

When poll is active, respond at **PollEv.com/yaleml**

Text **YALEML** to **22333** once to join

What's something fun you're looking forward to doing or trying this summer?



Smita Krishnaswamy Daniel Burkhardt



Scott Gigante



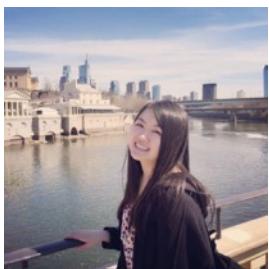
Andrew Benz



Egbert Castro



Pranik Chainani



Joanna Chen



Annie Gao



Michał Gerasimiu



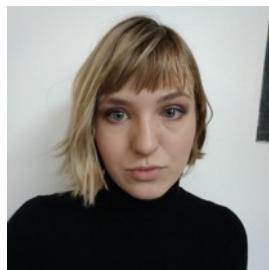
Wes Lewis



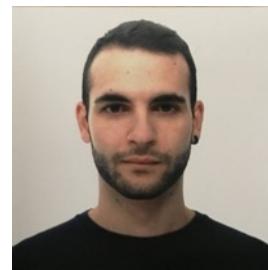
Francesc Lopez



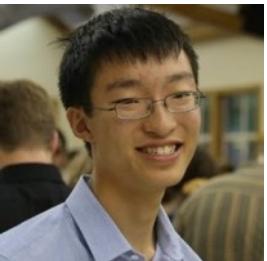
Sameet Mehta



Sasha Safonova



Giacomo Scanavini



Alexander Tong



Aarthi



Max Yuan



Jeffrey Zhou

How to train your network

Training the network

1. Randomly initialize all **weights** and **biases**
2. Run the network on a set of **training data** (batch)
3. Compare the output of the network to the expected output for the input using the **cost function**
4. Calculate the **gradient** of the cost function
5. Update the weights and biases according to the gradient using **backpropagation**
6. Repeat steps 2-5 until **convergence**
7. Compare results to the **validation set** to evaluate the performance of the network
8. If performance on validation set is poor due to **underfitting** or **overfitting**, reconfigure the network **hyperparameters** (e.g. # nodes, # layers, activation function, etc) and repeat 1 - 8

Running the network

1. Run the network on your data
2. If your data is sufficiently different from the training data, you will need to **retrain** the network

End of course survey

Machine Learning
for
Single Cell Analysis

End of course survey

* Required

Were you in a Beginner or Advanced group? *

Beginner

Advanced

Before the workshop, had you analyzed single cell data before? *

Yes

No

What is one thing you learned today? *

Your answer

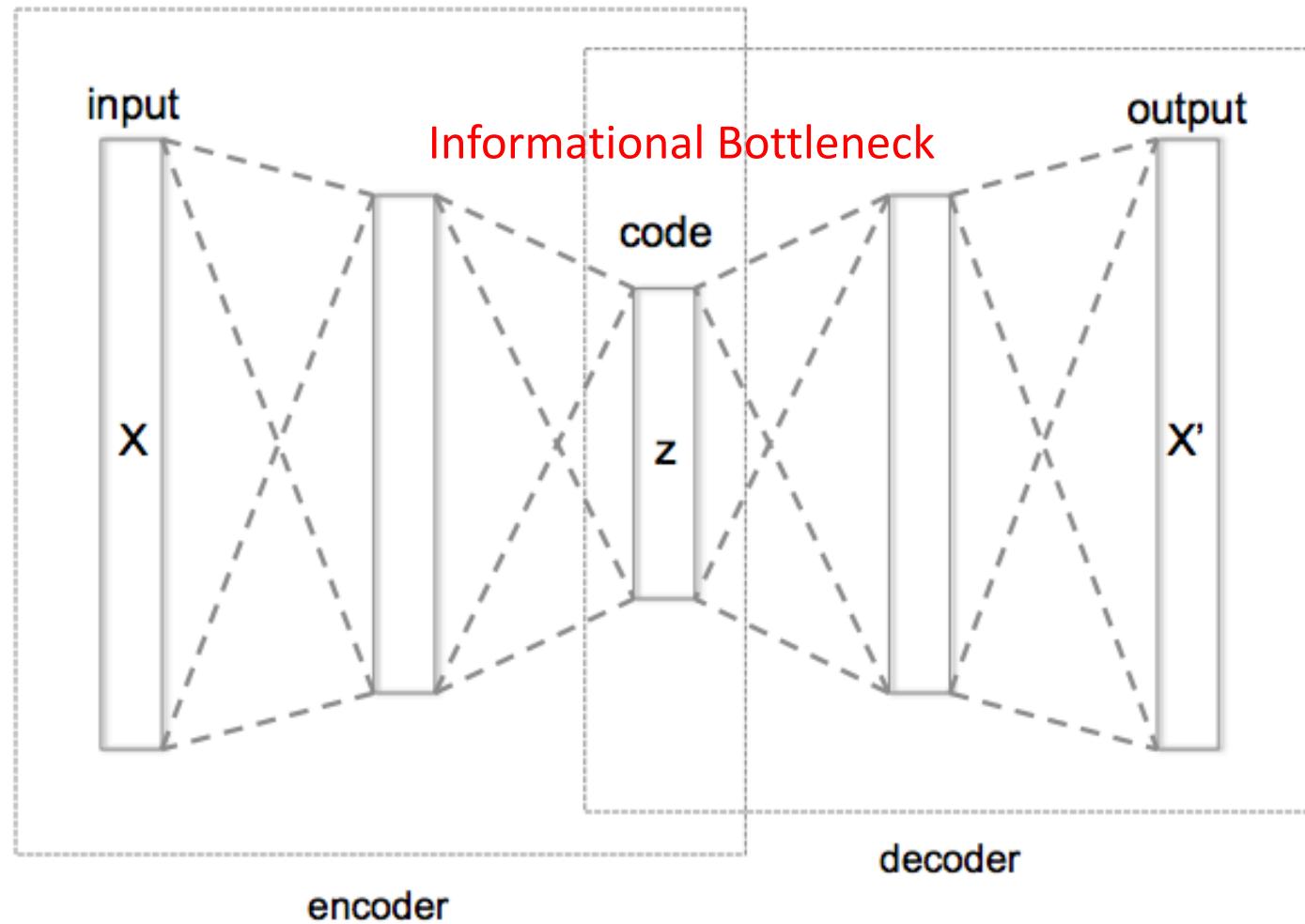
!

!

http://bit.ly/YaleML_Jan2021

Autoencoders, GANs, Conditional GANs, Cycle GANs in Single Cell Analysis

Autoencoder



[Hinton, Salakhutdinov, Science 2006]

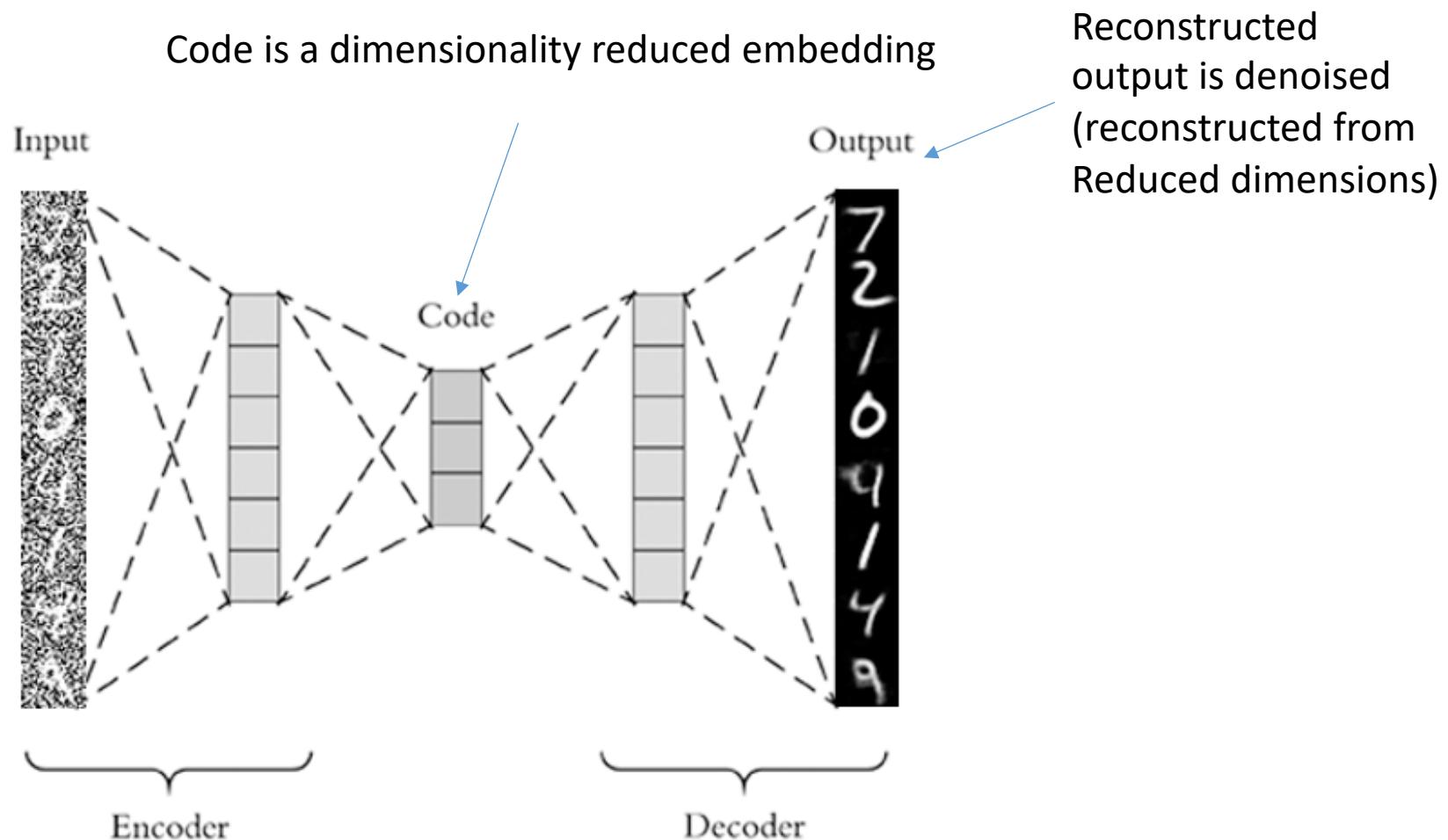
Autoencoder (AE)

- Neural network trained to copy its input x to its output
- Hidden layer z describes a **code** to represent the input
- Two parts
 - Encoder: $z = f(x)$
 - Decoder: $x' = g(z)$
- Goal: minimize $C(x, g(f(x)))$
 - C penalizes $g(f(x))$ for being dissimilar from x
 - Example: mean squared error

Autoencoders for Single-Cell Analysis

- Autoencoders can do most of the single cell tasks that we have talked about so far by construction!
- Dimensionality reduction
- Visualization
- Data denoising
- Batch correction
- Clustering
- (synthetic) Data generation

How?



Remaining tasks need special regularizations

SAUCIE

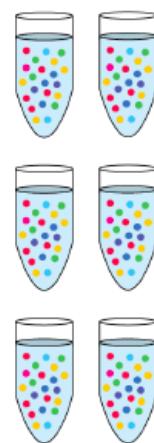


Exploring single-cell data with deep multitasking neural networks

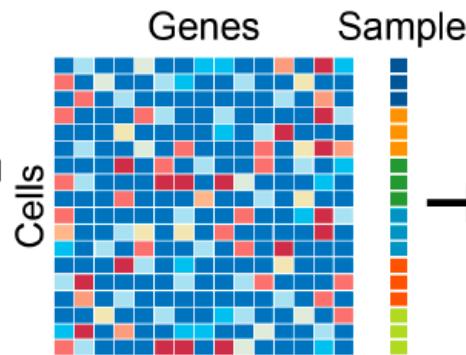
Matthew Amodio, David van Dijk, Krishnan Srinivasan, William S. Chen, Hussein Mohsen, Kevin R. Moon, Allison Campbell, Yujiao Zhao, Xiaomei Wang, Manjunatha Venkataswamy, Anita Desai, V. Ravi, Priti Kumar, Ruth Montgomery, Guy Wolf & Smita Krishnaswamy 

Nature Methods **16**, 1139–1145(2019) | [Cite this article](#)

