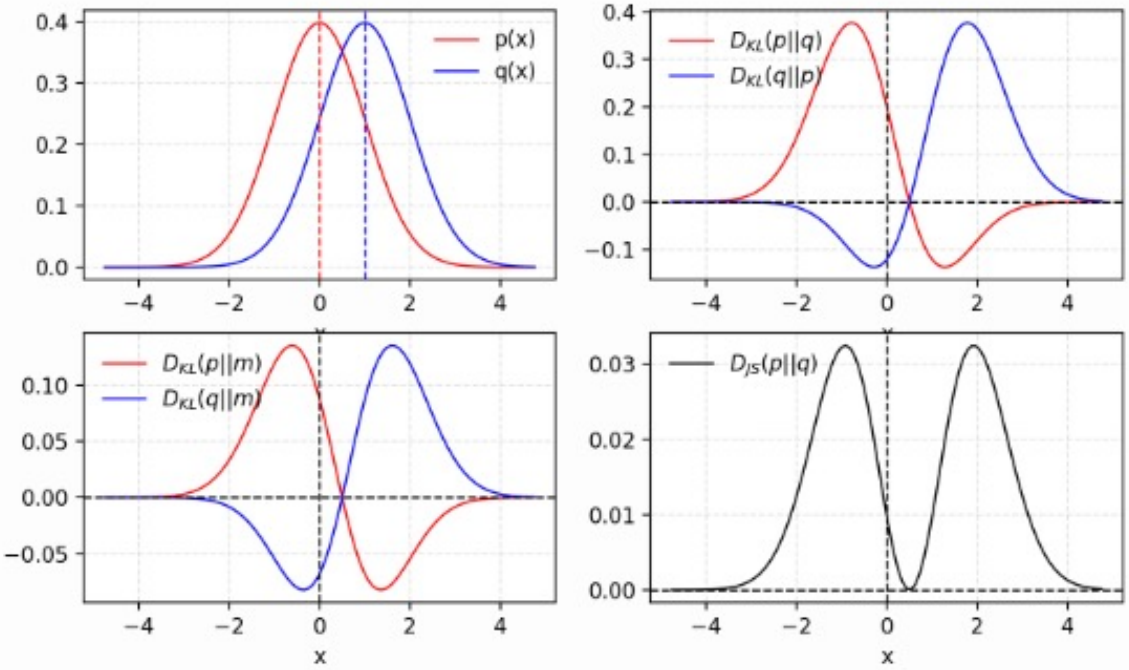
$$
\begin{aligned}
D_{JS}(p_r \| p_g) &= \frac{1}{2} D_{KL}(p_r || \frac{p_r + p_g}{2}) + \frac{1}{2} D_{KL}(p_g || \frac{p_r + p_g}{2}) \\
&= \frac{1}{2}\left(\log 2 + \int_x p_r(x) \log \frac{p_r(x)}{p_r + p_g(x)} dx \right) + \\
&\quad \frac{1}{2}\left(\log 2 + \int_x p_g(x) \log \frac{p_g(x)}{p_r + p_g(x)} dx \right) \\
&= \frac{1}{2}\left(\log 4 + L(G, D^*) \right)
\end{aligned}
$$

- Thus the discriminator converges to a JS divergence between the samples

Jensen-Shannon Divergence

KL Divergence $$D_{\mathrm{KL}}(P \parallel Q) = -\sum_{x \in \mathcal{X}} P(x) \log\left(\frac{Q(x)}{P(x)}\right)$$

JS divergence $$D_{JS}(p\|q) = \frac{1}{2} D_{KL}(p\|\frac{p+q}{2}) + \frac{1}{2} D_{KL}(q\|\frac{p+q}{2})$$



GANs

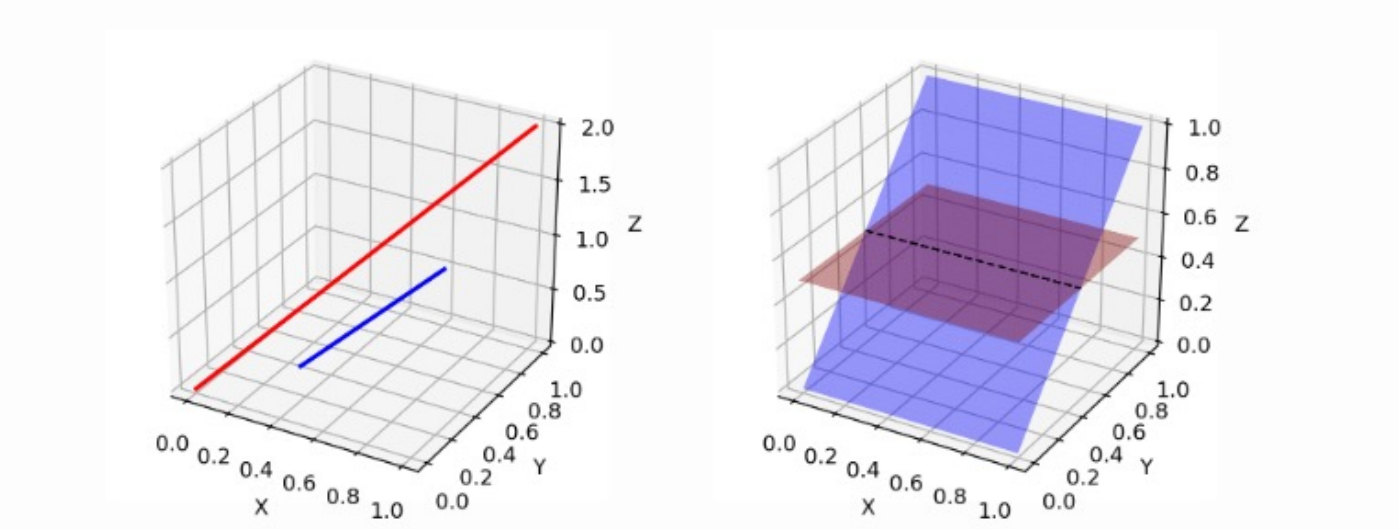
- The JS describes the *equilibrium* point
- But that doesn't mean we will get there
- Why?

$$\nabla_{\theta_G}\mathbb{E}_{z\sim p(z)}\left[\log(1-D_{\theta_D}(G_{\theta_G}(z)))\right]=\mathbb{E}_{z\sim p(z)}\left[\frac{1}{1-\sigma_{\theta_D}(s)}(-\sigma'_{\theta_D}(s))\Big|_{s=G_{\theta_G}(z)}\nabla_{\theta_G}G_{\theta_G}(z)\right]$$

From the derivative of $\log(u)$
With $u = 1 - sigmoid(D(G(z)))$

Problems with GANs 1: Small gradients when generator is bad

- When generated data is very far from training data the two distributions are far away discriminator can be very good
- Small gradients, generator does not know how to change itself to learn well



Vanishing Gradients

$$\nabla_{\theta_G}\mathbb{E}_{z\sim p(z)}\left[\log(1-D_{\theta_D}(G_{\theta_G}(z)))\right] = \mathbb{E}_{z\sim p(z)}\left[\frac{1}{1-\sigma_{\theta_D}(s)}(-\sigma'_{\theta_D}(s))\Big|_{s=G_{\theta_G}(z)} \nabla_{\theta_G}G_{\theta_G}(z)\right]$$

From the derivative of $\log(u)$
With $u = 1 - sigmoid(D(G(z)))$

- What happens when D is very confident a point is fake?
  - $\sigma = sigmoid(D(G(z)))$ is small
  - σ' is small
  - This term goes to zero ($\frac{1}{1-0} \cdot 0$)
- Vanishing gradients!
- Generator cannot improve

Alternative minimization

- This observation motivated an alternate objective for G to minimize:

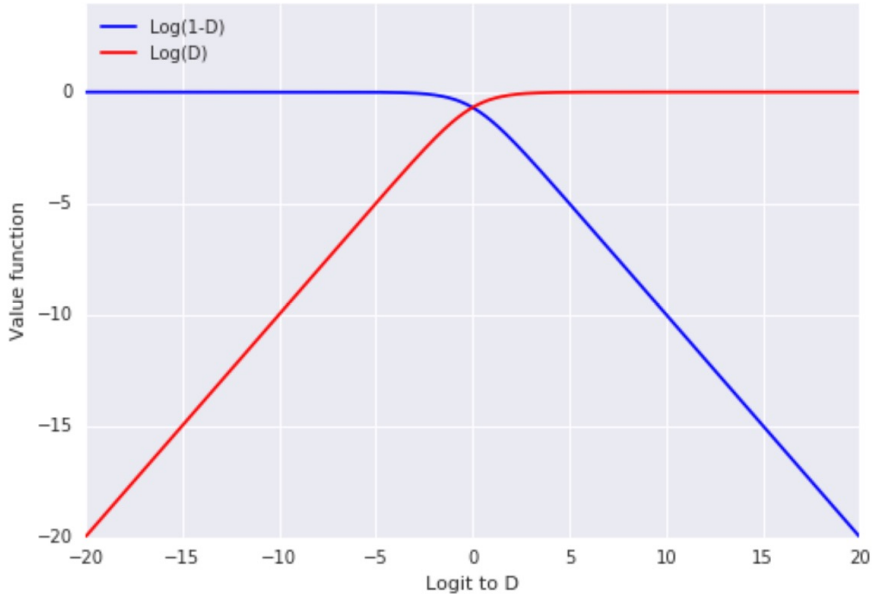
$$\textbf{Minimax} \quad J^{(G)}(G) = \mathop{\mathbb{E}}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}} \log[1 - D(G(\boldsymbol{z}))].$$

$$\textbf{Non-saturating} \quad J^{(G)}(G) = -\mathop{\mathbb{E}}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}} \log D(G(\boldsymbol{z})).$$

- Instead of the first term in gradient being $\frac{1}{1-\sigma} \cdot -\sigma'$ it is now $\frac{1}{\sigma} \cdot \sigma'$
  - $\frac{1}{\sigma}$ and σ' diverge as they approach 0, leading to non-vanishing gradients

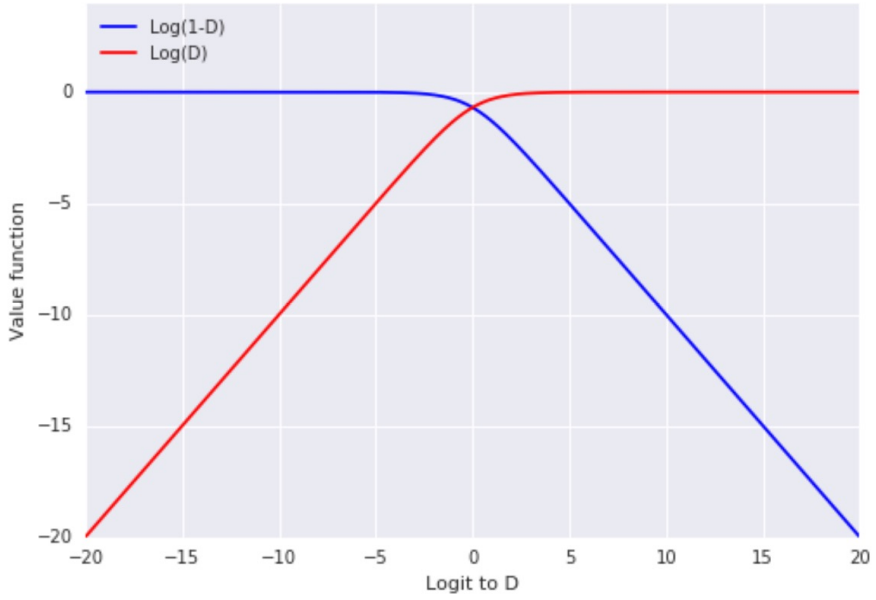
GANs

- Minimax (blue line) has extremely small gradient when D is confident the point is fake (left half, $\sigma \approx 0$)
  - This is bad because that is precisely when G need gradient signal most
- Non-saturating (red line) has extremely small gradient when D is confident the point is real (right half, $\sigma \approx 1$)
  - Not bad, because this isn't the term D is optimizing for fake points