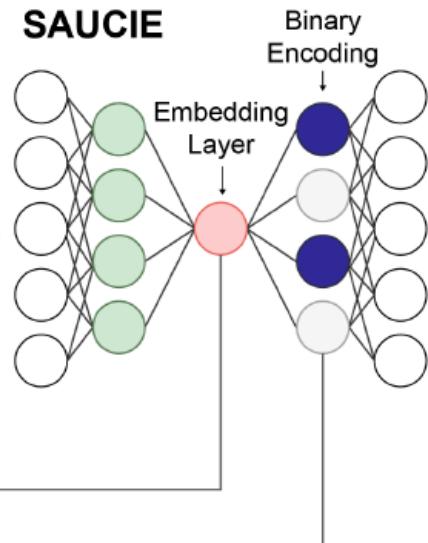
Samples collected from many individuals



scRNA-seq



SAUCIE



Individuals are grouped by cell type proportions

Cell types



Group 1



Group 2

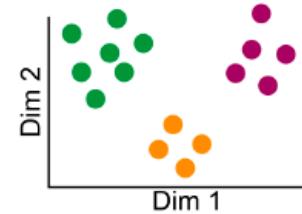


Group 3

→ Cell type identification

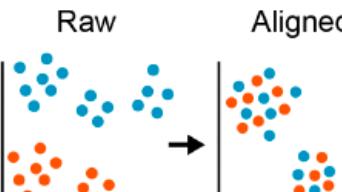


→ Visualization

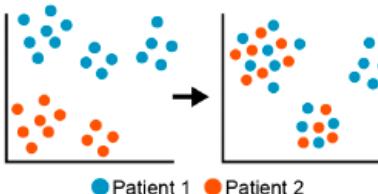


→ Batch Normalization

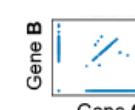
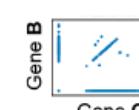
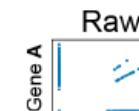
Raw



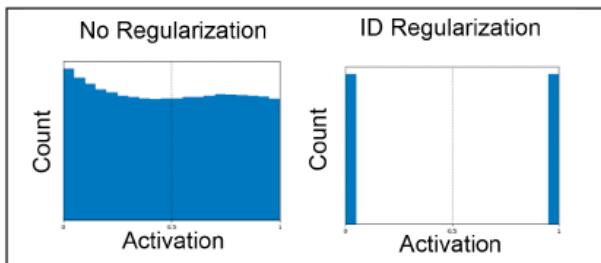
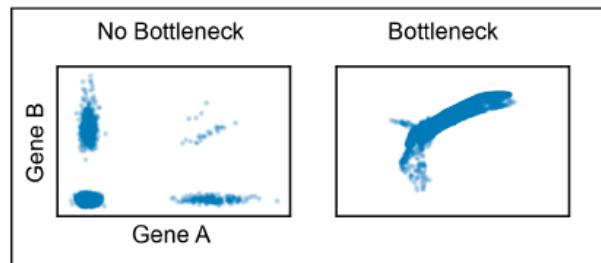
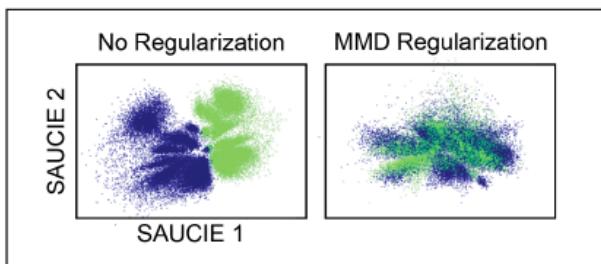
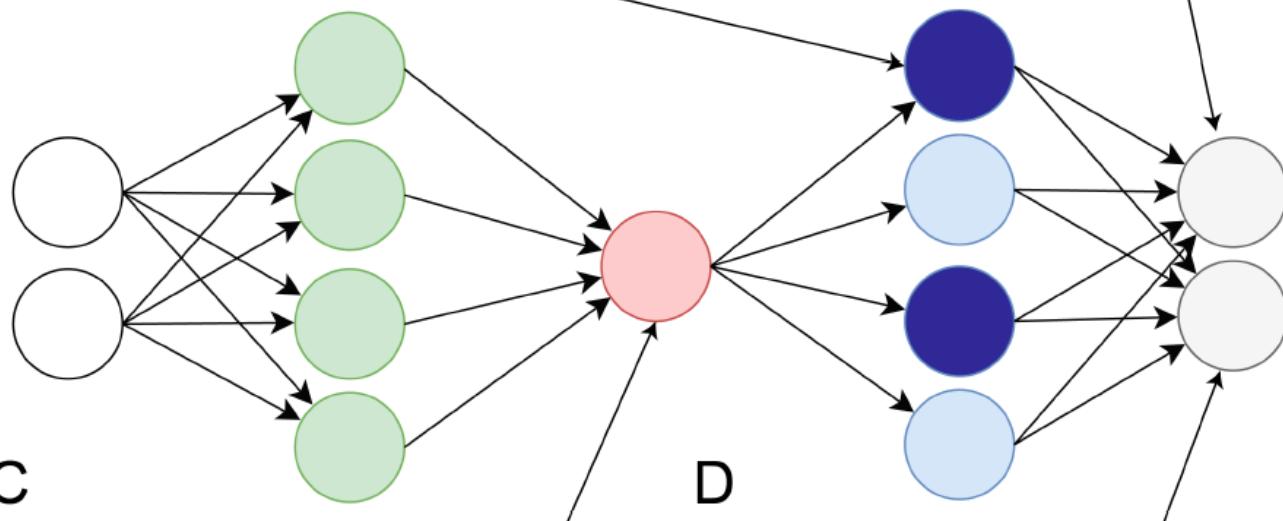
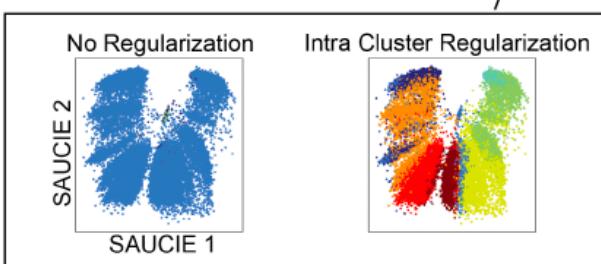
Aligned



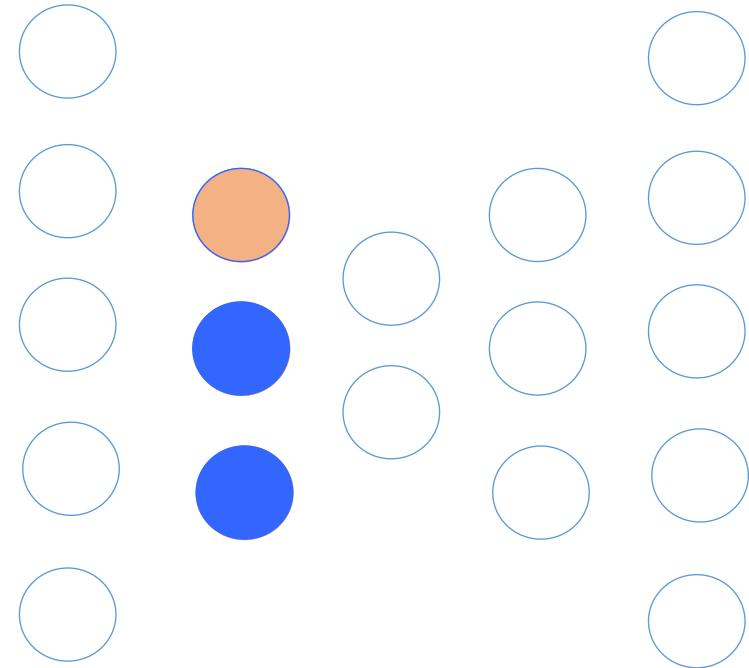
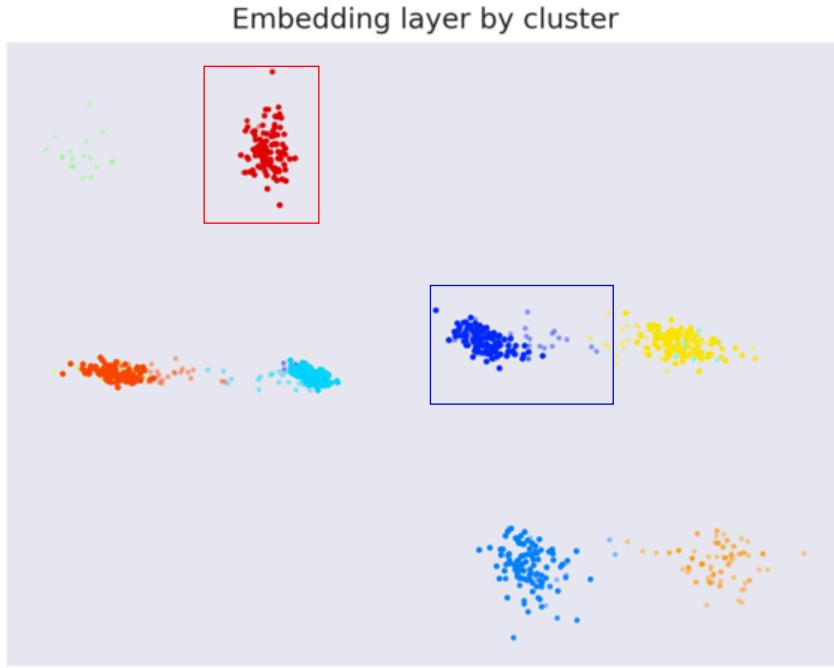
→ Imputation



● Patient 1 ● Patient 2

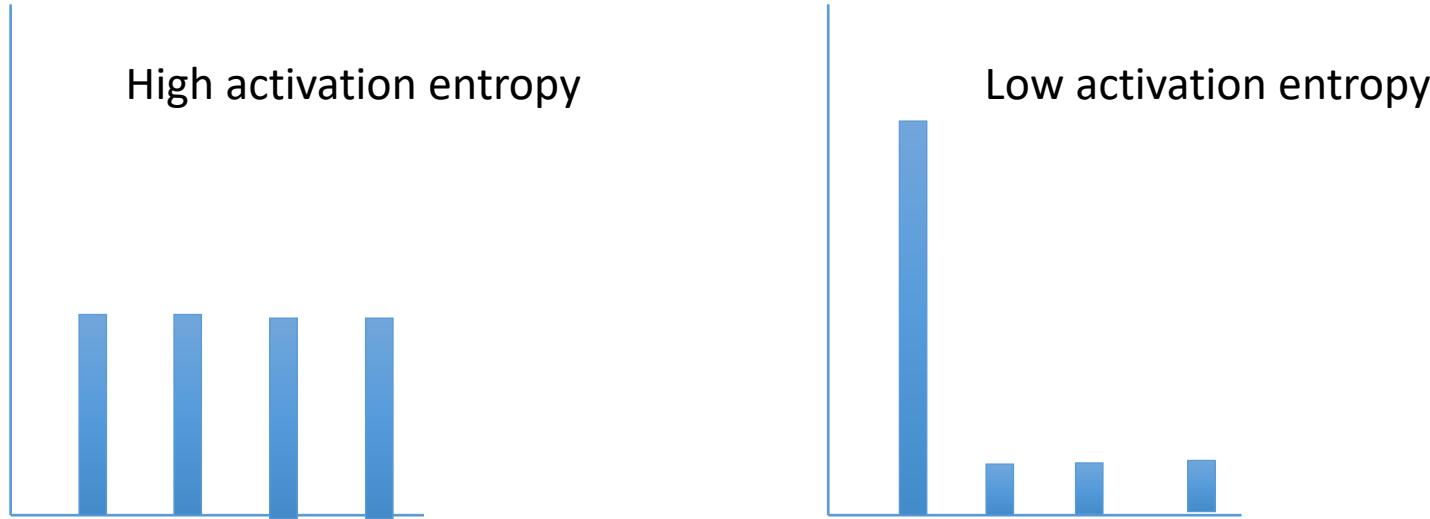
A**B****C****D**

Representation for Clustering



We want: codes for subspaces
Binary activation, good spacing

Clustering: Information Dimension Regularization

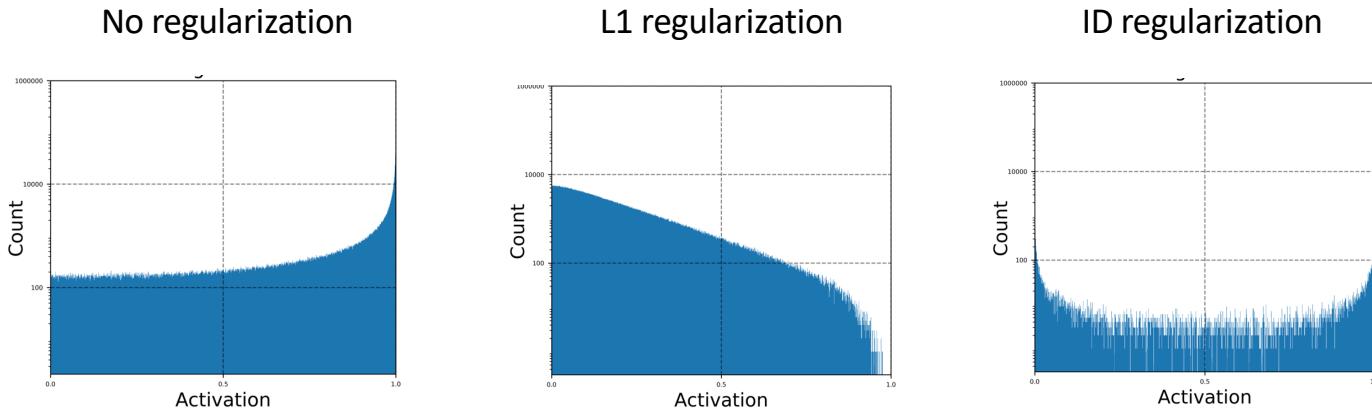


$$-\sum a_i \log(a_i)$$

Penalty:

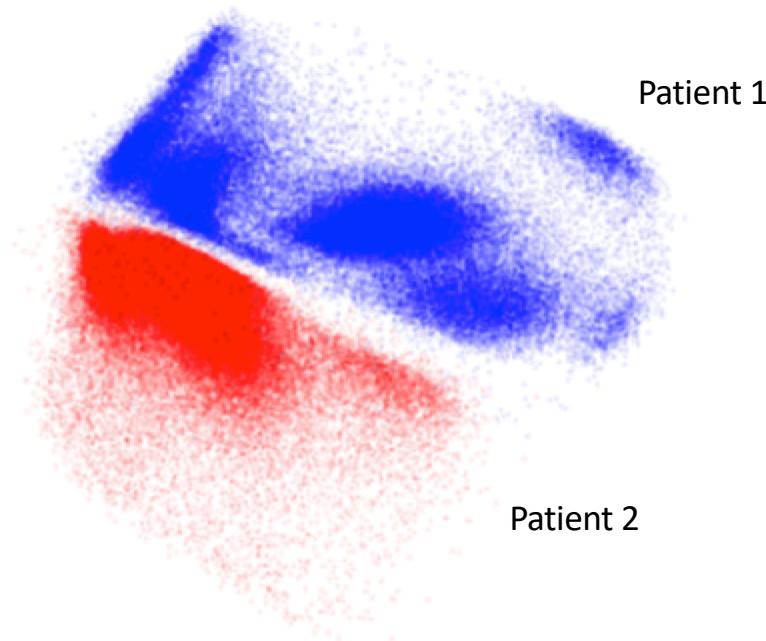
$$a_i = \frac{A_i}{\sum A_i}$$

Information Dimension (ID) Regularization

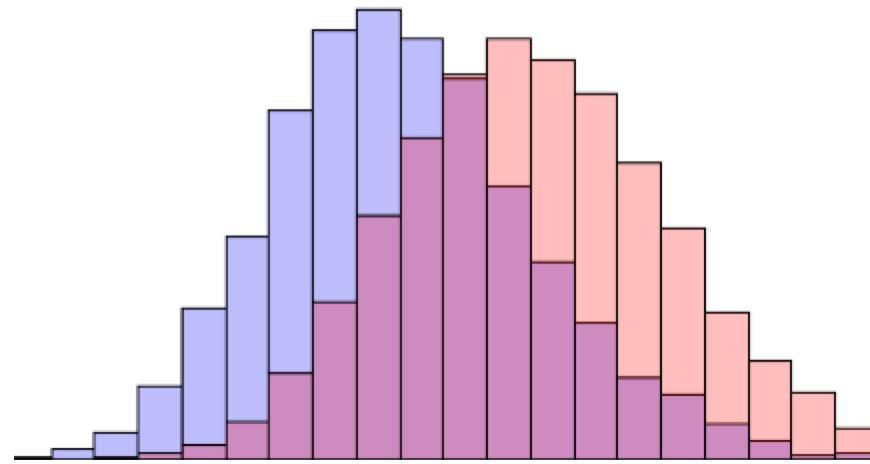


ID regularization enforces a binary-like code

Batch Effects: Manifold Alignment



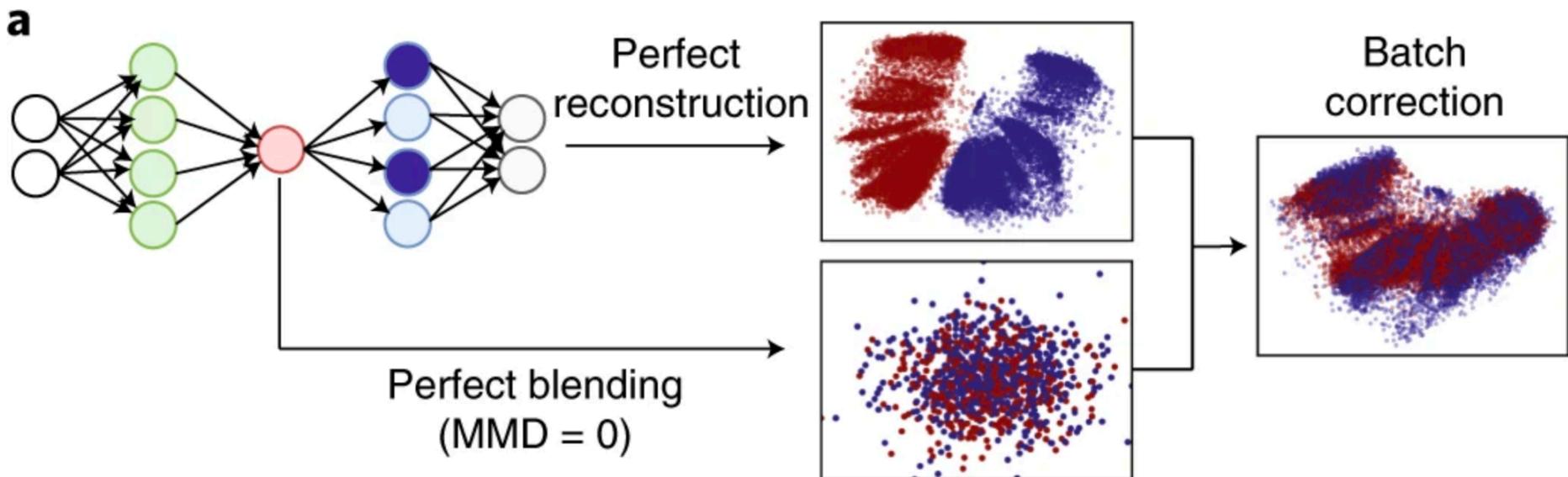
Mean Maximal Discrepancy



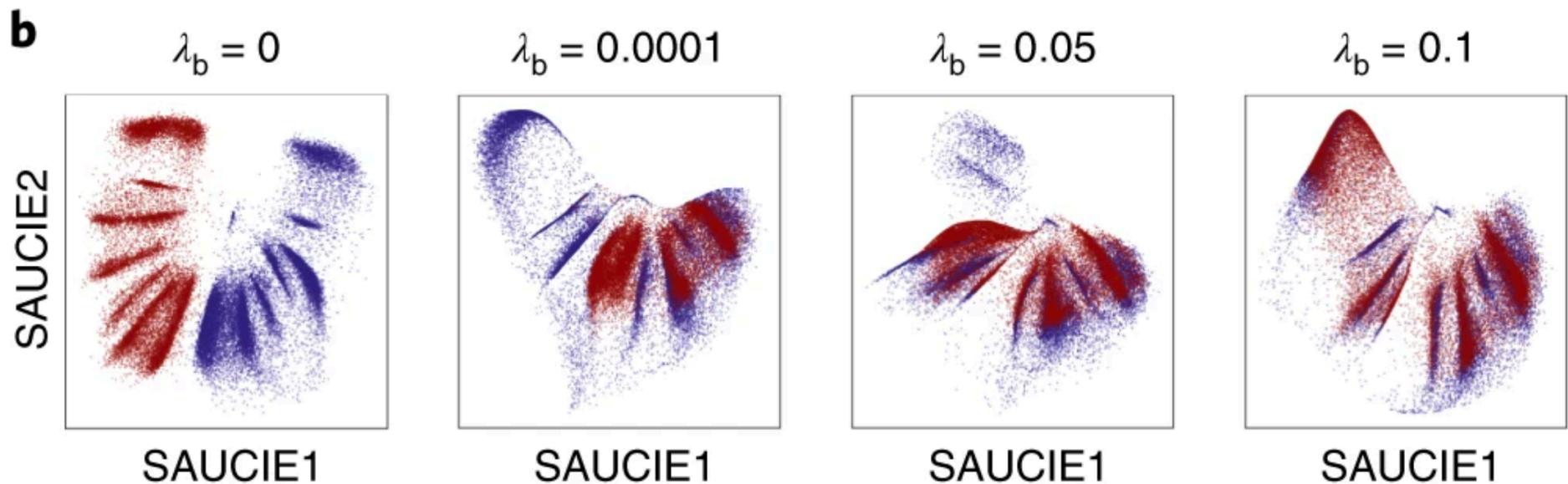
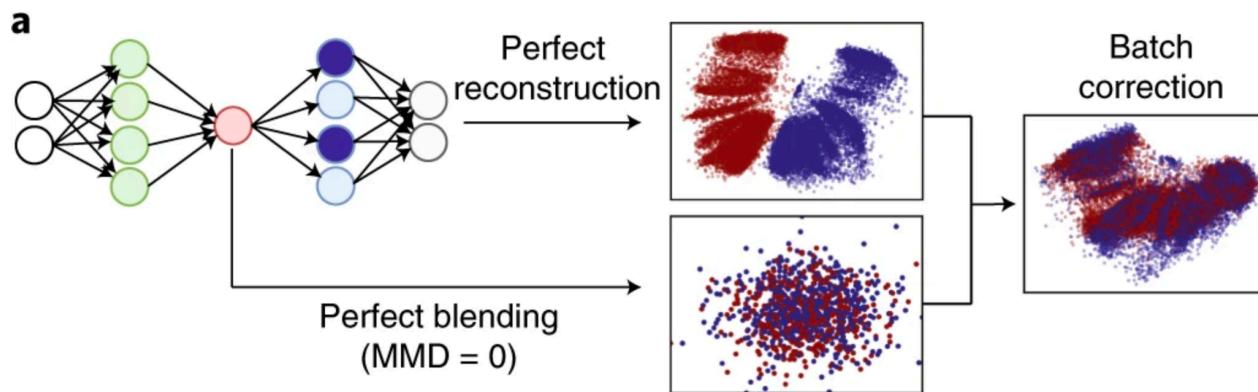
$$MMD(p, q) = \frac{1}{m^2} \sum_{i,j \in m} K(p_i, p_j) - \frac{2}{mn} \cdot \sum_{i,j} K(p_i, q_j) + \frac{1}{n^2} \sum_{i,j \in n} K(q_i, q_j)$$

[Dziugaite, Roy, Ghahramani, UAI 2015]

MMD Penalty on middle layer is opposed by reconstruction



MMD Penalty on middle layer is opposed by reconstruction



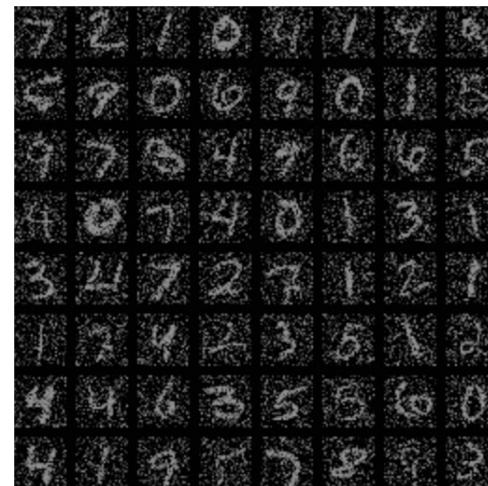
Reconstruction as Denoising

Original

7	2	1	0	4	1	4	9
6	9	0	6	9	0	1	5
9	7	3	4	9	6	6	5
4	0	7	4	0	1	3	1
3	0	7	2	7	1	2	1
1	7	4	2	3	5	1	2
4	4	6	3	5	5	6	0
4	1	9	5	7	8	9	3

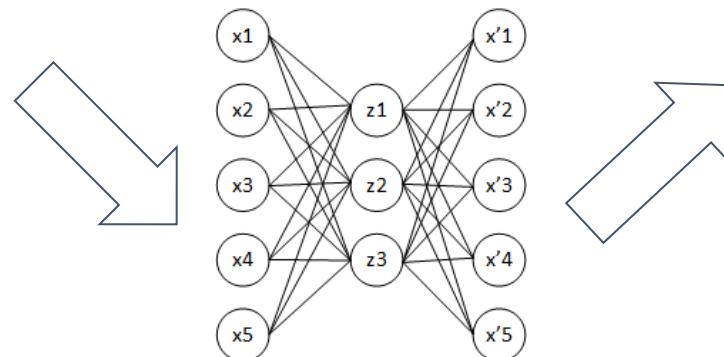


Dropped-out data



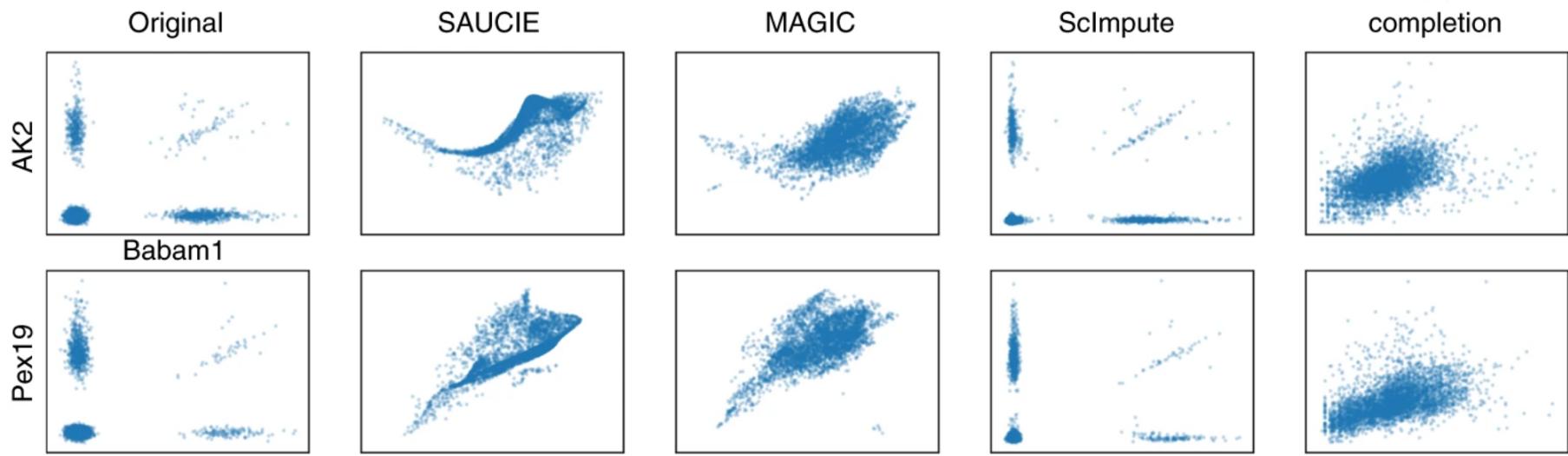
Reconstructed

7	2	1	0	4	1	4	9
6	9	0	6	9	0	1	5
9	7	3	4	9	6	6	5
4	0	7	4	0	1	3	1
3	0	7	2	7	1	2	1
1	7	4	2	3	5	1	2
4	4	6	3	5	5	6	0
4	1	9	5	7	8	9	3