GANs

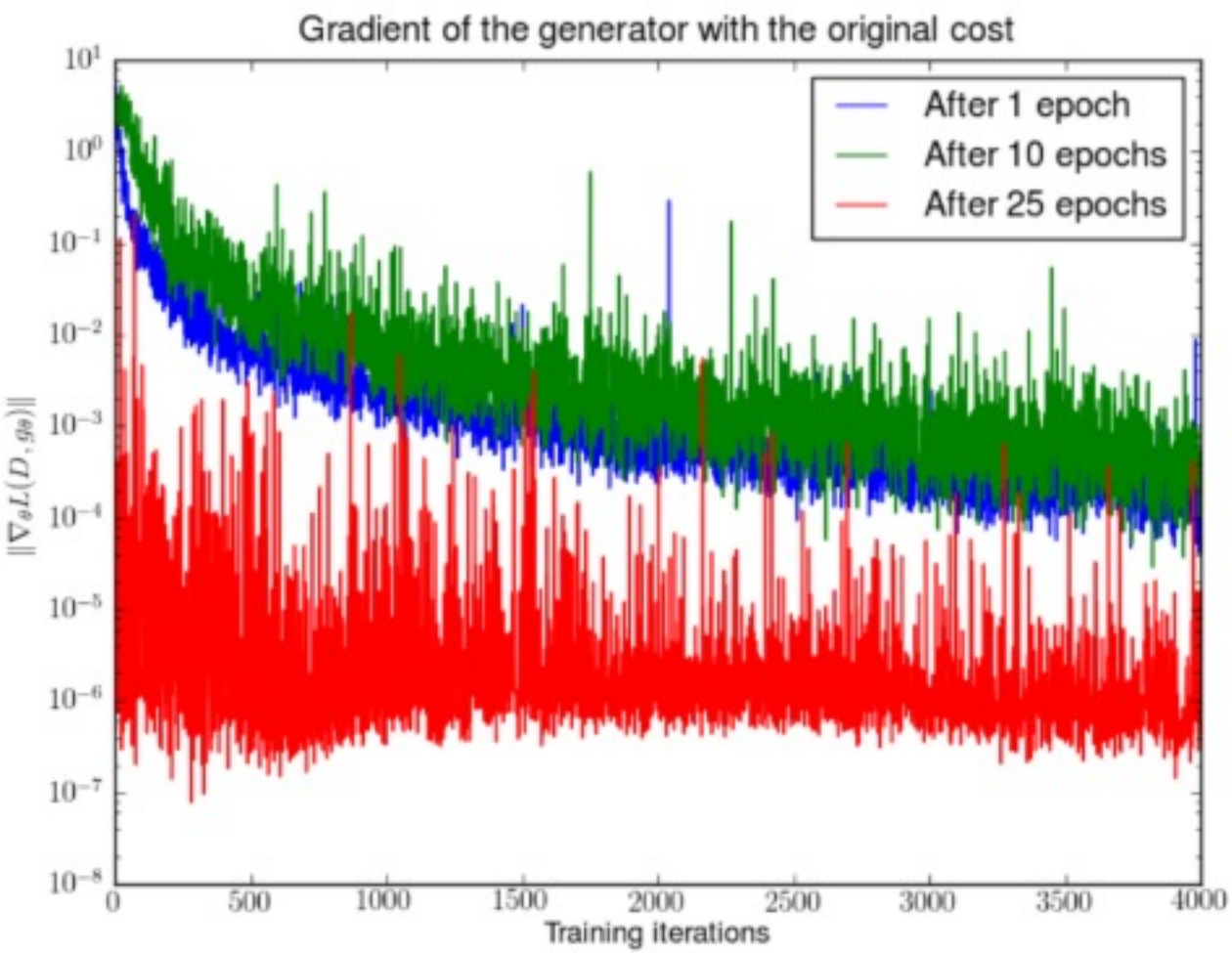
- G receives largest signal when D thinks points are fake
- D receives largest signal when it thinks fake points are real



GANs

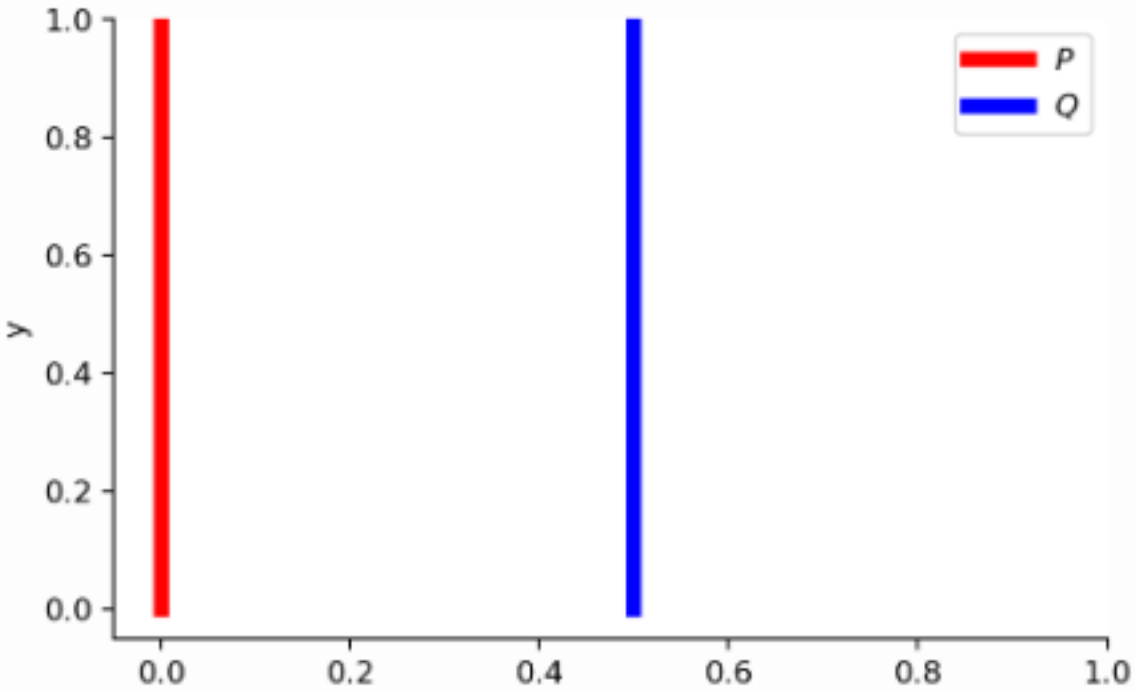
- GANs can generate sharper images generally
  - Do not need to explicitly define distribution distances
- Unfortunately, learning GANs with neural networks can be difficult in practice
  - Convergence to equilibrium is not guaranteed
  - Equilibria for minimax game are saddle points

Problem 2: Vanishing gradient for generator when discriminator gets good



Wasserstein GAN [Arjovsky et al. 2017]

Uses Wasserstein metric instead of JS



Distance between manifolds with disjoint support

Distances on parallel lines

KL Divergence:

$$D_{KL}(P\|Q) = \sum_{x=0, y\sim U(0,1)} 1 \cdot \log \frac{1}{0} = +\infty$$

$$D_{KL}(Q\|P) = \sum_{x=\theta, y\sim U(0,1)} 1 \cdot \log \frac{1}{0} = +\infty$$

JS Divergence:

$$D_{JS}(P, Q) = \frac{1}{2}\left(\sum_{x=0, y\sim U(0,1)} 1 \cdot \log \frac{1}{1/2} + \sum_{x=0, y\sim U(0,1)} 1 \cdot \log \frac{1}{1/2} \right) = \log 2$$

Same value independent of distance between lines!

Wasserstein on Parallel Lines

$$W(P, Q) = |\theta|$$

Matches with intuition of moving dirt.

Wasserstein distance can be better when the supports of the two probability distributions are far away.

This can occur when generated data is far from training data

Wasserstein Distance by Discrimnator

- Instead of the Discriminator giving a decision “0” for fake data but “1” for real data gives a value of a function f
- The witness function f can be estimated by the Discriminator
- The Discriminator would output a low value for fake samples and high value for real data

Kantorovich-Rubenstein duality:

$$W(p_r, p_g) = \frac{1}{K} \sup_{\|f\|_L \leq K} \mathbb{E}_{x \sim p_r}[f(x)] - \mathbb{E}_{x \sim p_g}[f(x)]$$

- This function has to be lipschitz continous (can’t be jagged) has to vary smoothly

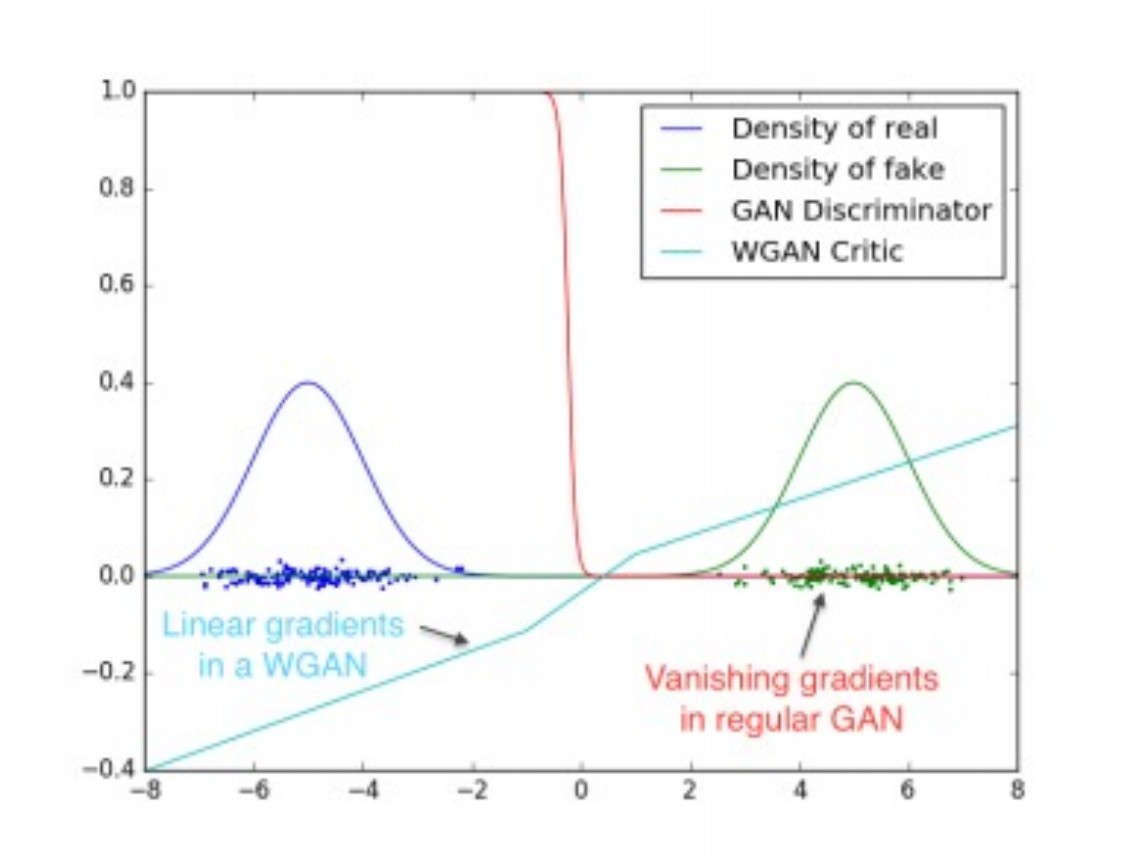
$$d_Y(f(x_1), f(x_2)) \leq K d_X(x_1, x_2)$$

WGAN In practice

- Takes off sigmoidal decision layer
- After every gradient update to the witness function f, clamp weights to a small fixed range
- New loss function does not have logarithm, just difference of expected values

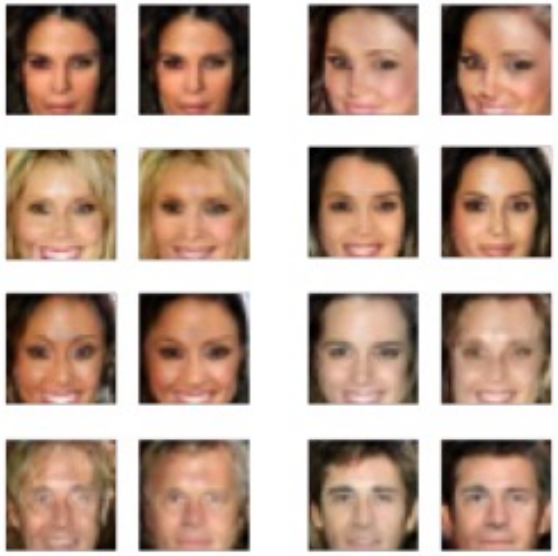
$$L(p_r, p_g) = W(p_r, p_g) = \max_{w \in W} \mathbb{E}_{x \sim p_r}[f_w(x)] - \mathbb{E}_{z \sim p_r(z)}[f_w(g_\theta(z))]$$

Improves vanishing Gradient

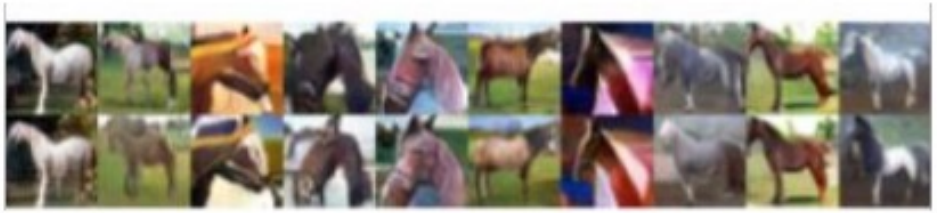


Problem 3: Mode Collapse

The generator can fool the discriminator by memorizing a small number of images.



evidence of lack of diversity



Improvements in Training GANs

- Feature matching
  - Optimize the discriminator to inspect whether the generator's output matches the statistics of real samples, i.e., does the generated mean equal the real mean?

$$\|\mathbb{E}_{x\sim p_r} f(x) - \mathbb{E}_{z\sim p_z(z)} f(G(z))\|_2^2$$

- Minibatch discrimination
  - Take the batch as a whole instead of one input at a time and learn relationships between pairs of samples $c(x_i, x_j)$,

$$o(x_i) = \sum_j c(x_i, x_j).$$

GANs

- Can learn well with carefully selected model architectures
  - Radford et al. (2015): deep convolutional GAN (DCGAN)
  - Generated images of bedrooms
- Can break the generation process into many levels
  - Can learn to sample from conditional distributions
  - Train a series of conditional GANs to first generate low-res versions and then incrementally add details
  - Denton et al. (2015): LAPGAN
  - Generated images of churches

Are these faces real or fake?

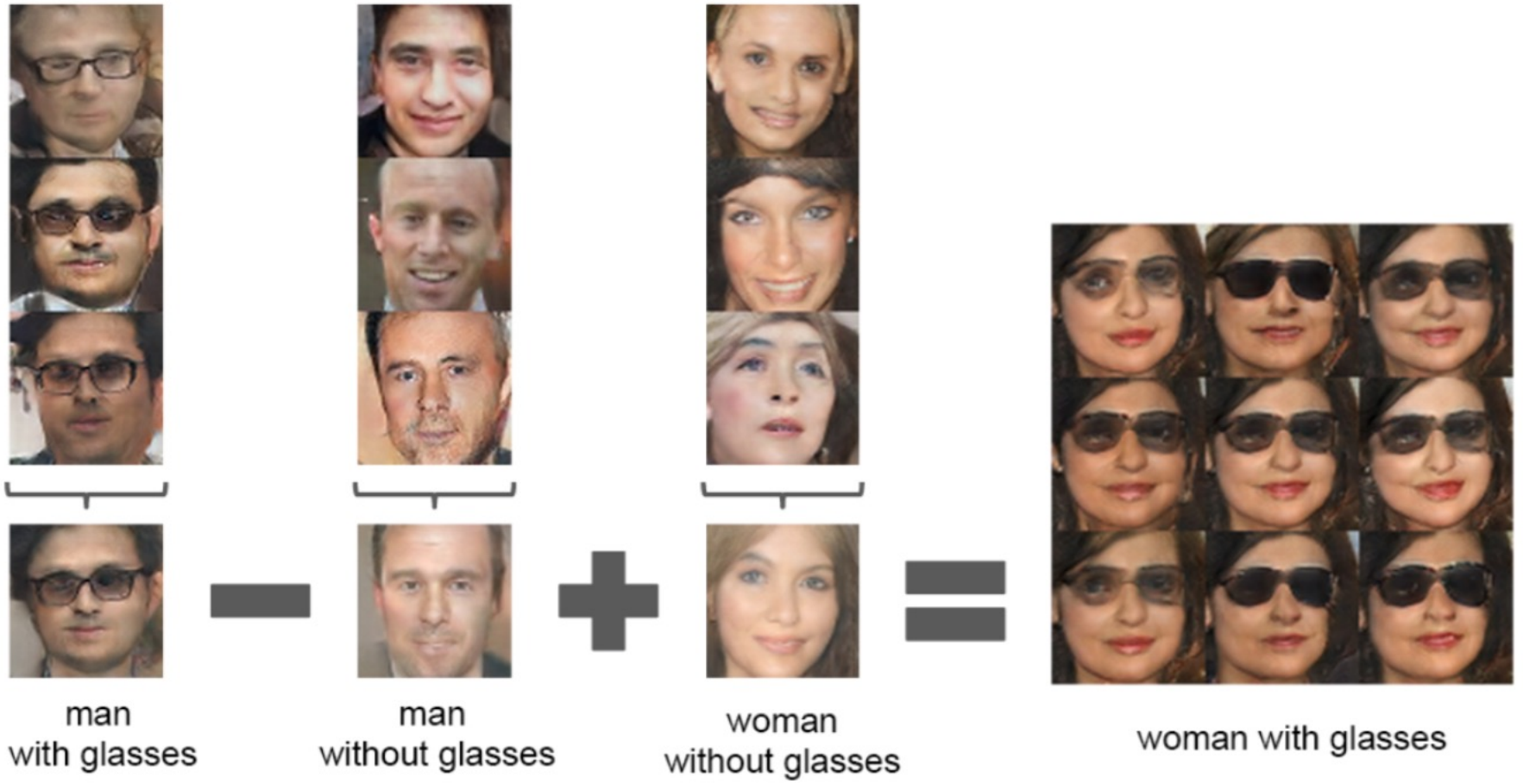
fake

Increased Control Over Generation

- Key to realistic generation is control over which vectors map to which features
- This allows for tuning of noise ”like a knob”
- StyleGAN, InfoGAN and other methods use this approach
- Unsupervised approach:
  - Take a vector that generates a *man* x
  - Take a vector that generates a *man with glasses y*
  - Take a woman z
  - z+(x-y) should put glasses on the woman!

Learning realistic generation

Style GAN



https://www.youtube.com/watch?v=kSLJriaOumA

InfoGAN

- Info GAN tries to provide a naturally disentangled latent code to do this
- In other words it tries to make the knobs separated from the start

Entangled

纠缠不清

Disentangled

理清

Different vectors controls different features one by one

Mutual Information Term

in loss

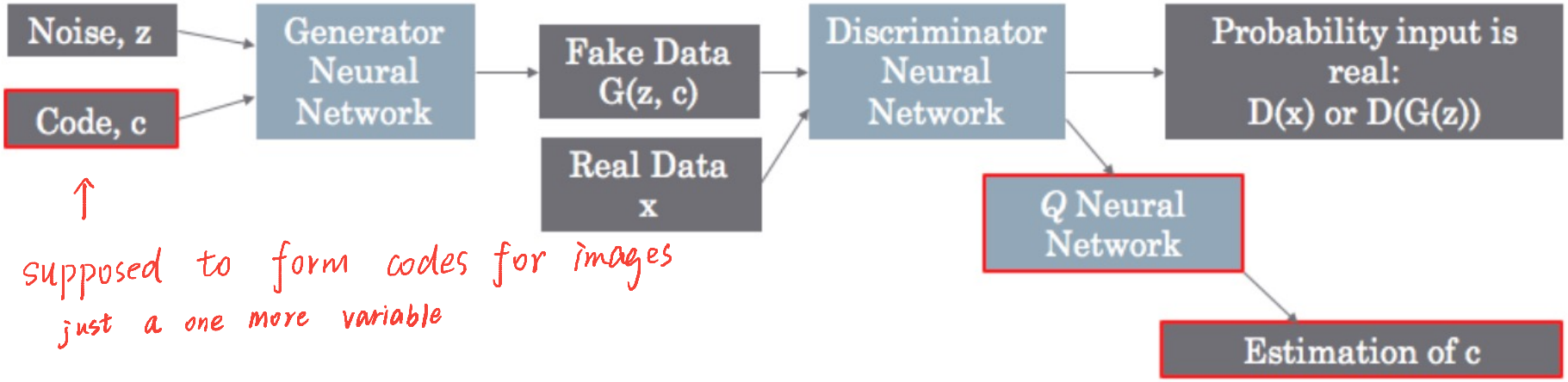
add

$$\min_G \max_D V_I(D,G) = V(D,G) - \lambda I(c; G(z,c))$$

Mutual info between code c and fake data G(z, c)

Actual Architecture

Noise has 2 parts z and c

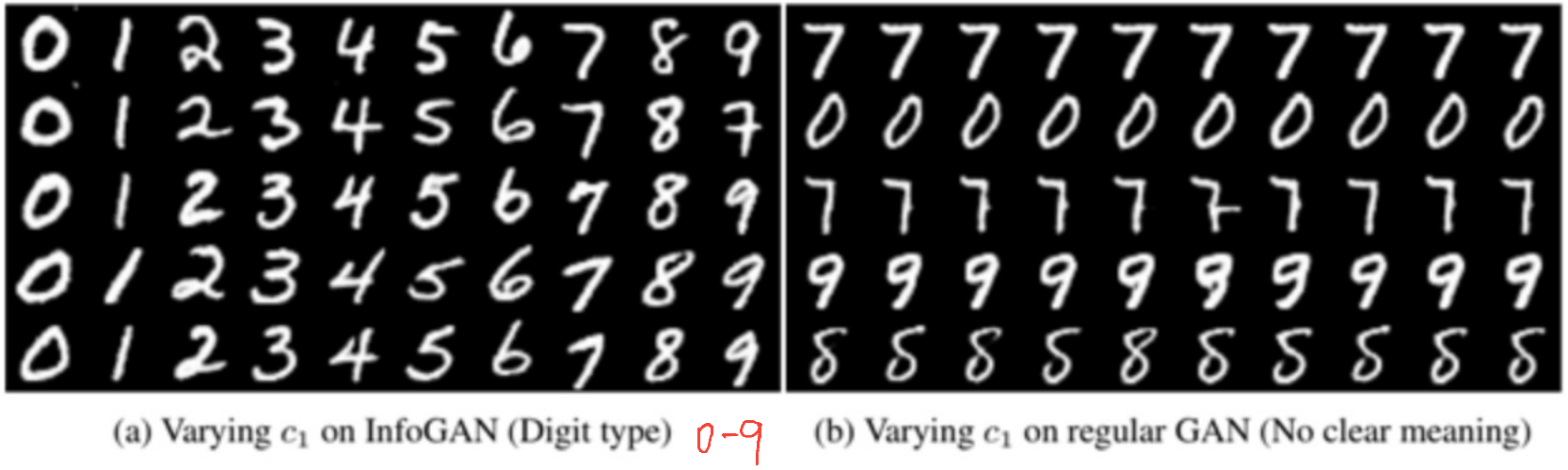


supposed to form codes for images

just a one more variable

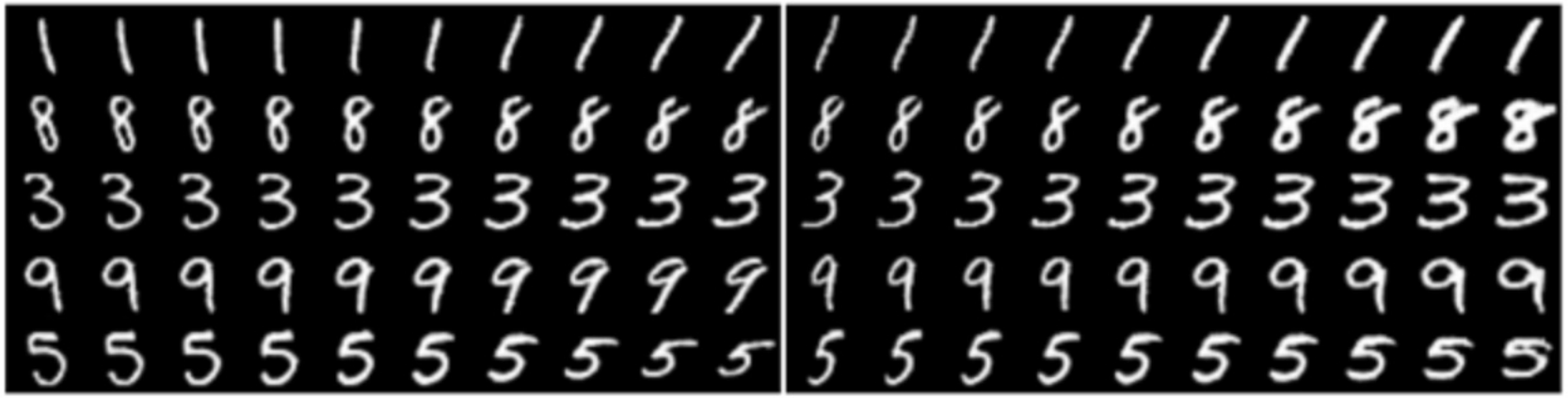
enforce more mutual info between code c and generated data G(z, c)

Knob at work



(a) Varying c_1 on InfoGAN (Digit type) 0-9

(b) Varying c_1 on regular GAN (No clear meaning)



(c) Varying c_2 from -2 to 2 on InfoGAN (Rotation)

(d) Varying c_3 from -2 to 2 on InfoGAN (Width)

Face Angles

(a) Azimuth (pose)

(b) Elevation

(c) Lighting

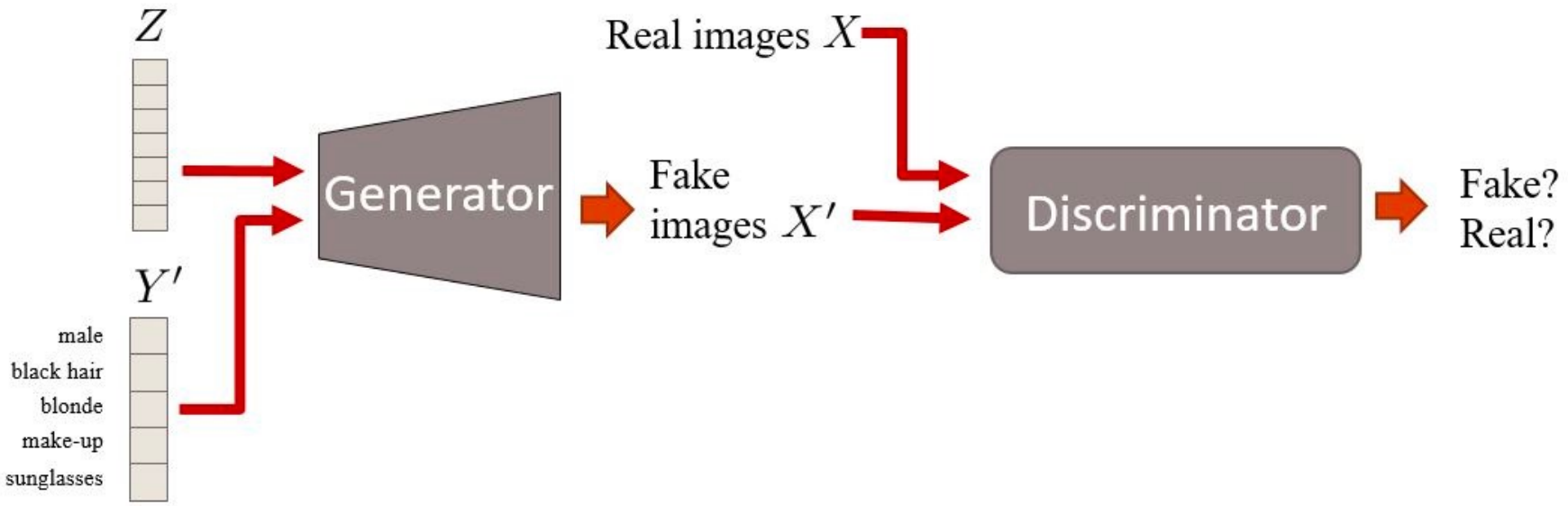
(d) Wide or Narrow

Conditional GANs

- GANs take noise z and turn it into our data x
- Great! But what if we want to control it?
  - “It generated ***another*** image of a 3? I wanted a 7 this time!”
- Conditional GANs:
  - If we have labels, we want to teach it to generate not just any x, but an x of a particular label when given some noise z