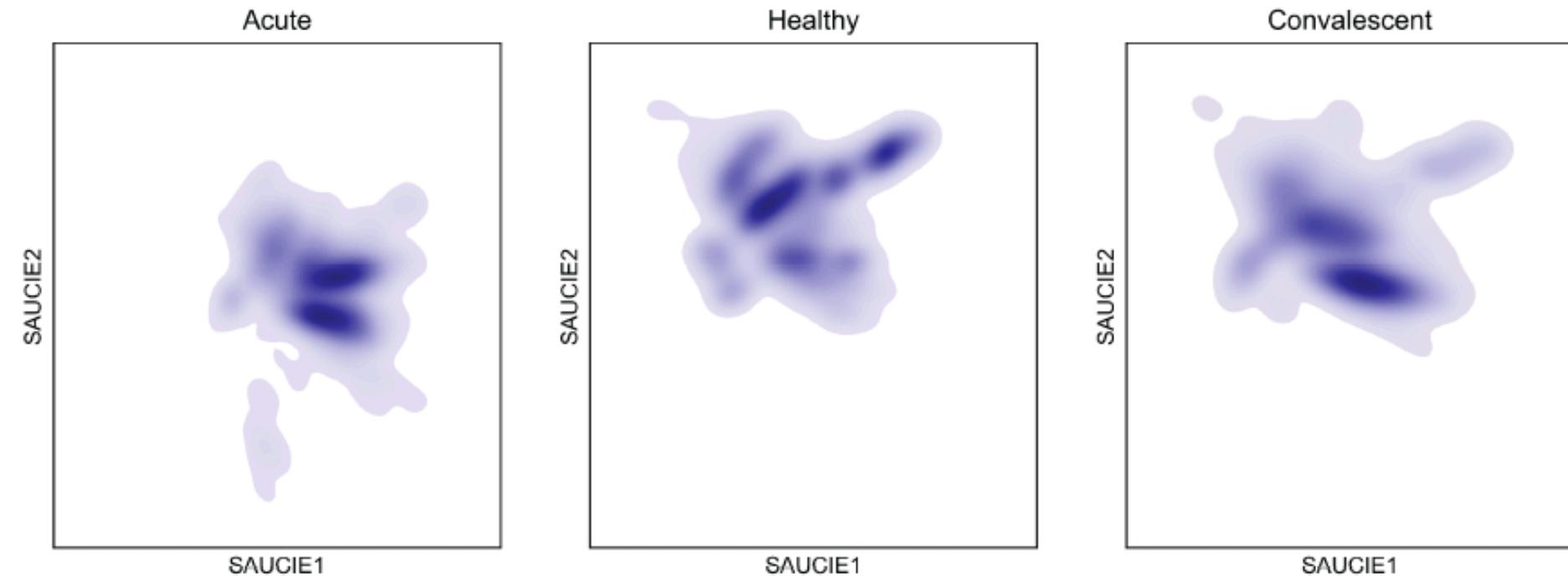


SAUCIE data denoising

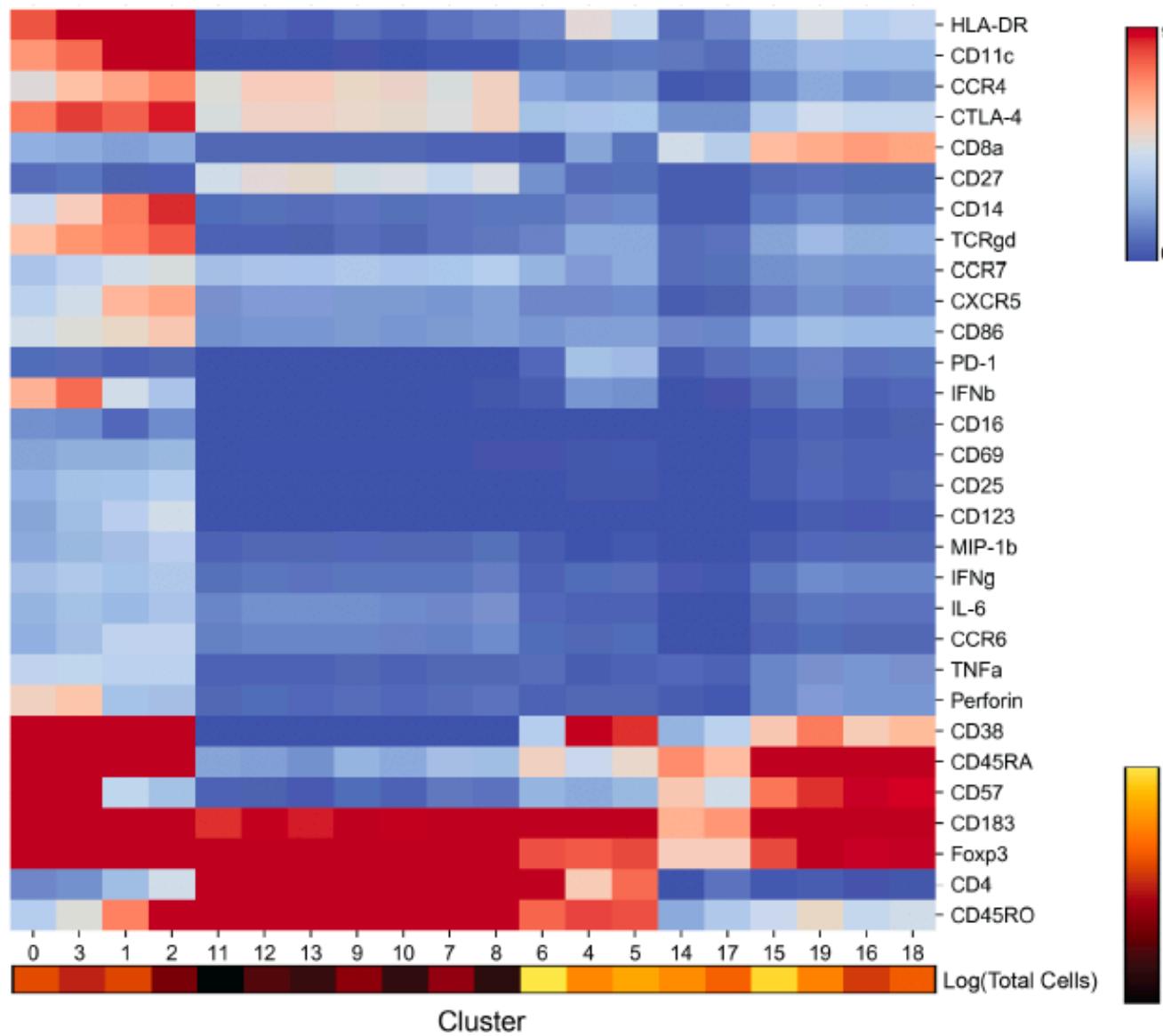
c



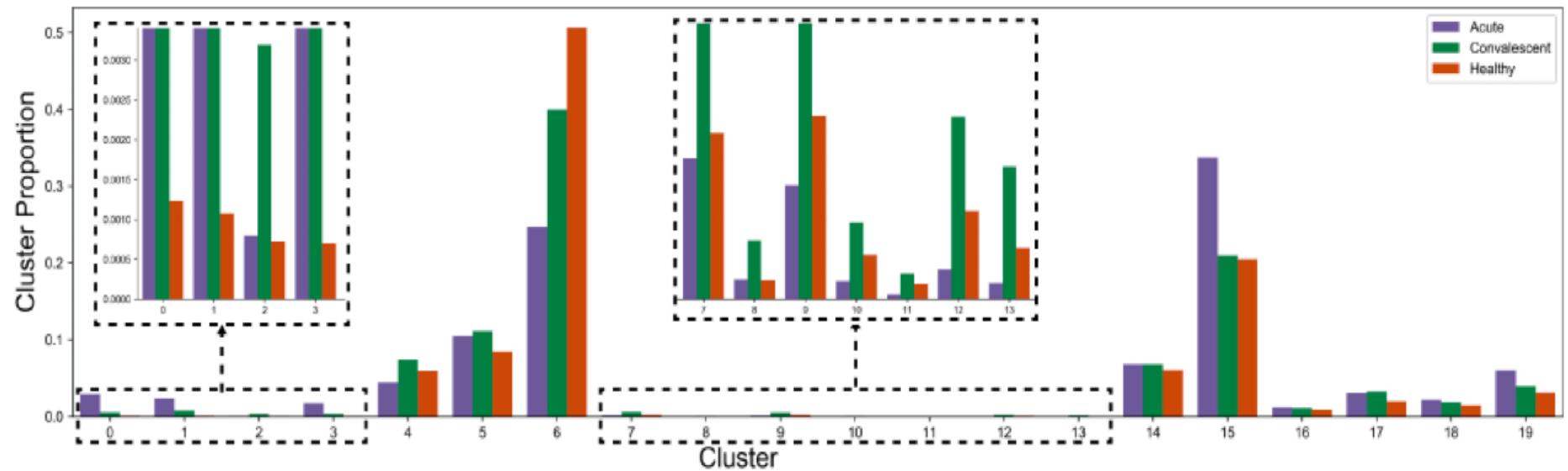
Cellular Manifolds of Dengue Patients



Cell Clusters: 180 Samples Combined

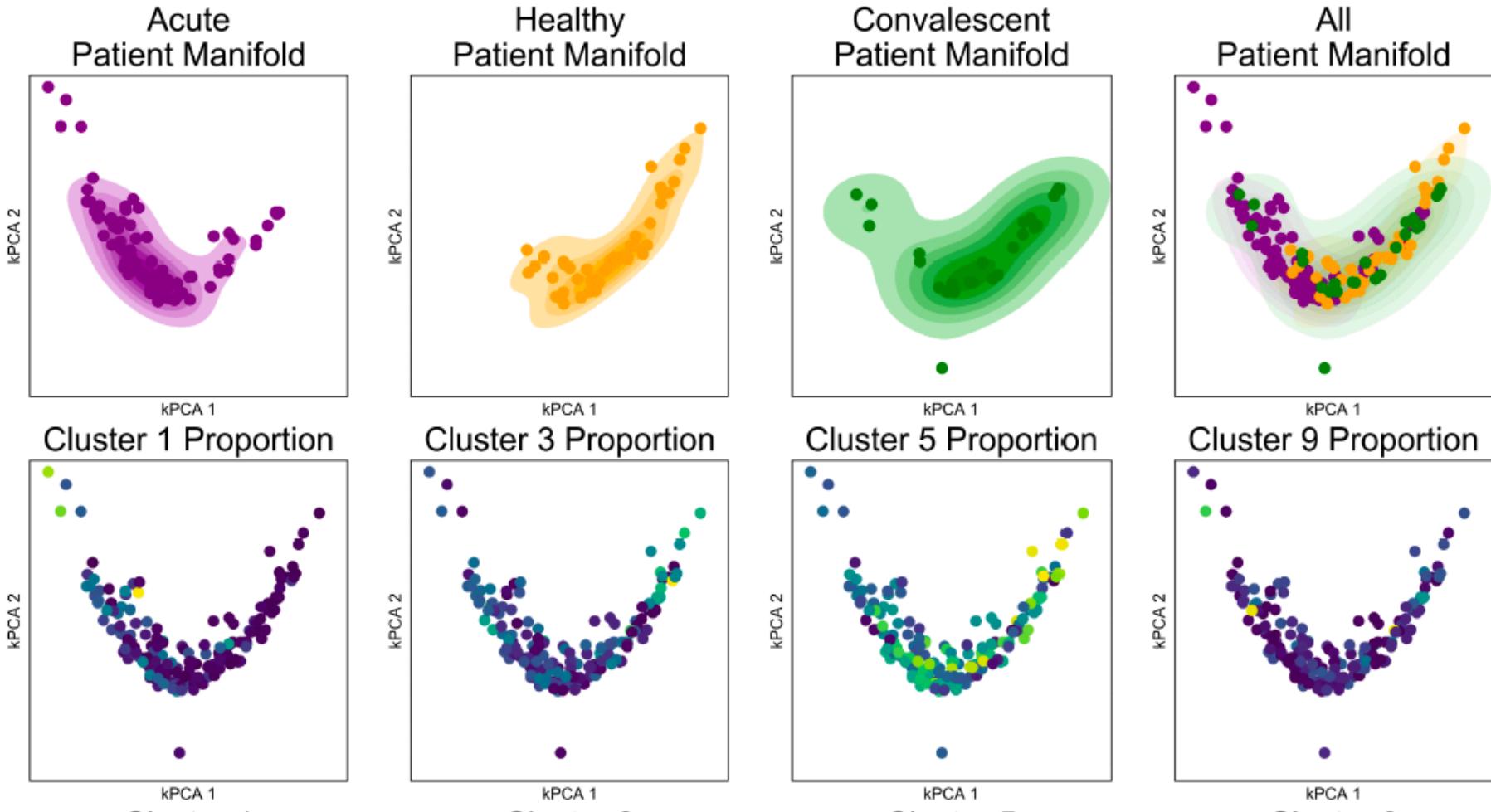


Patient Cluster Signatures

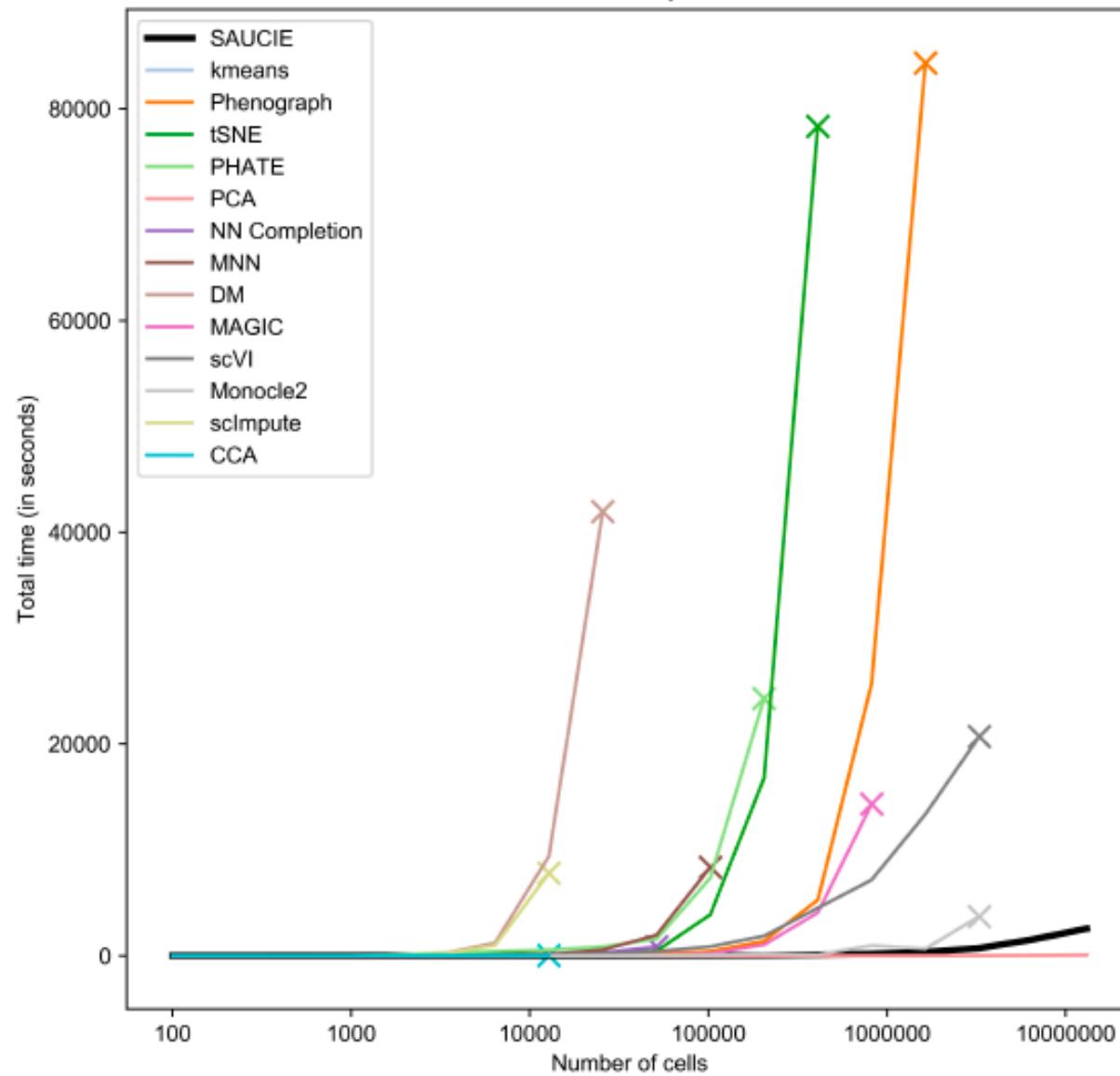


MMD distance can be derived between patient cluster proportions and embedded

Patient manifolds show coherent organization



Runtime Comparison

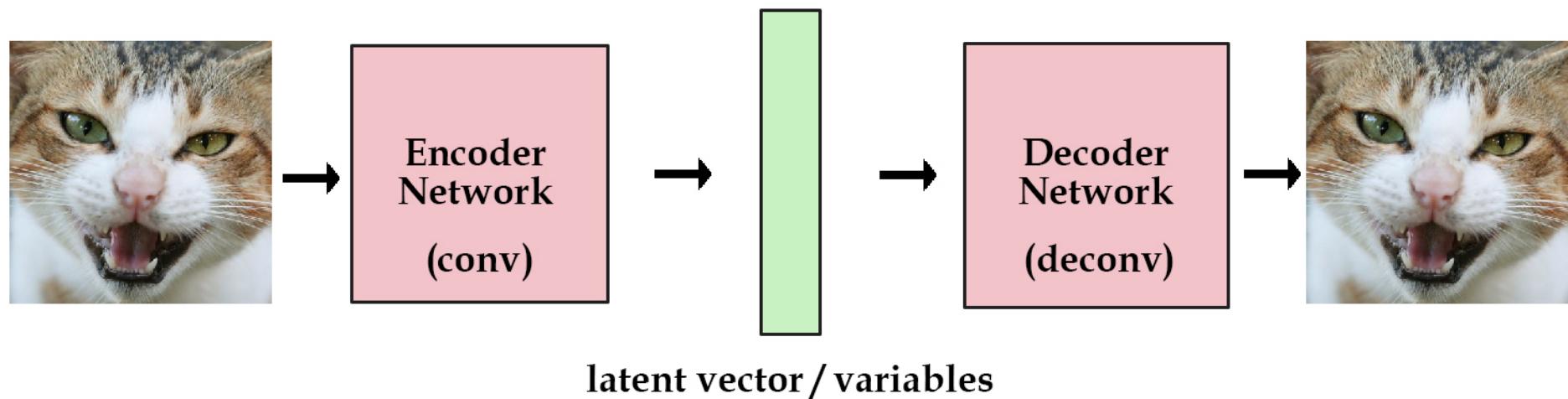


Generative Models

Generating samples from the data

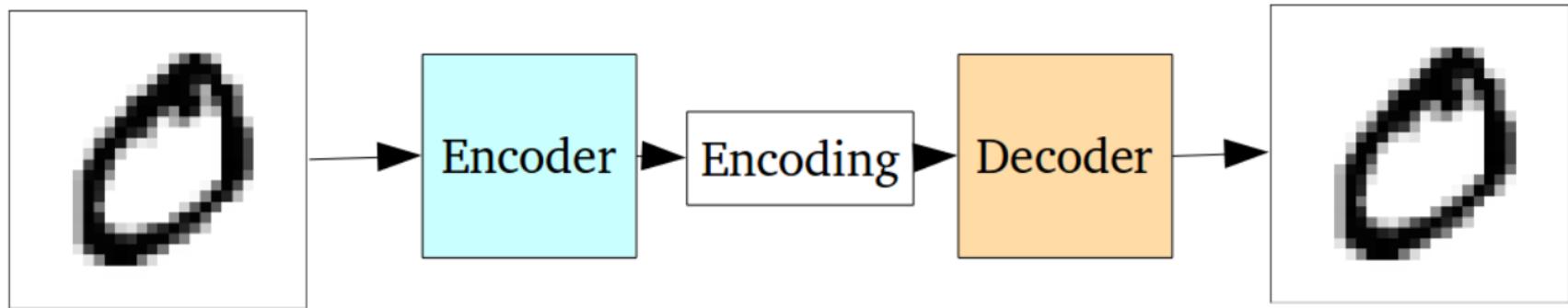
- Key use of neural networks is generating samples from data
- One idea is to use the decoder without the encoder, sample from z and then have it decode to a datapoint
- For single cell data, this would mean generating synthetic cells from a distribution

Variational Autoencoders (VAEs)



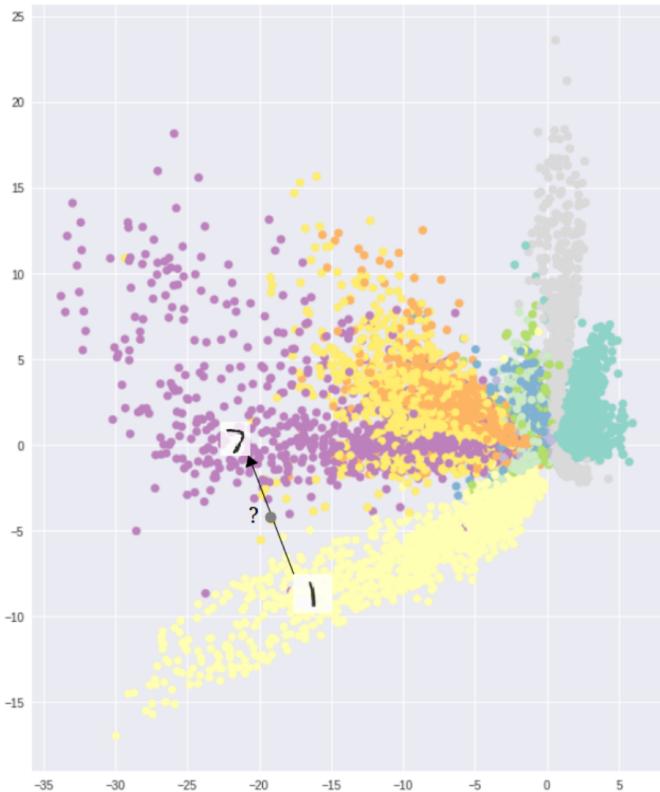
- A type of neural network whose latent space is designed for sampling

Why not autoencoders?