Conditional GANs

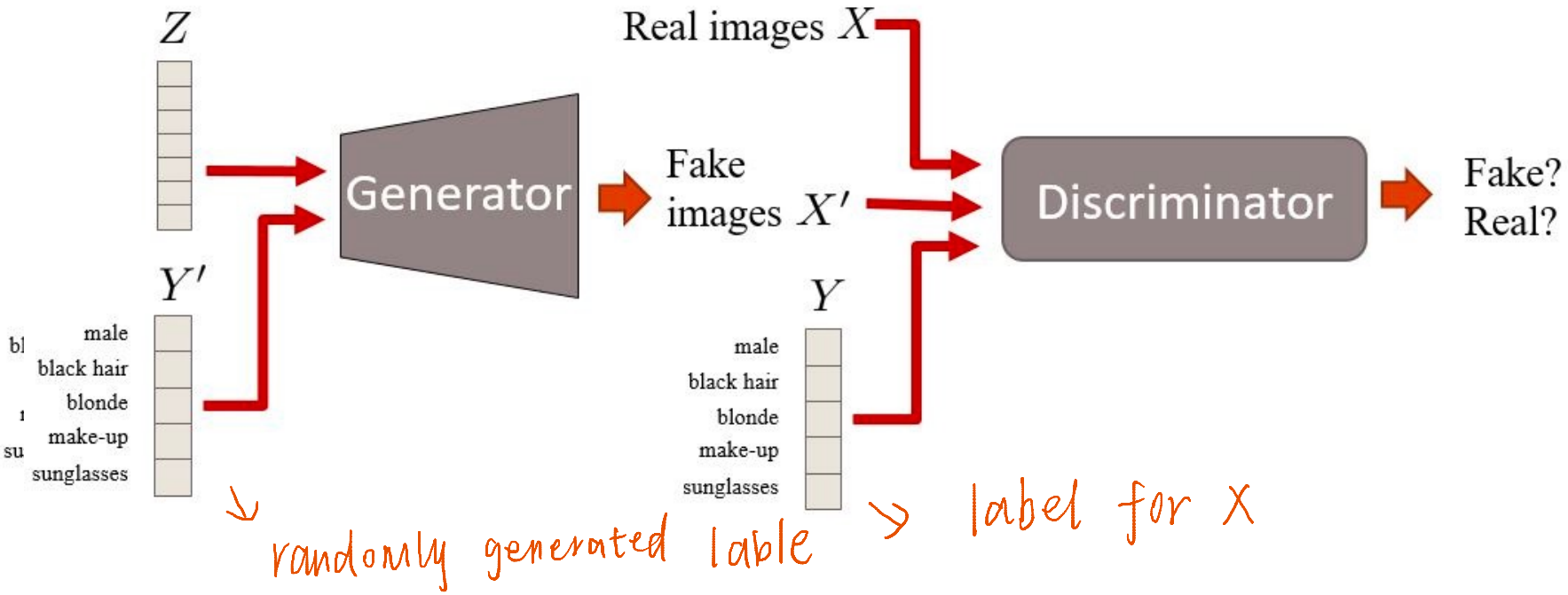
- Feed the generator both a sample from z and the condition
- Discriminator tries to tell real/fake apart as before
  - The generator could generate really realistic images that don't match the condition and the discriminator would never know



http://guimperarnau.com/blog/2017/03/Fantastic-GANs-and-where-to-find-them

Conditional GANs

- Have to also give the discriminator the labels for each point
  - The discriminator not only looks to see if it is realistic, but also if it is an image/condition pair that doesn't appear in real data



http://guimperarnau.com/blog/2017/03/Fantastic-GANs-and-where-to-find-them

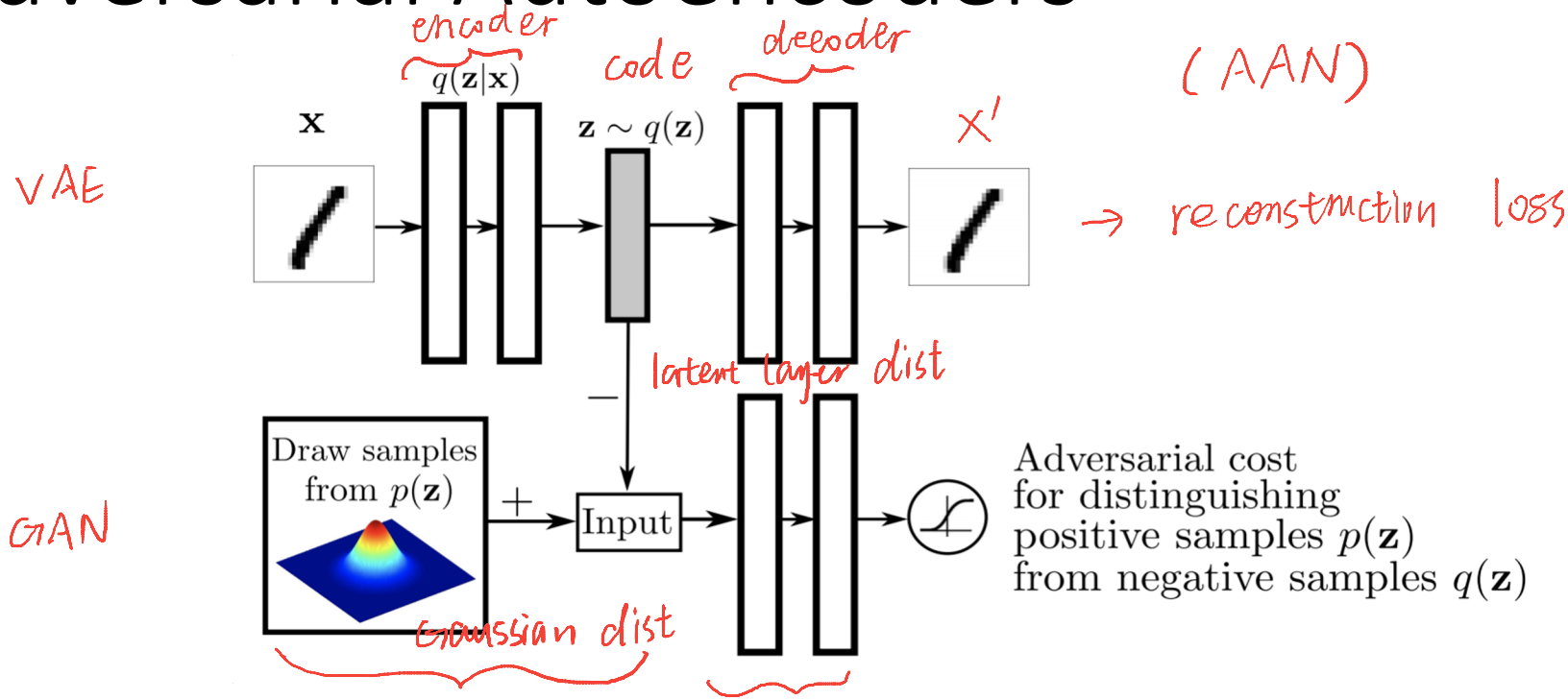
Can adversarial training be used on Autoencoders ?

- Can we use it to make an improved version of a VAE? yes
- Can the role of the KL-divergence penalty be replaced by an adversary? yes

Adversarial Autoencoders

combine VAE & GAN

(AAN)

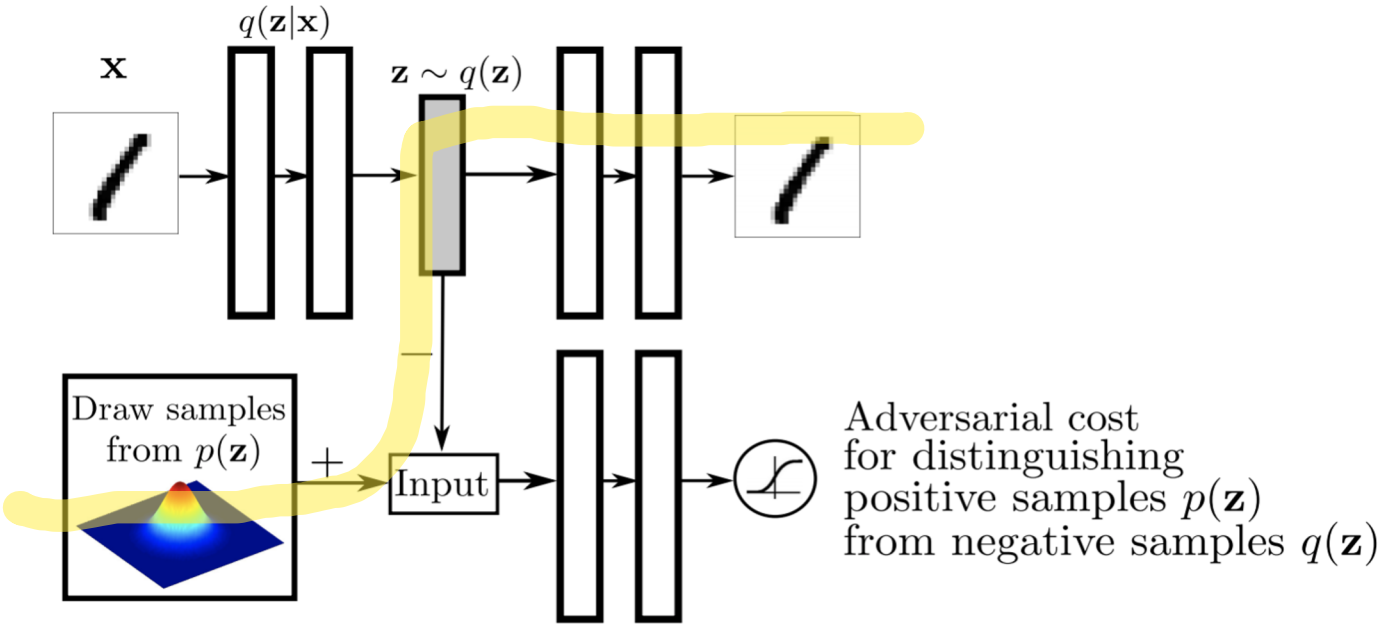


Differences

1. Discriminator classifies samples from **z** against the AE hidden layer (rather than reconstructed samples X')
   a) Low-dimensional and easy distributions
2. Generator has both encoder and decoder
3. Generator receives gradient signal from discriminator **and** reconstruction loss

Makhzani, Alireza, et al. "Adversarial autoencoders." *arXiv preprint arXiv:1511.05644* (2015).

Adversarial Autoencoders



- Generate new sample as follows:
  1. Sample from $p(\boldsymbol{z})$
  2. Decode $\boldsymbol{z}$
- Encoder and discriminator only needed for training, not sampling

Makhzani, Alireza, et al. "Adversarial autoencoders." *arXiv preprint arXiv:1511.05644* (2015).

Other advantages of AANs

- The discriminator has a simpler job: just needs to distinguish between latent layer distribution and gaussian distribution $p(z)$ $q(z)$

- Has denoising and other advantages of an autoencoder

Domain Transfer

- Refers to the transfer of data from one domain to another
- Example: Style Transfer

From Gatys et al

DiscoGAN: Motivation

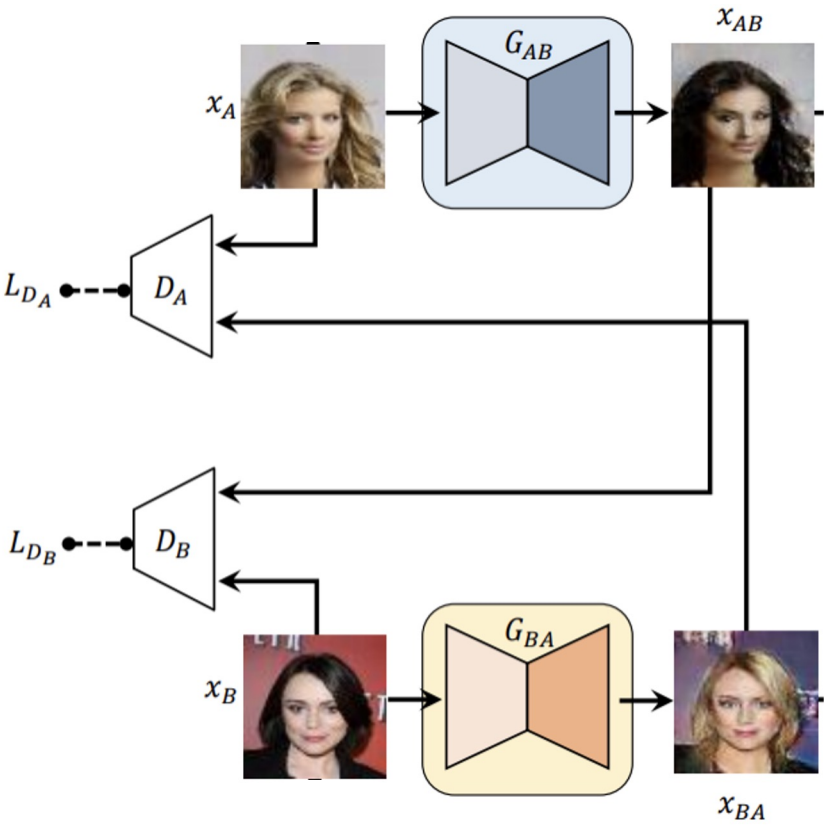
- GANs and adversarial AEs sample from one domain
- What if we have two domains, A and B?
- Can we learn to turn a point in A into a point that looks like its from B?
- Other attempts required paired input (a_i, b_i)
  - Uses are limited since labels can be expensive or difficult to obtain
- Can we do this with an unsupervised training setup?

DiscoGAN: Motivation

- Domain A: blond hair

- Domain B: black hair

Previous approach

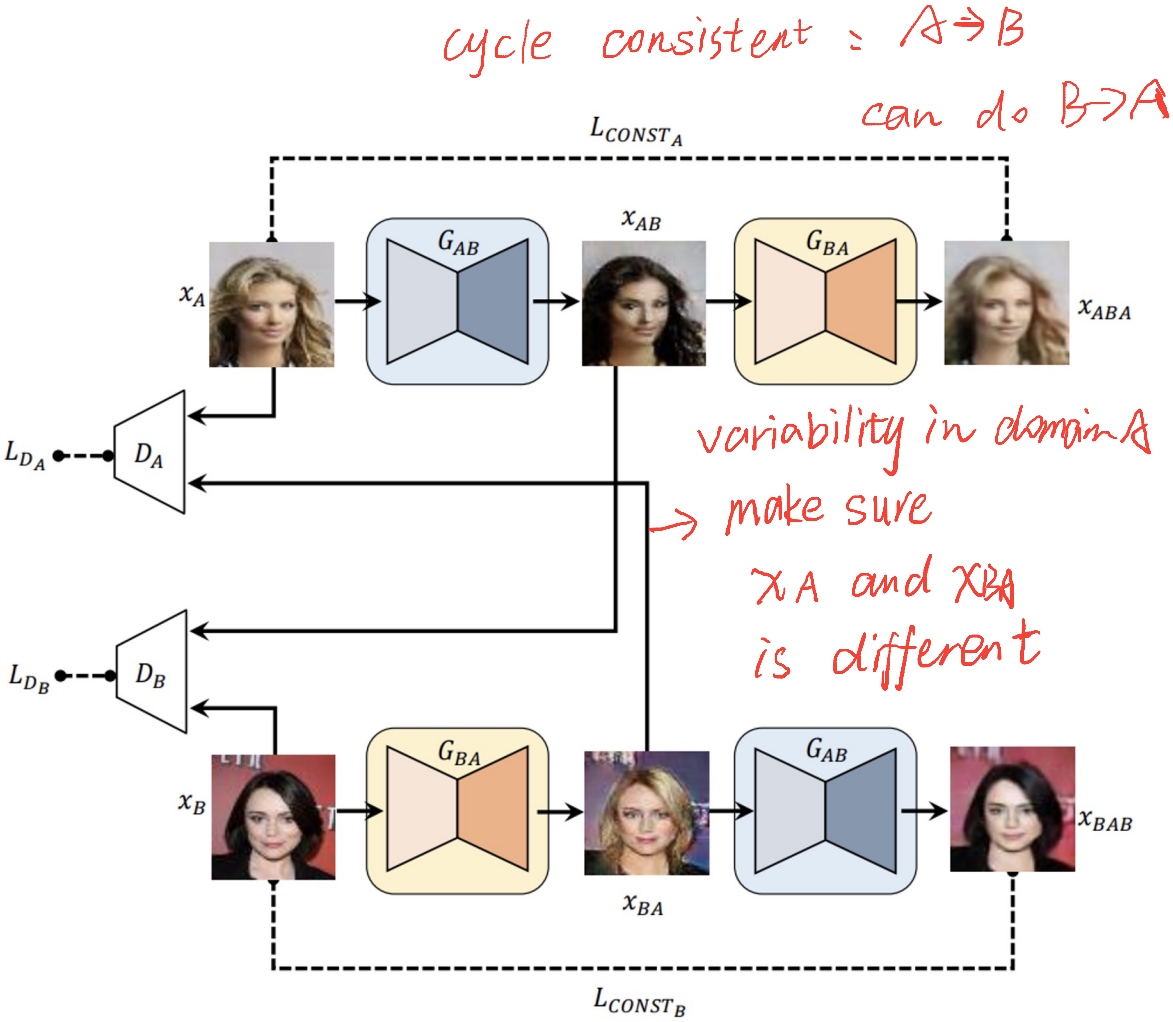


Kim, Taeksoo, et al. "Learning to discover cross-domain relations with generative adversarial networks." *arXiv preprint arXiv:1703.05192* (2017).

DiscoGAN

- Domain A: blond hair

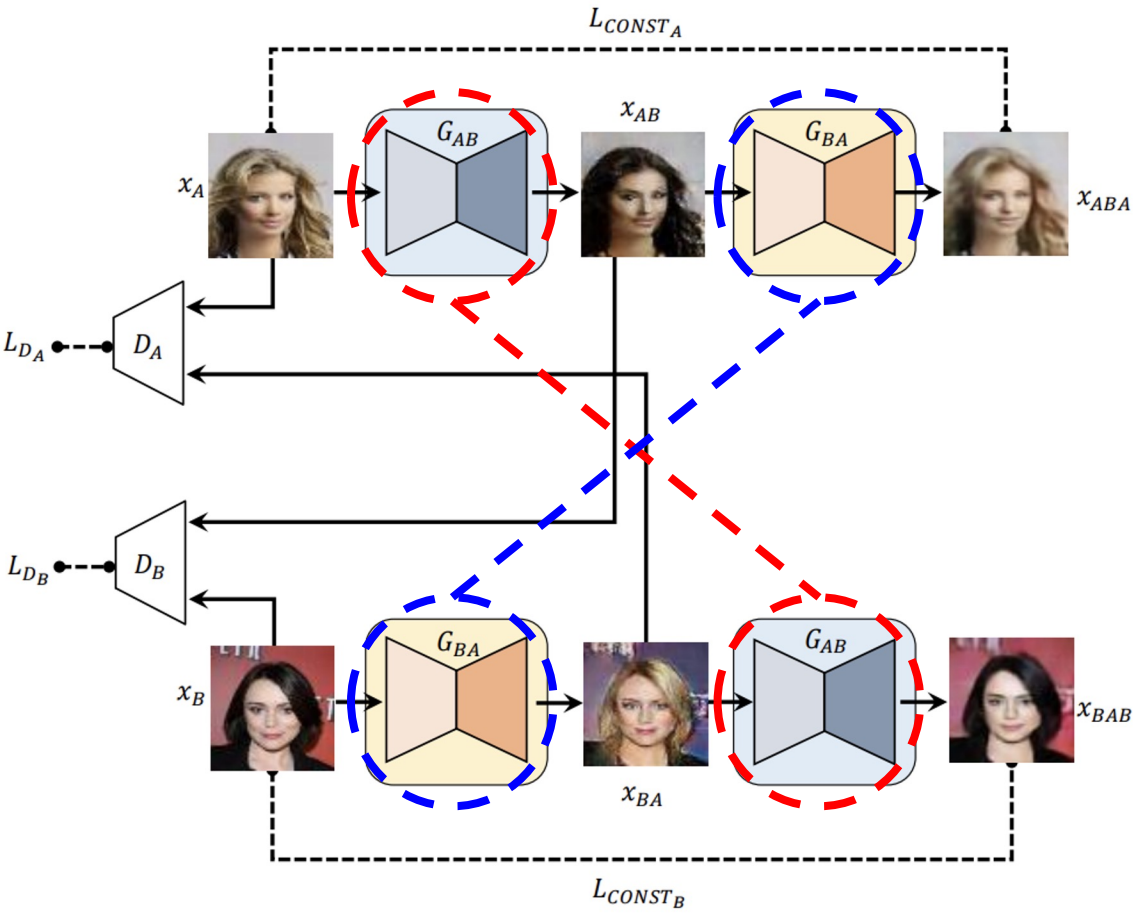
- Domain B: black hair



Kim, Taeksoo, et al. "Learning to discover cross-domain relations with generative adversarial networks." *arXiv preprint arXiv:1703.05192* (2017).

DiscoGAN

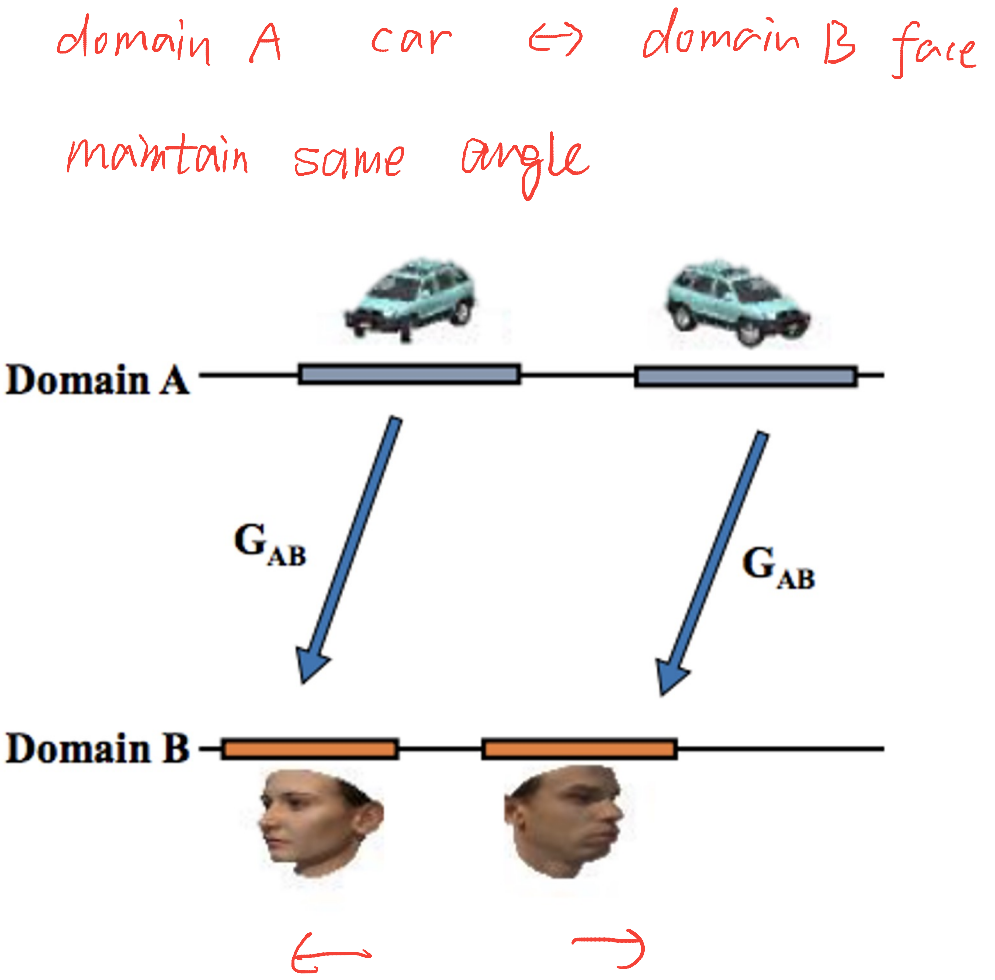
- Weight sharing/reuse is important easier to train
- Same network weights/activations must perform multiple tasks
  - Indirectly increases number of training samples and generalizability
- Reduces number of weights to train
  - Optimization is easier



Kim, Taeksoo, et al. "Learning to discover cross-domain relations with generative adversarial networks." *arXiv preprint arXiv:1703.05192* (2017).

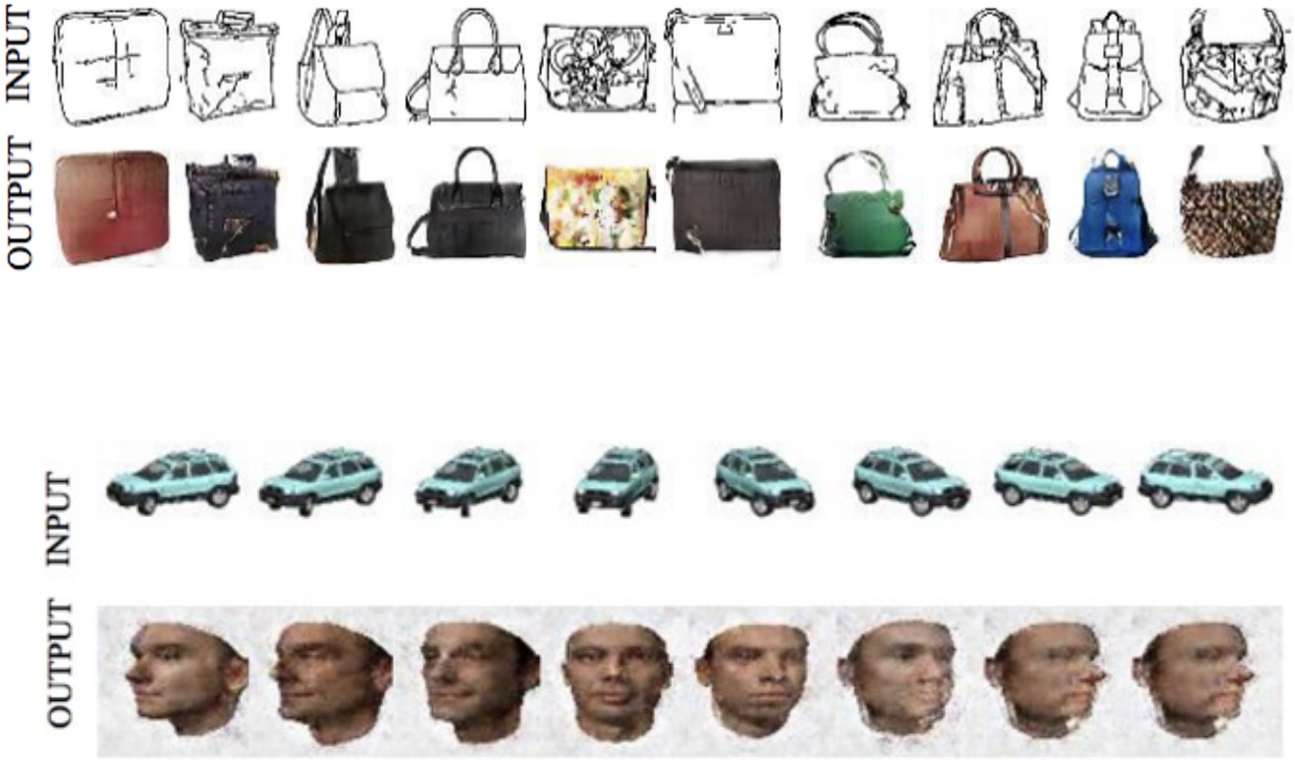
DiscoGAN

- Weight sharing/reuse is important
  - Same reason CNNs work
- Same network weights/activations must perform multiple tasks
  - Indirectly increases number of training samples and generalizability
- Reduces number of weights to train
  - Optimization is easier



Kim, Taeksoo, et al. "Learning to discover cross-domain relations with generative adversarial networks." *arXiv preprint arXiv:1703.05192* (2017).