- Optimized with Reconstruction Loss
- Not explicitly penalized for Generative Purposes

Latent Space Not Optimal for Generation

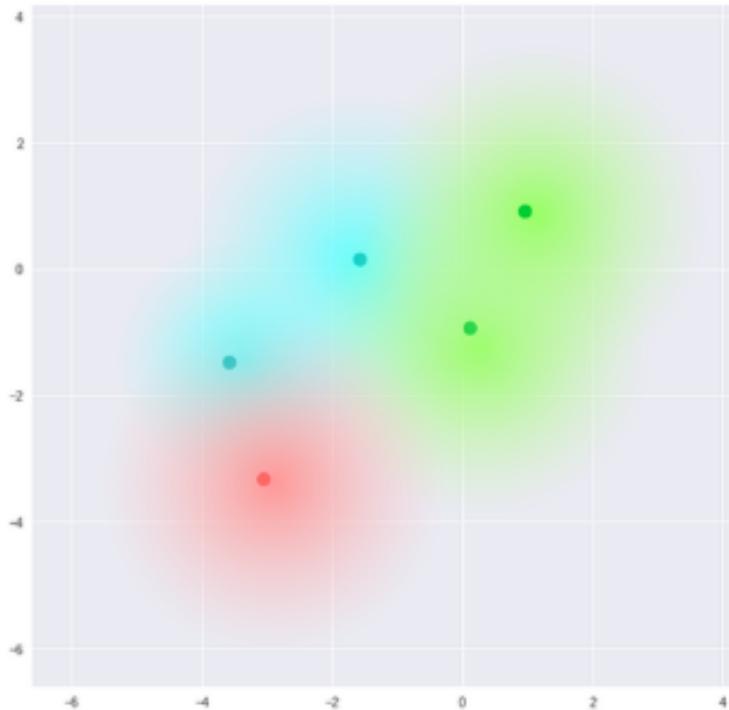


Formation of clusters helps decoding

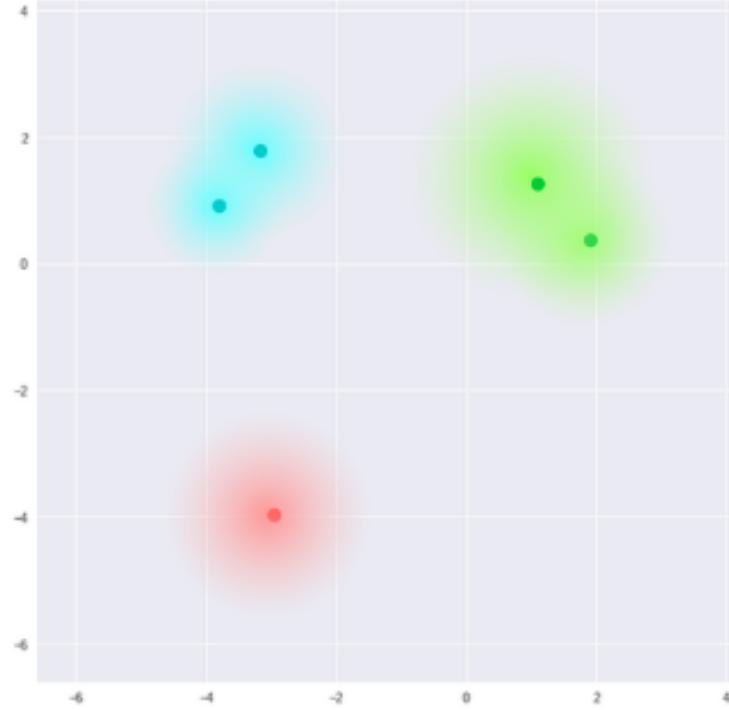
Does not help generation

Latent space cannot be interpolated

Encouraging continuous latent spaces



What we require



What we may inadvertently end up with

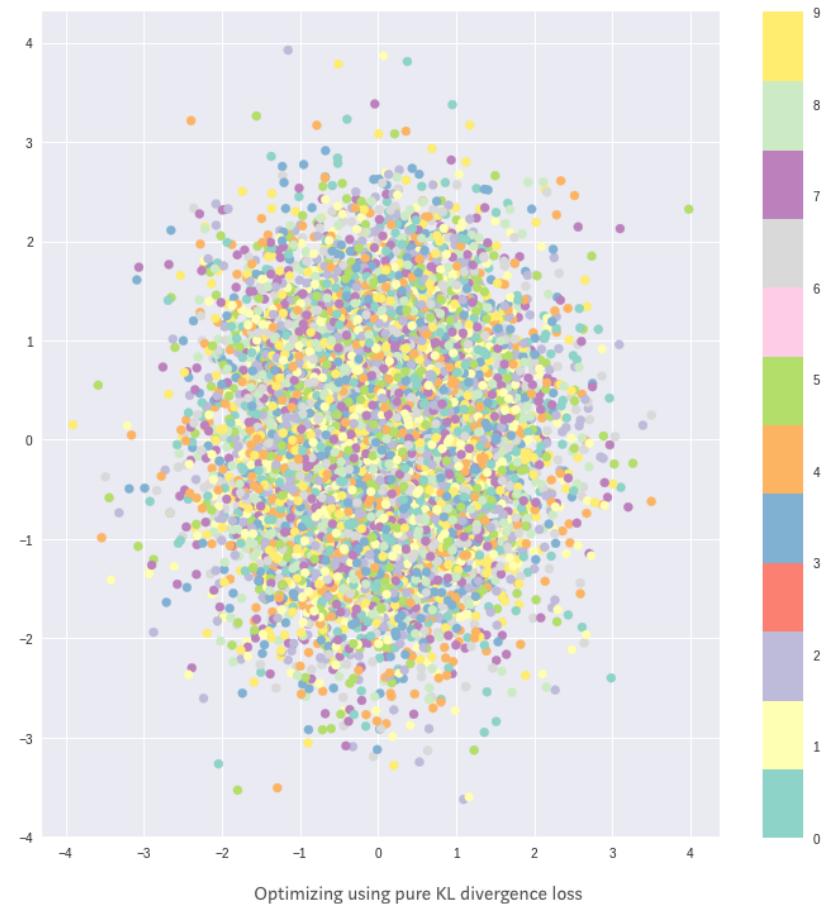
Solution: Penalize the Gaussian balls if they don't overlap!

KL Divergence Penalty

$$\text{KL}(q_\theta(z | x_i) || p(z))$$

Penalizing distributions
In the latent space from being
too far from a standard normal

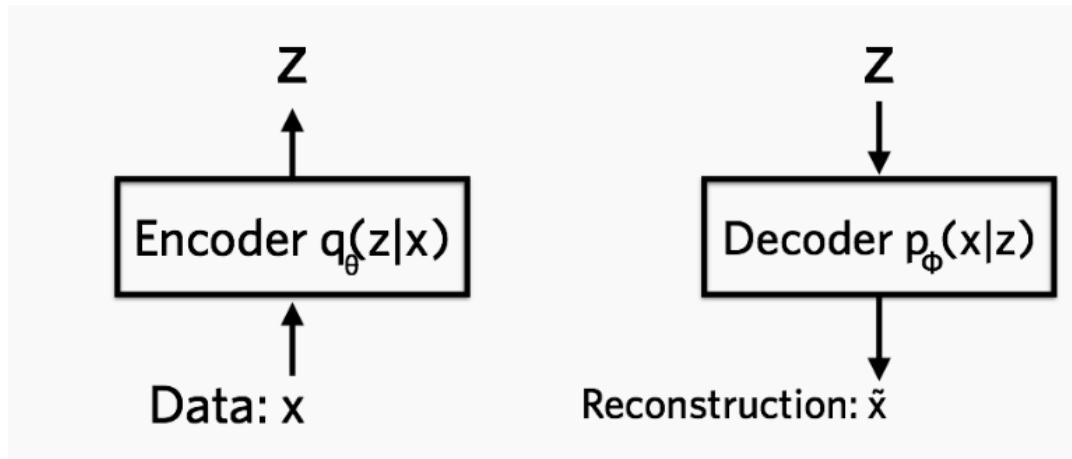
$$p(z) = \text{Normal}(0, 1).$$



Encourages every ball to be the same!

Loss Function of VAE

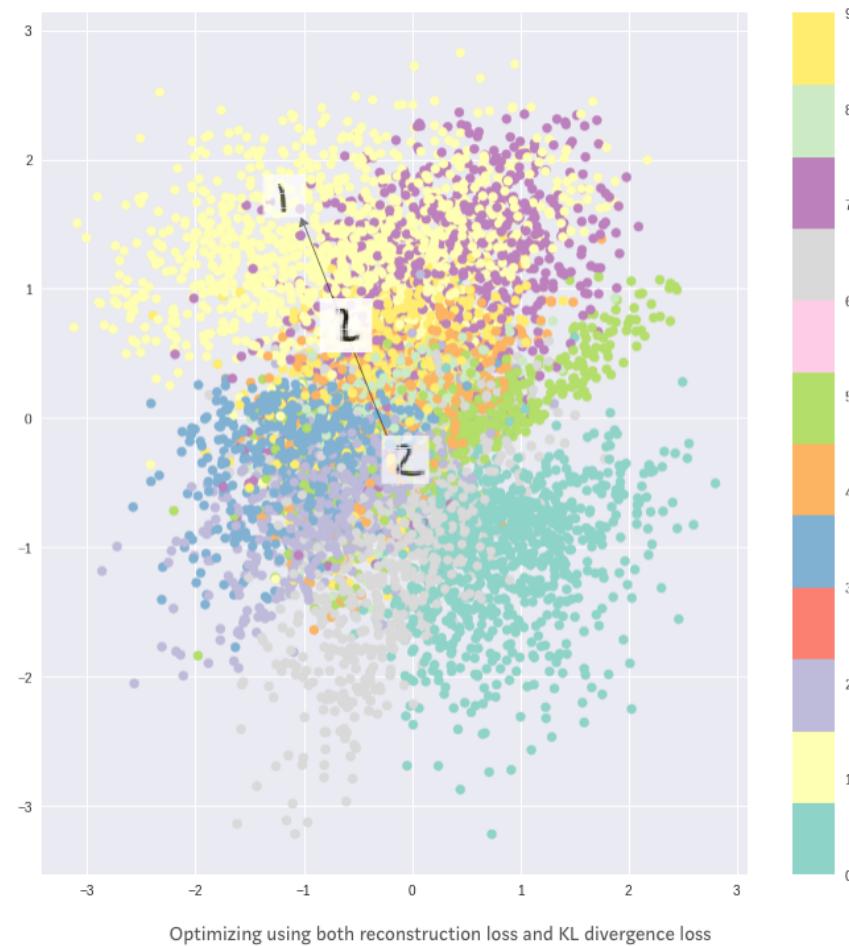
$$l_i(\theta, \phi) = -\mathbb{E}_{z \sim q_\theta(z|x_i)} [\log p_\phi(x_i | z)] + \mathbb{KL}(q_\theta(z | x_i) || p(z))$$



Reconstruction penalty encourages separation

$$-\mathbb{E}_{z \sim q_\theta(z|x_i)} [\log p_\phi(x_i | z)]$$

KL+Reconstruction Loss



Variational Autoencoders (VAEs)



1st Epoch



9th Epoch



Original

VAEs in Single Cell Analysis.

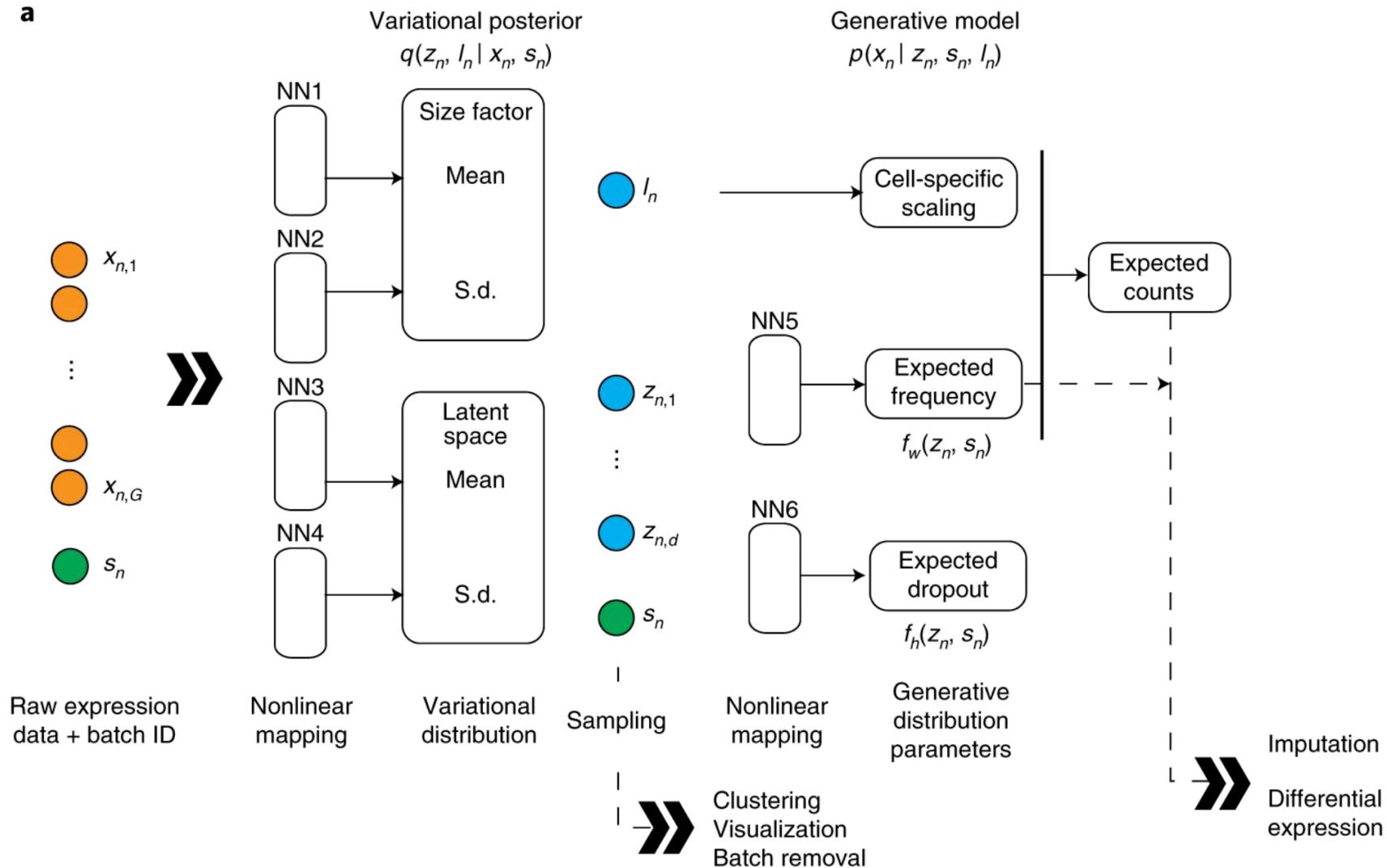
scVI

Deep generative modeling for single-cell transcriptomics

Romain Lopez, Jeffrey Regier, Michael B. Cole, Michael I. Jordan & Nir Yosef 

scVI

a



scVI uses latent space of VAE as a dimensionality reduced encoding fed to other algorithms