DiscoGAN



colorization

DiscoGAN

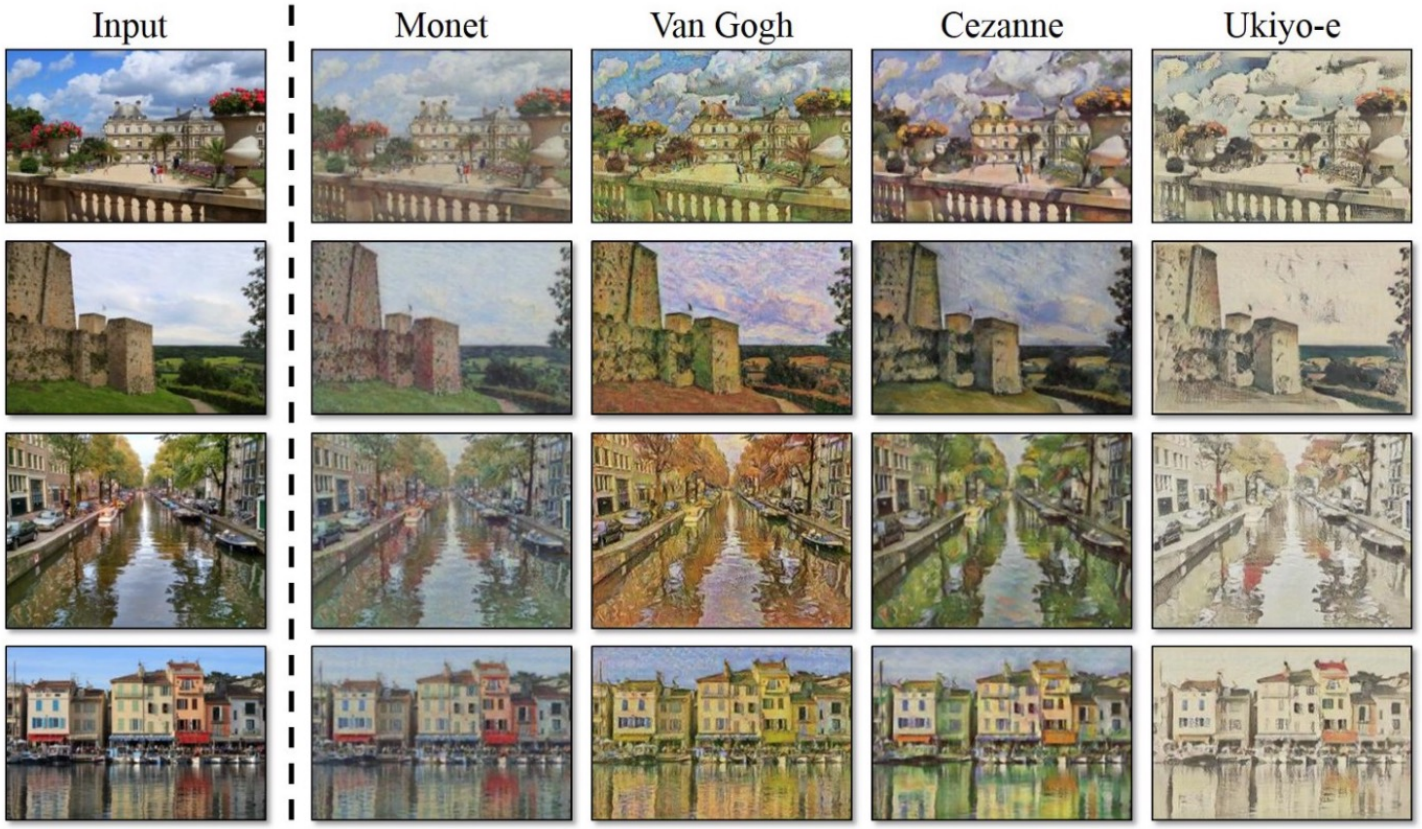
Issues

- Is this the correct shoe for this purse?
- Is there a better match?
- Original paper doesn't talk about it

- Sometimes the mapping isn't correct

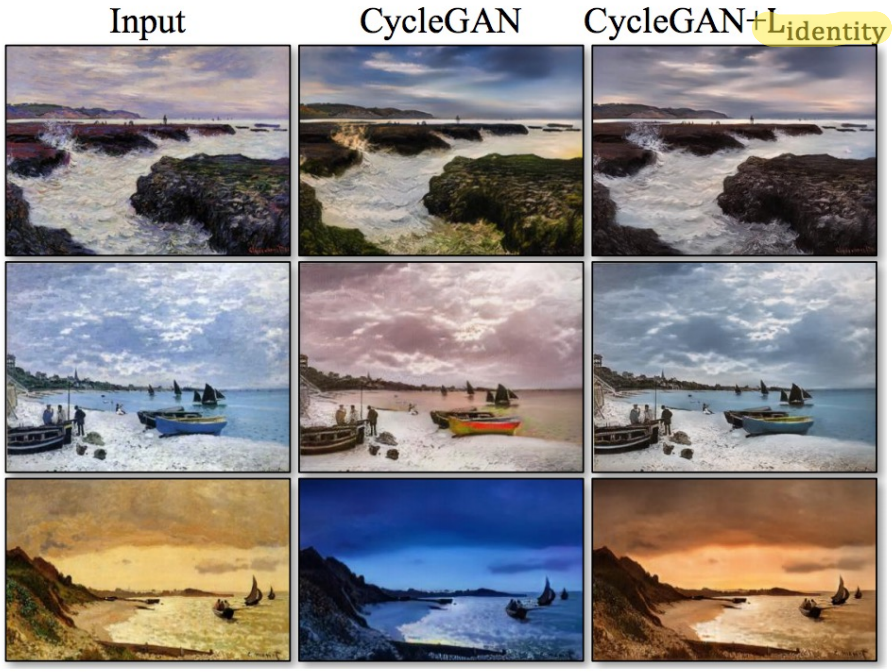
CycleGANs for style transfer

- CycleGANs (generically) use two gans, one to transfer from domain A to domain B and one to transfer B to A



CycleGAN

- Came out shortly after DiscoGAN, popularized the idea
- Added another component to the loss:
  - Normally G_{AB} takes x_A to domain B
  - This means $G_{AB}(x_A)$ will be different from x_A
  - Add additional constraint that $G_{AB}(x_B) = x_B$
    - Intuitively, G should take something not in A to B, but if it's already in B, it shouldn't change it



Zhu, Jun-Yan, et al. "Unpaired image-to-image translation using cycle-consistent adversarial networks." *Proceedings of the IEEE International Conference on Computer Vision*. 2017.

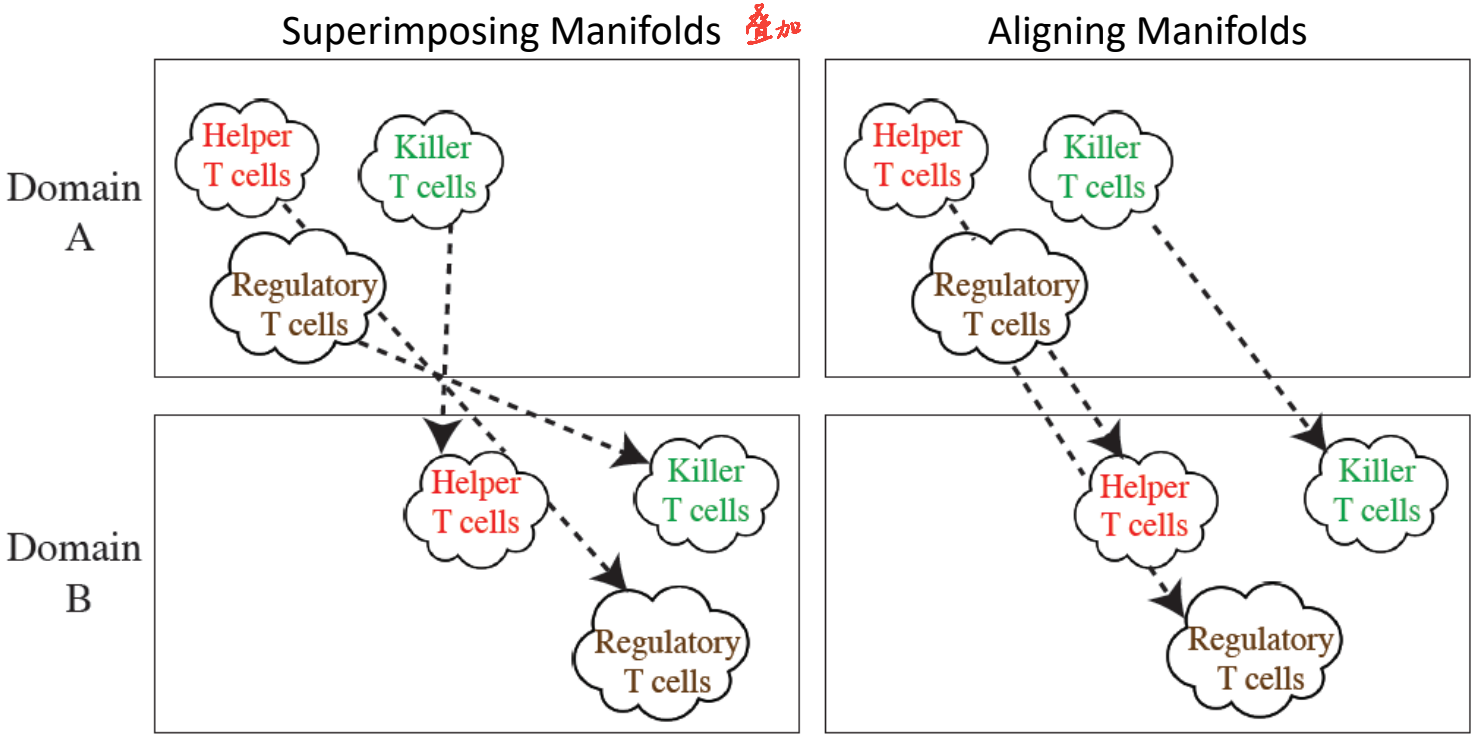
Limitations of Cycle-consistency

not necessary unless domain A and B are very different

- Can only learn simple **style-transfer** transformations
  - e.g. paint a horse to look like a zebra
- Can only learn ***one*** transformation
  - e.g. all horses become zebras, and nothing else is transformed
  - Thus domains must be homogenous
  - Papers do a lot of cropping and centering…
- Transform based on the **pixel space**
  - Is MSE between two 256x256x3-dimensional images meaningful?
- Assumes **any invertible** transformation is meaningful
  - What does invertibility have to do with semantic meaning?
- Not **interpretable**
  - Why did individual *i* in domain 1 get paired with this particular individual in domain 2? CycleGAN offers no insight into its decision

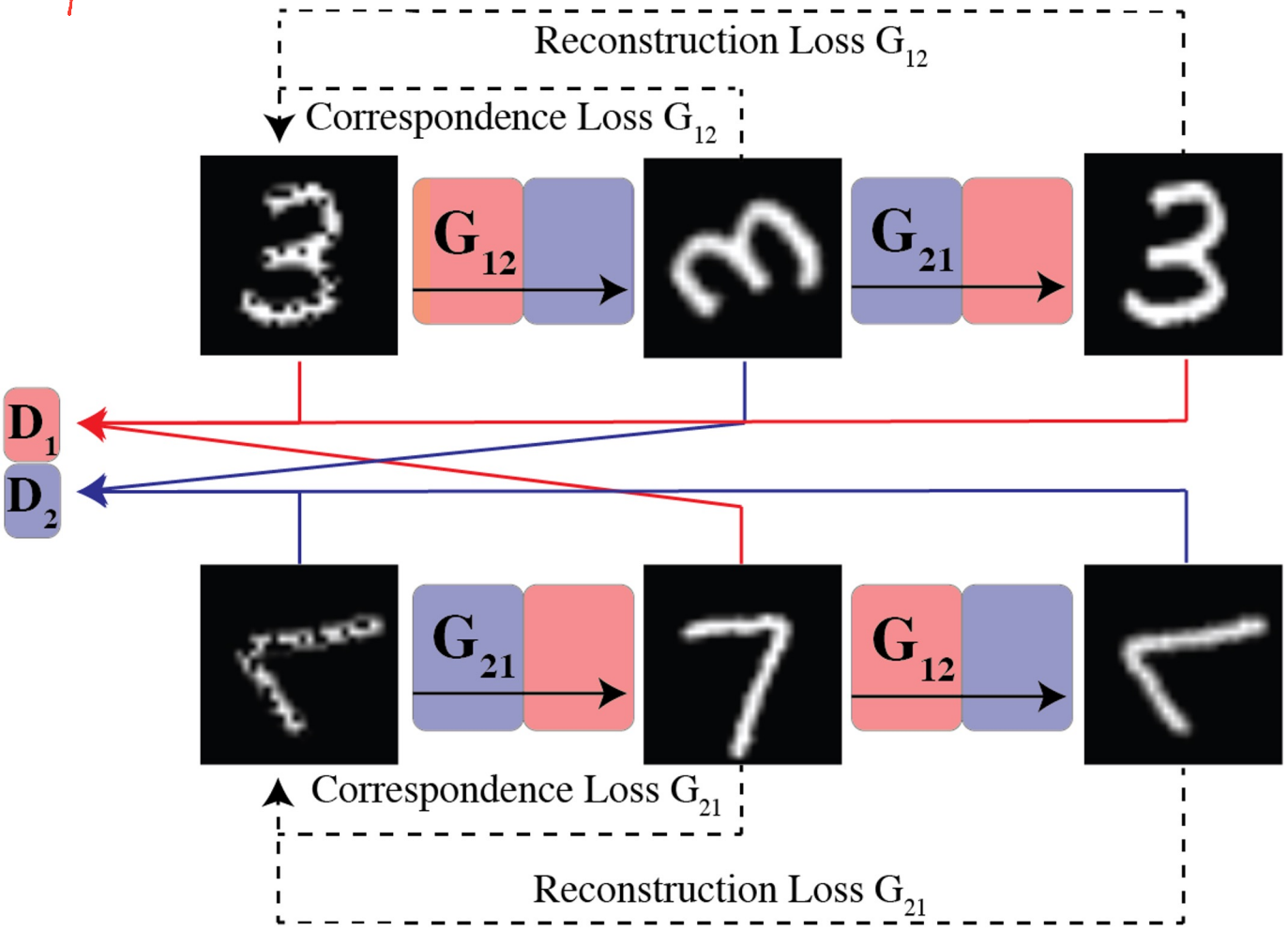
Manifold Alignment GAN (MAGAN)

- Amodio & Krishnaswamy (2018)
- Consider manifolds A and B to be two CyTOF panels
  - Different markers, samples from same tissue
- Panel A measures 40 markers
- Panel B measures 40 markers
- 10 shared markers and 30 unique markers, each
- Using 10 shared markers, find corresponding cells across panels



MAGAN Architecture

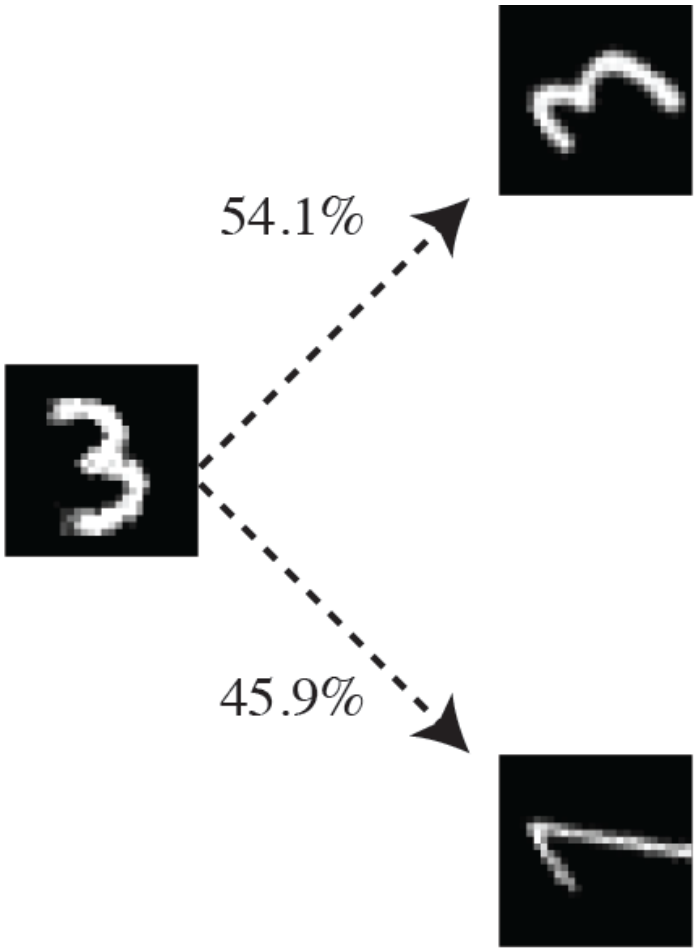
keep 2 GANs and reconstruction loss
add a correspondense loss measures correlation between input and generated data



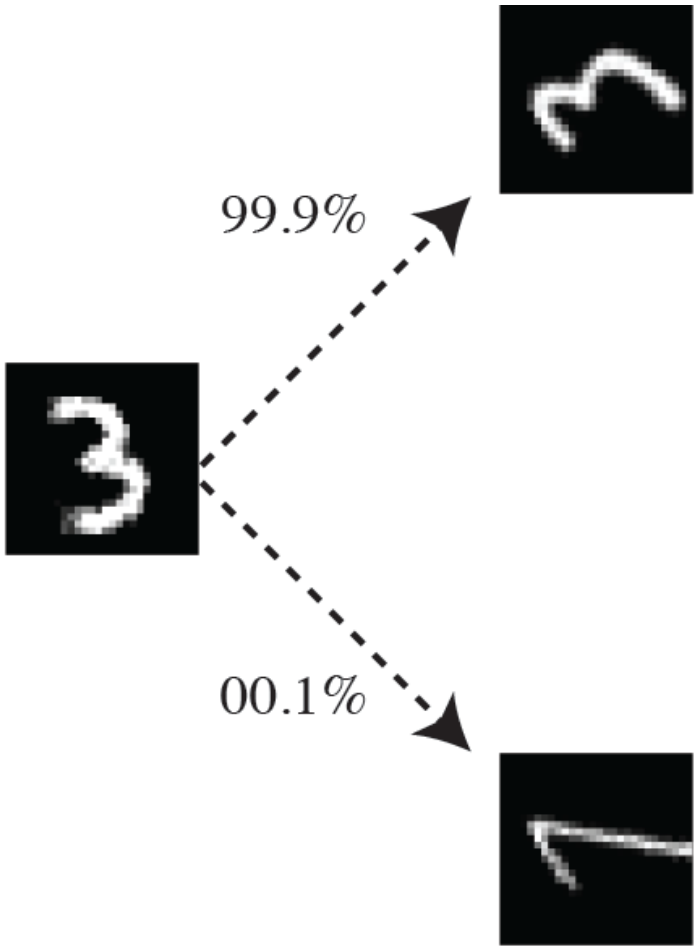
Matching rotated digits

- 100 simulations of each model

Here correspondence is digit identity



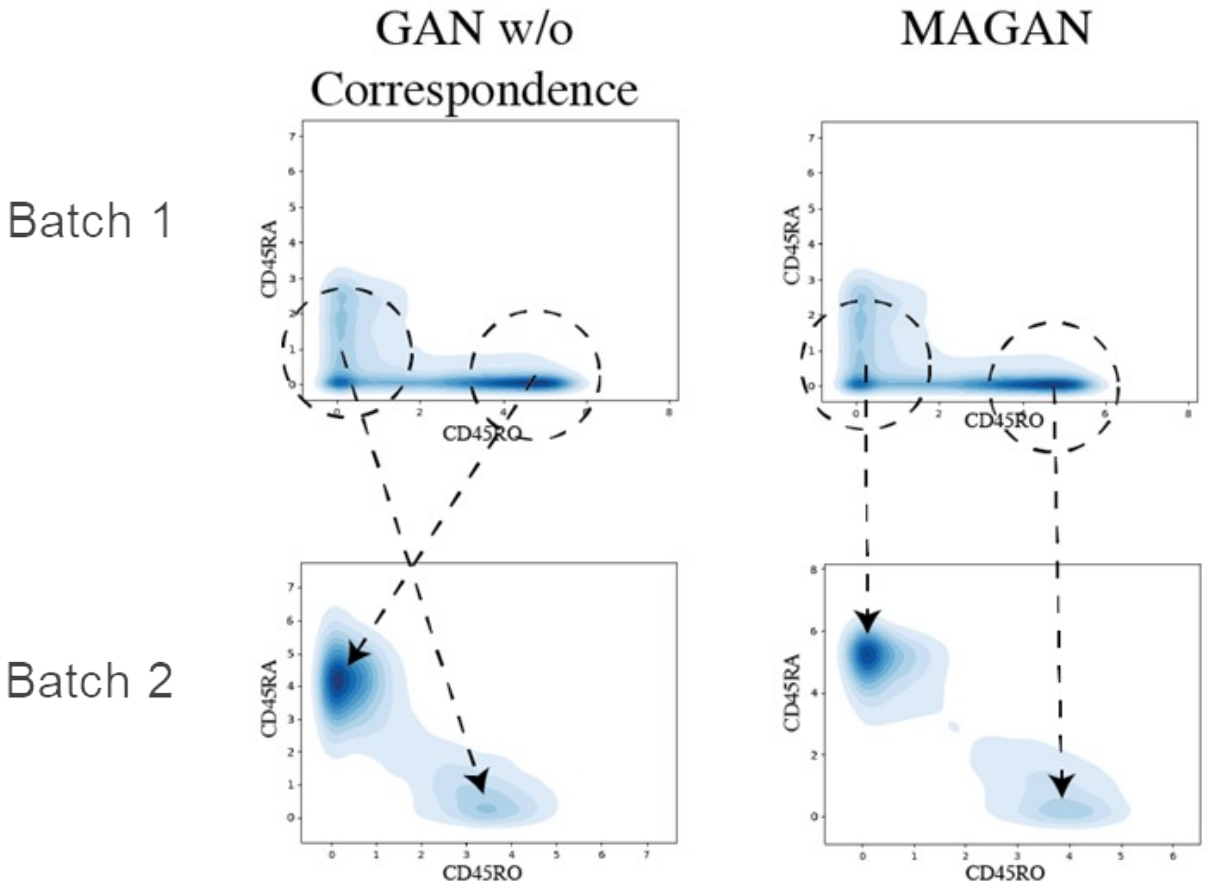
GAN w/o Correspondence



MAGAN

Aligning CyTOF Batches w/ MAGAN

- Two underlying cell populations measured in each batch (naïve T cells and central memory T cells)
  - Dropout in CD45RA in batch 1
- MAGAN correctly matches the cell types in the two batches



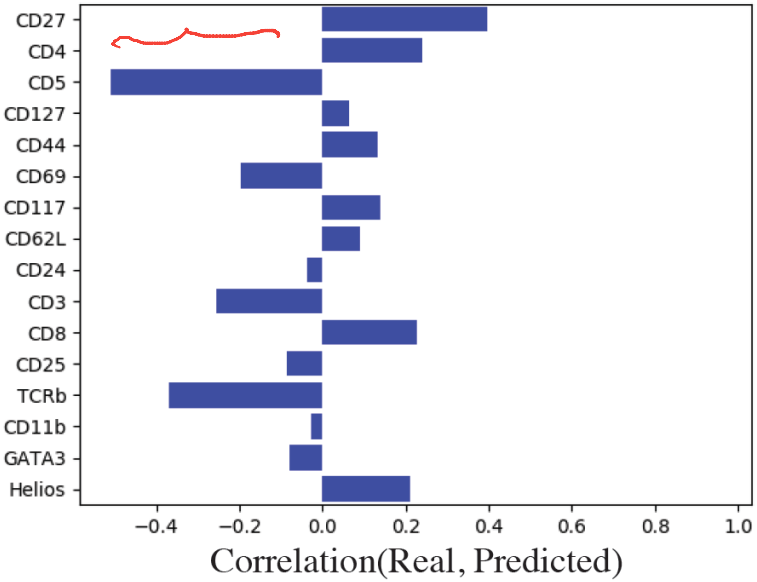
Aligning CyTOF Panels w/ MAGAN

- Validation on publicly available CyTOF data from developing mouse brain*
- Hold out a shared marker in one panel
- Apply MAGAN to reconstruct held-out marker
  - Compare correlation between real value and prediction

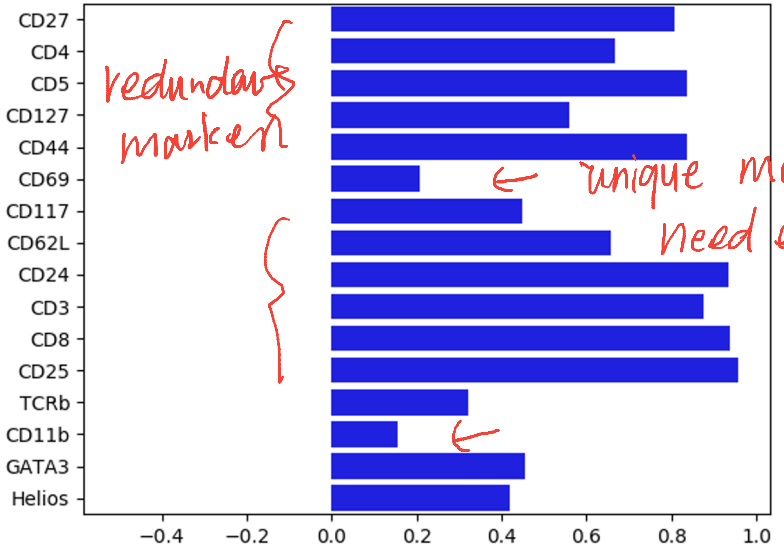
strong negative correlation coz symmetry in mapping

pos correlation

GAN w/o Correspondence



MAGAN



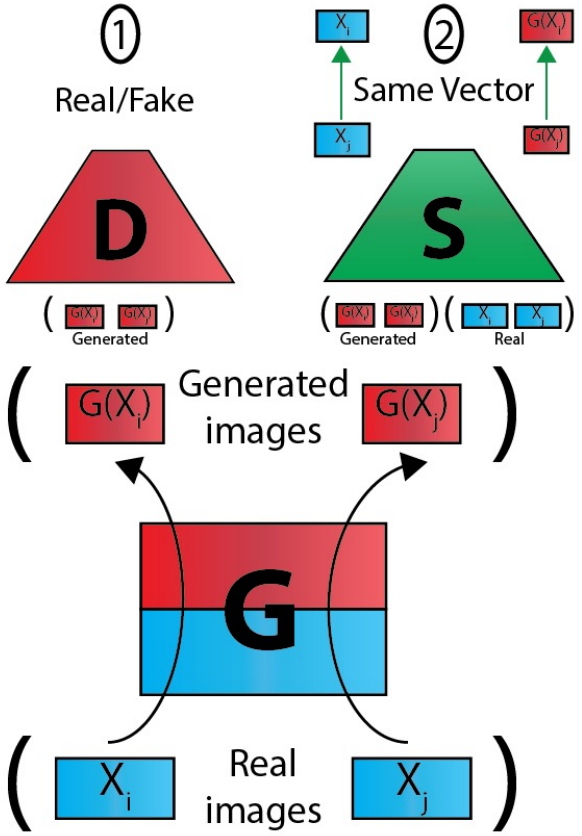
* Setty, Manu et al. "Wishbone identifies bifurcating developmental trajectories from single-cell data". *Nature Biotechnology.*