Probabilistic assumptions

$$z_n \sim \text{Normal}(0, I)$$

$$\ell_n \sim \text{log normal}(\ell_\mu, \ell_\sigma^2)$$

$$\rho_n = f_w(z_n, s_n)$$

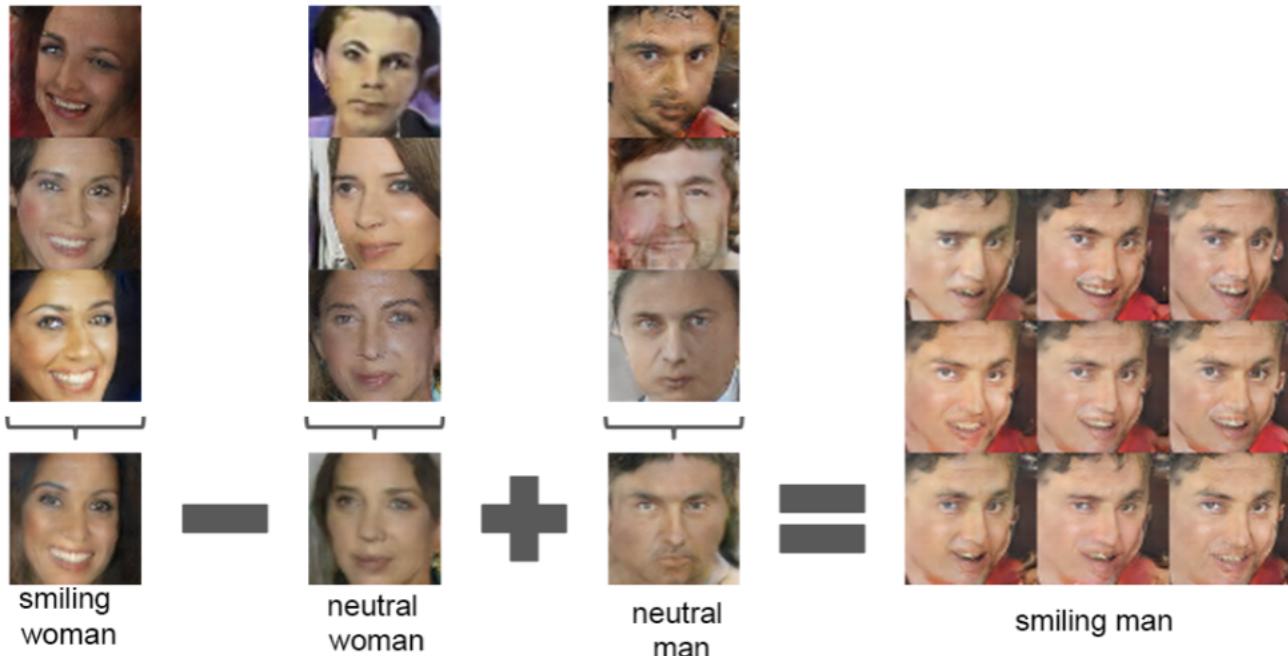
$$w_{ng} \sim \text{Gamma}(\rho_n^g, \theta)$$

$$y_{ng} \sim \text{Poisson}(\ell_n w_{ng})$$

$$h_{ng} \sim \text{Bernoulli}(f_h^g(z_n, s_n))$$

$$x_{ng} = \begin{cases} y_{ng} & \text{if } h_{ng} = 0 \\ 0 & \text{otherwise} \end{cases}$$

Latent Space Arithmetic



scGEN

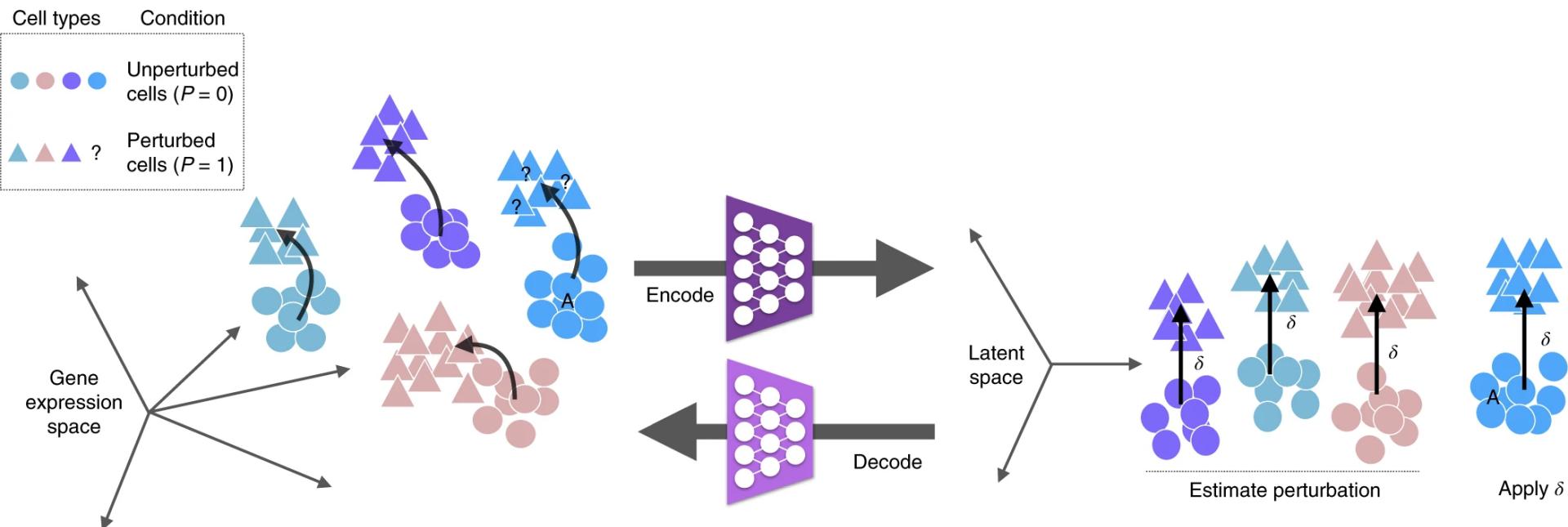
scGen predicts single-cell perturbation responses

Mohammad Lotfollahi, F. Alexander Wolf  & Fabian J. Theis 

Nature Methods **16**, 715–721(2019) | [Cite this article](#)

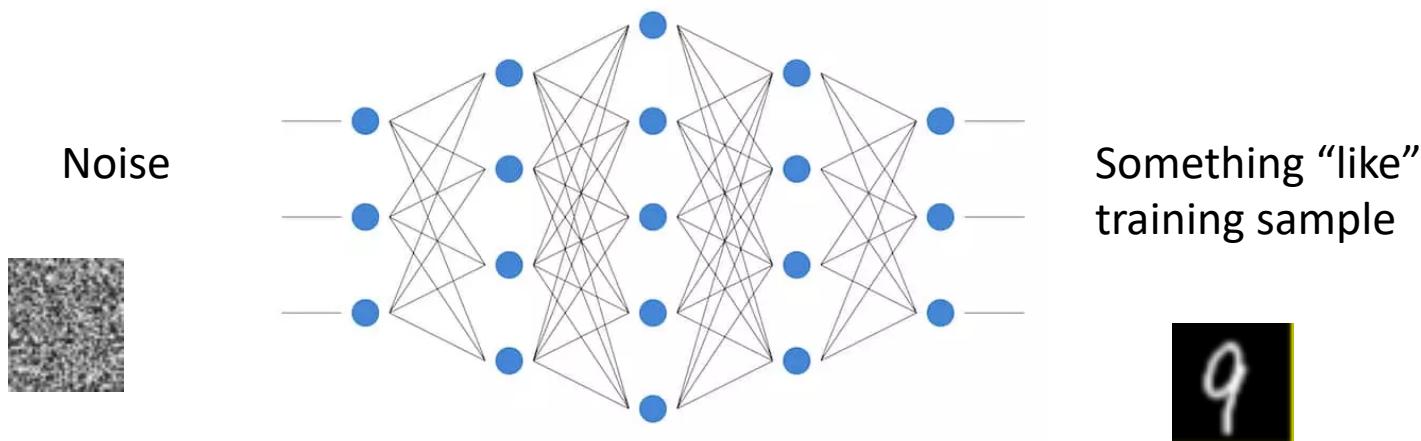
scGEN

- Generates single cell response by doing latent space arithmetic



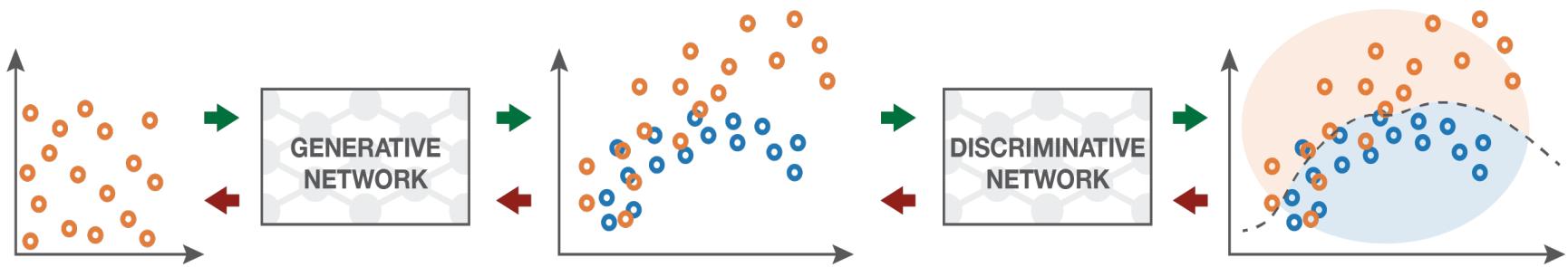
Perturbations are modelled as shifts

Generative Adversarial Networks



GAN functionality

Forward propagation (generation and classification) Backward propagation (adversarial training)



Input random variables.

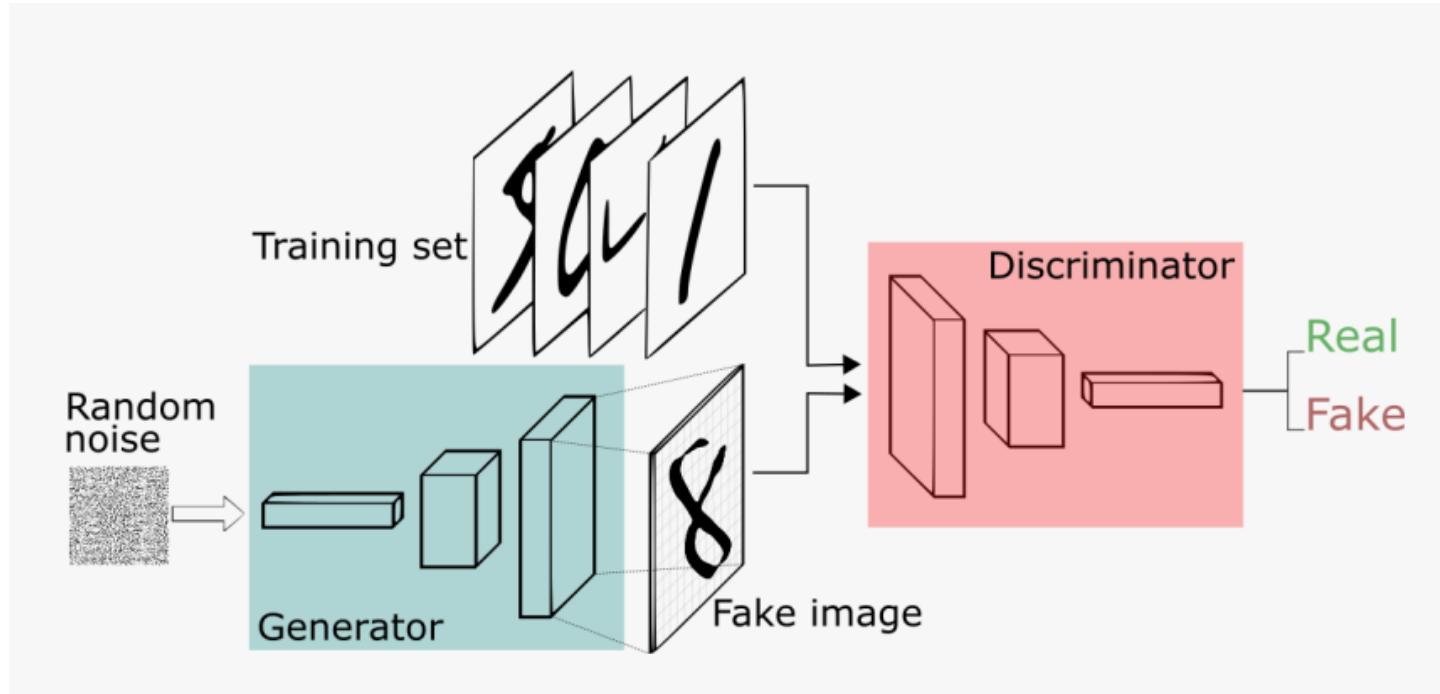
The generative network is trained to **maximise** the final classification error.