Limitations of Cycle-consistency

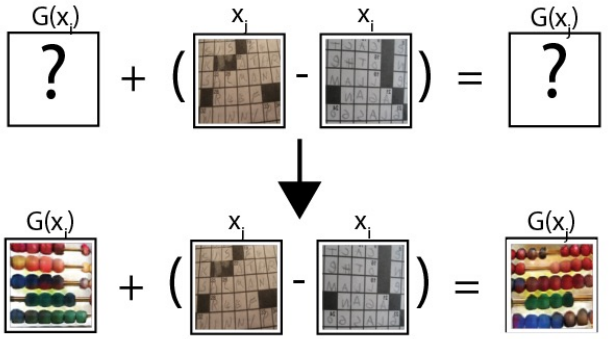
- Can only learn simple **style-transfer** transformations
  - e.g. paint a horse to look like a zebra
- Can only learn ***one*** transformation
  - e.g. all horses become zebras, and nothing else is transformed
  - Thus domains must be homogenous
  - Papers do a lot of cropping and centering…
- Transform based on the **pixel space**
  - Is MSE between two 256x256x3-dimensional images meaningful?
- Assumes **any invertible** transformation is meaningful
  - What does invertibility have to do with semantic meaning?
- Not **interpretable**
  - Why did individual *i* in domain 1 get paired with this particular individual in domain 2? CycleGAN offers no insight into its decision

TraVeLGAN

- Instead of enforcing cycle-consistency, enforce that the vector between two real points is the same as the vector between their corresponding two generated points
  - Key: do this in a latent space, not pixel space!
  - Where do we get this latent space?
- Learn another network (**S**) to go along with G and D to find a latent space where these transformations are preserved
  - Train S alongside the other networks, gradually developing an appropriate semantics
- Since we are no longer enforcing pixel-wise MSE with cycle-consistency, we have a lot more freedom to change the image for domain transfer now

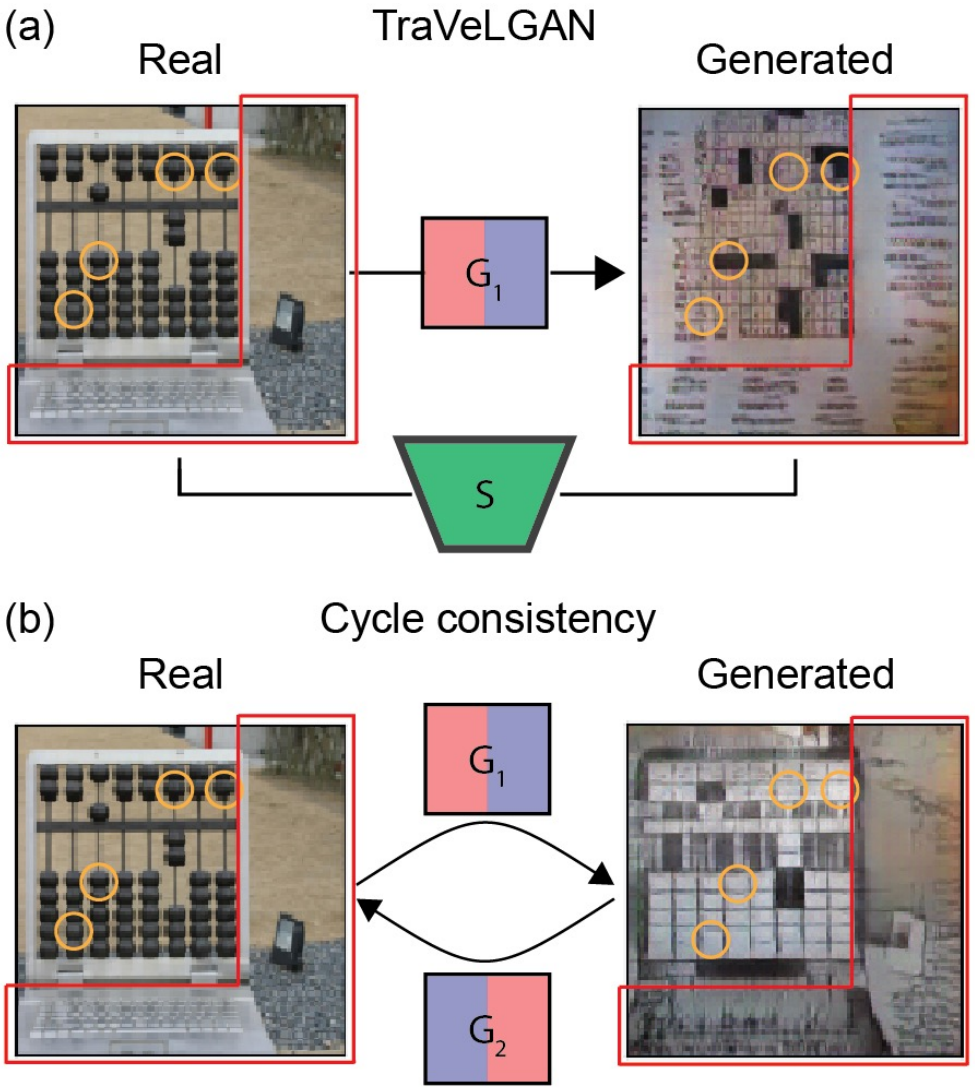


Transformation **Ve**ctor **L**earning (**TraVeL**ing)

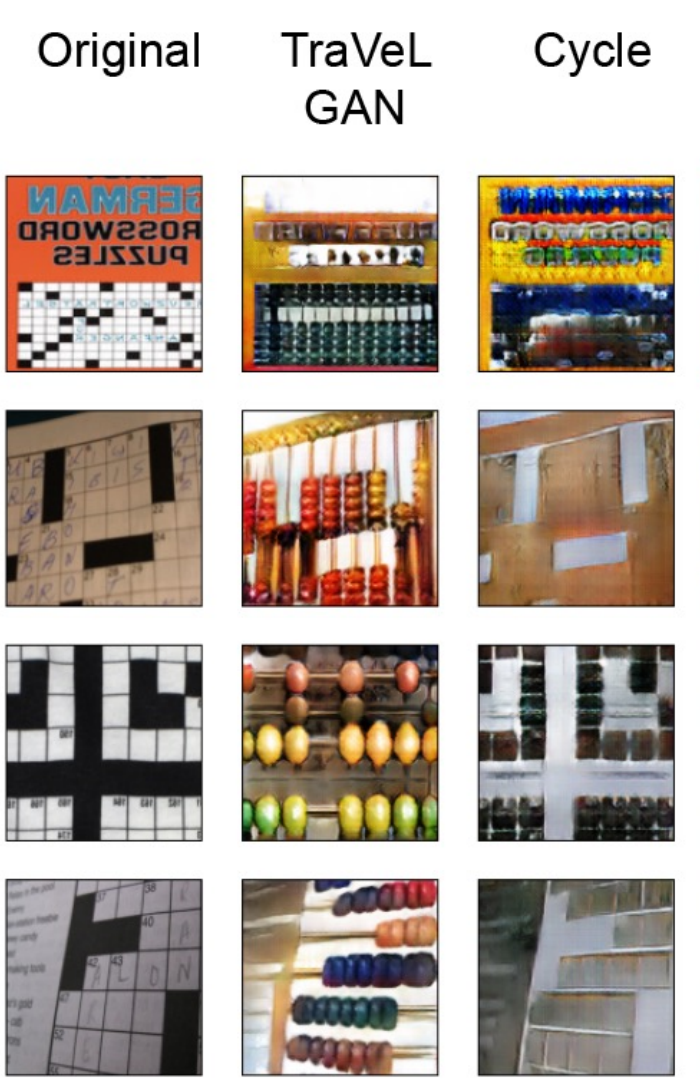


TraVeLGAN: examples

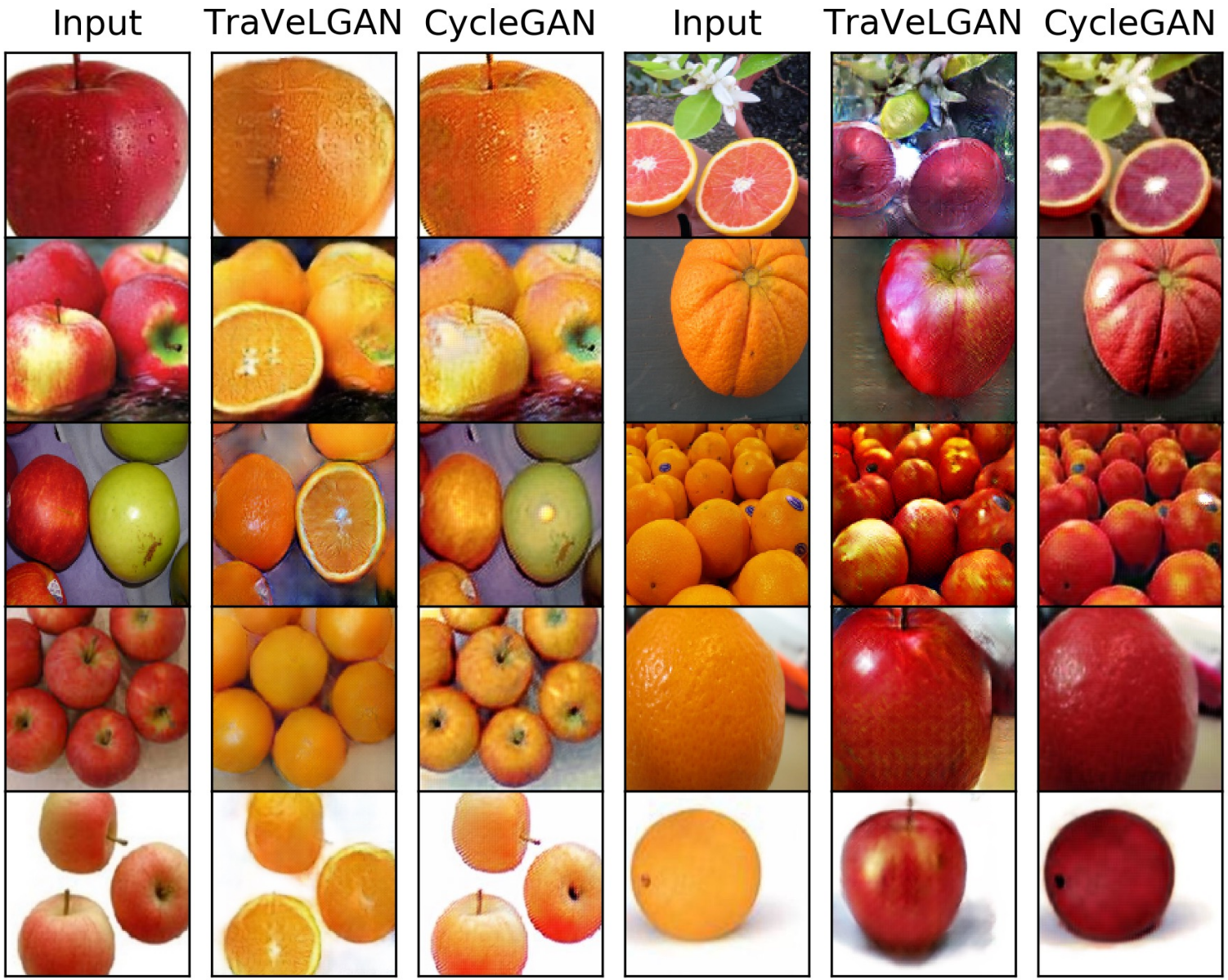
- Transferring abaci to crosswords
- The desired function is not invertible
- Any crossword configuration is a valid abacus configuration, but the reverse is not true
- Changes to the background of an image usually have to be lossy, which we can't have if we require pixel-wise cycle-consistency



TraVeLGAN: examples



TraVeLGAN: examples



Further reading

- Goodfellow et al., Section 20.10.4
- Lectures on GANs CS 11-785 at CMU
- Goodfellow et al., Section 20.10.4
- Lectures on GANs CS 11-785 at CMU
- StyleGAN: https://arxiv.org/pdf/1812.04948.pdf
- InfoGAN: https://arxiv.org/pdf/1606.03657.pdf
- DiscoGAN: https://arxiv.org/abs/1703.05192
- CycleGAN: https://arxiv.org/abs/1703.10593
- MAGAN: http://proceedings.mlr.press/v80/amodio18a.html
- TraVeLGAN: https://arxiv.org/abs/1902.09631
- Conditional GAN: https://arxiv.org/pdf/1411.1784.pdf