The **generated distribution** and the **true distribution** are not compared directly.

The discriminative network is trained to **minimise** the final classification error.

The classification error is the basis metric for the training of both networks.

GANs

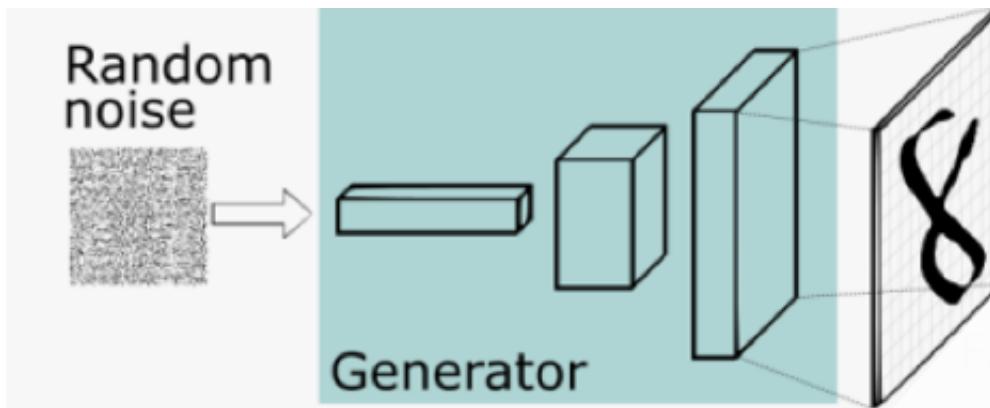


- Sample from an easy noise distribution (uniform, Gaussian)
- One network learns a mapping $G(\mathbf{Z})$
- Second network tries to distinguish true examples of \mathbf{X} from “fake” samples $G(\mathbf{Z})$

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

Only use generator after training



- Generator network takes noise and transmutes them into samples from the data distribution
- Implicitly learns high dimensional distributions

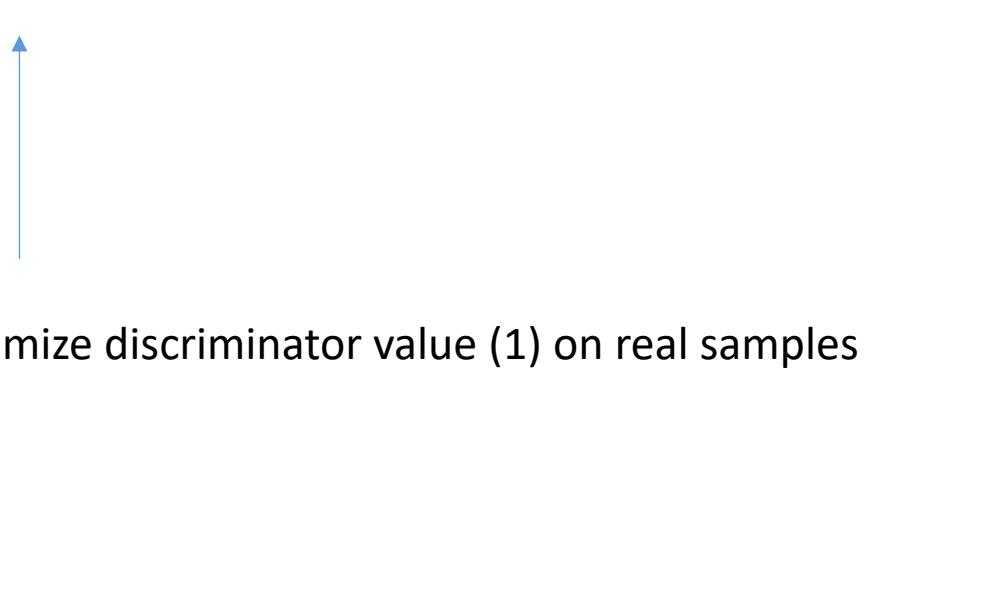
Generated Examples

7	3	9	3	9	9
1	1	0	6	0	0
0	1	9	1	2	2
6	3	2	0	8	8



GAN Loss function

$$E_x[\log(D(x))] + E_z[\log(1 - D(G(z)))]$$



Generator wants the discriminator to do badly!

GAN Loss function

$$\min_G \max_D E_x[\log(D(x))] + E_z[\log(1 - D(G(z)))]$$

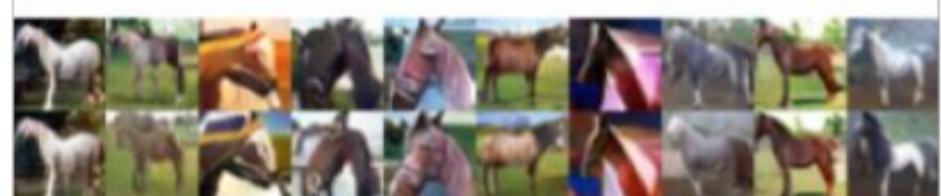
This is the minimax game that the two adversaries play

Mode Collapse

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



evidence of lack of diversity



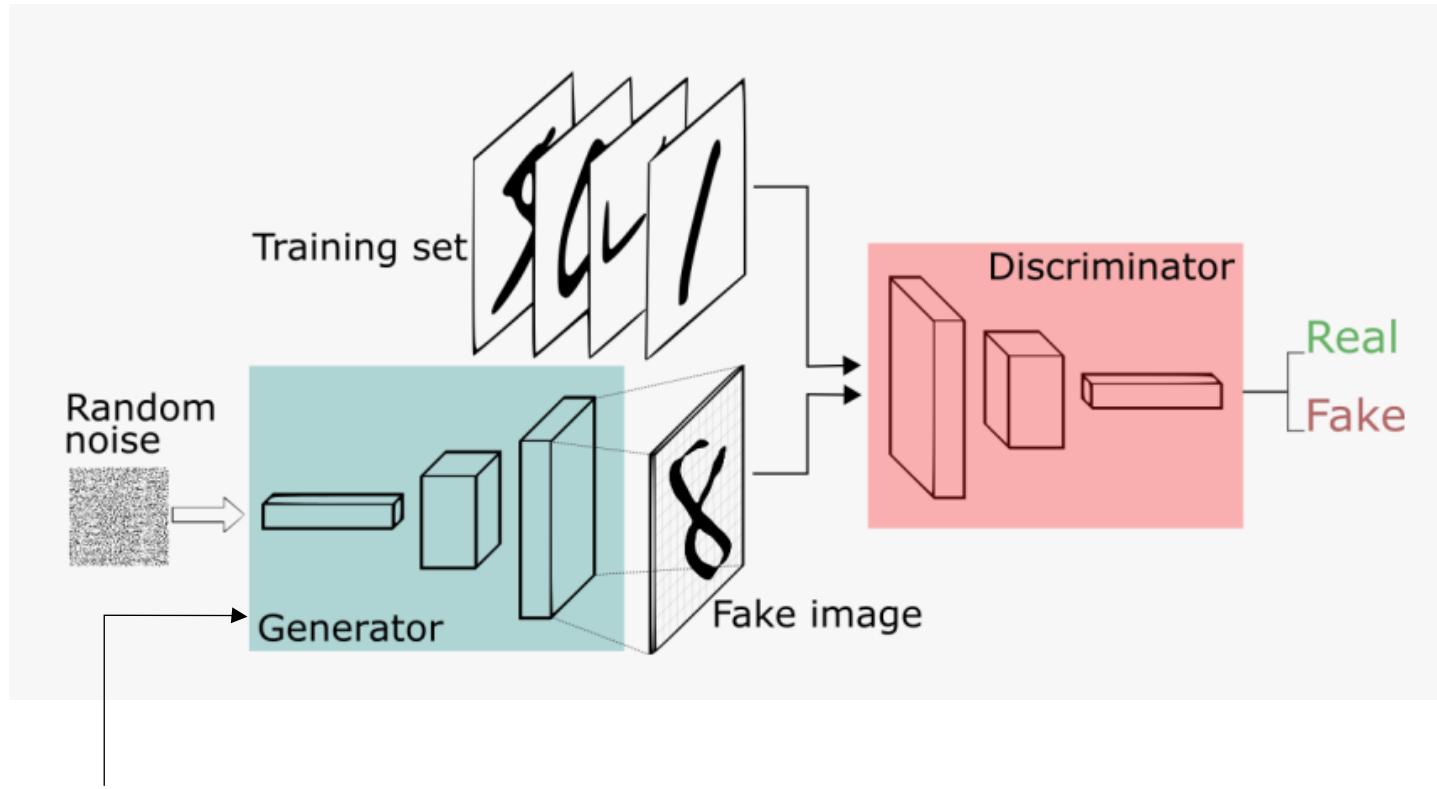
GANs can be hard to train

- Instability during training
- Mode collapse
 - Take the whole batch as a training sample
 - Penalize for a property of the distribution (mean, variance)

Conditional GANs

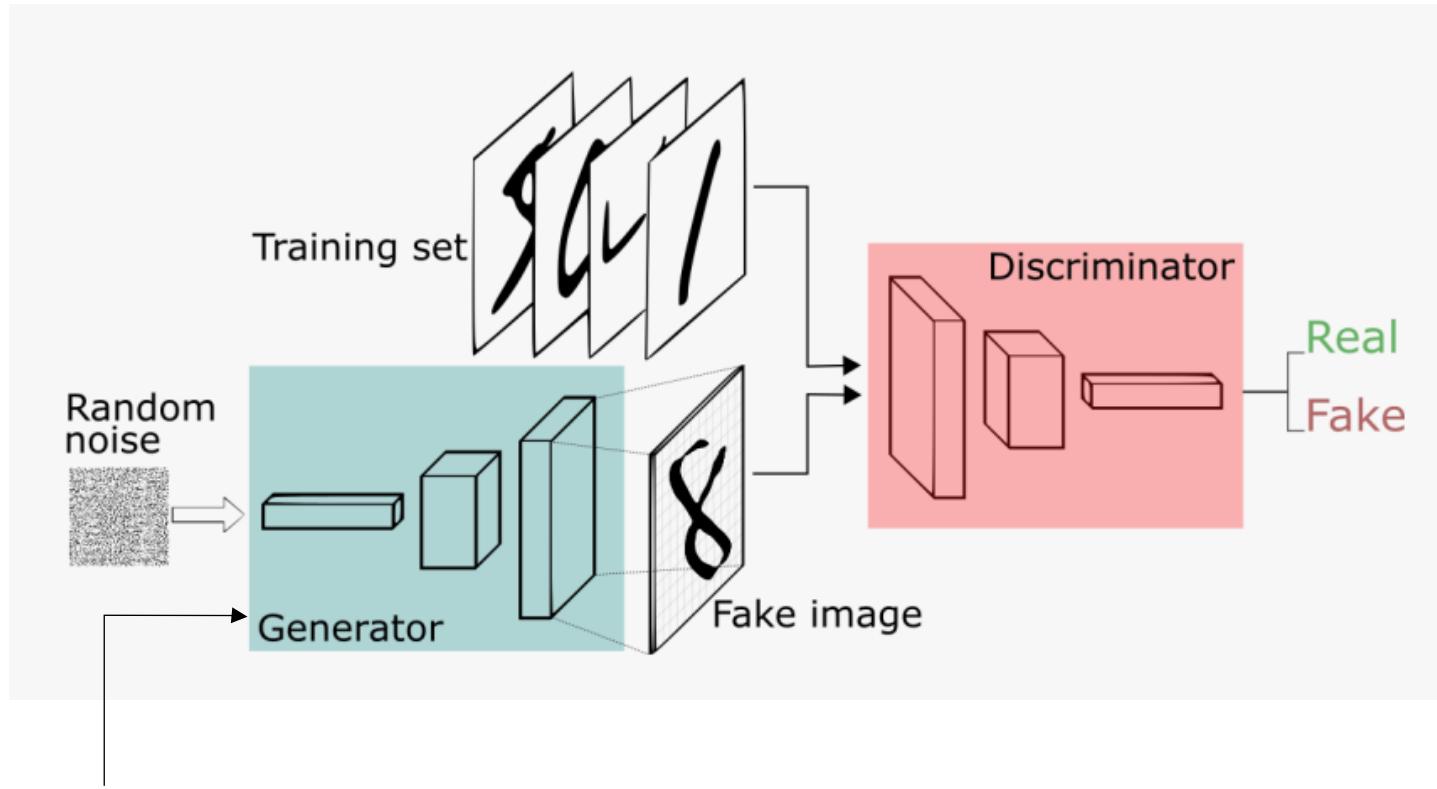
- GANs take noise z and turn it into our data x
- Great! But what if we want to control it?
 - “It generated *another* image of a 3? I wanted a 7 this time!”
- Conditional GANs:
 - If we have labels, we want to teach it to generate not just any x , but an x of a particular label when given some noise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

Conditional GANs



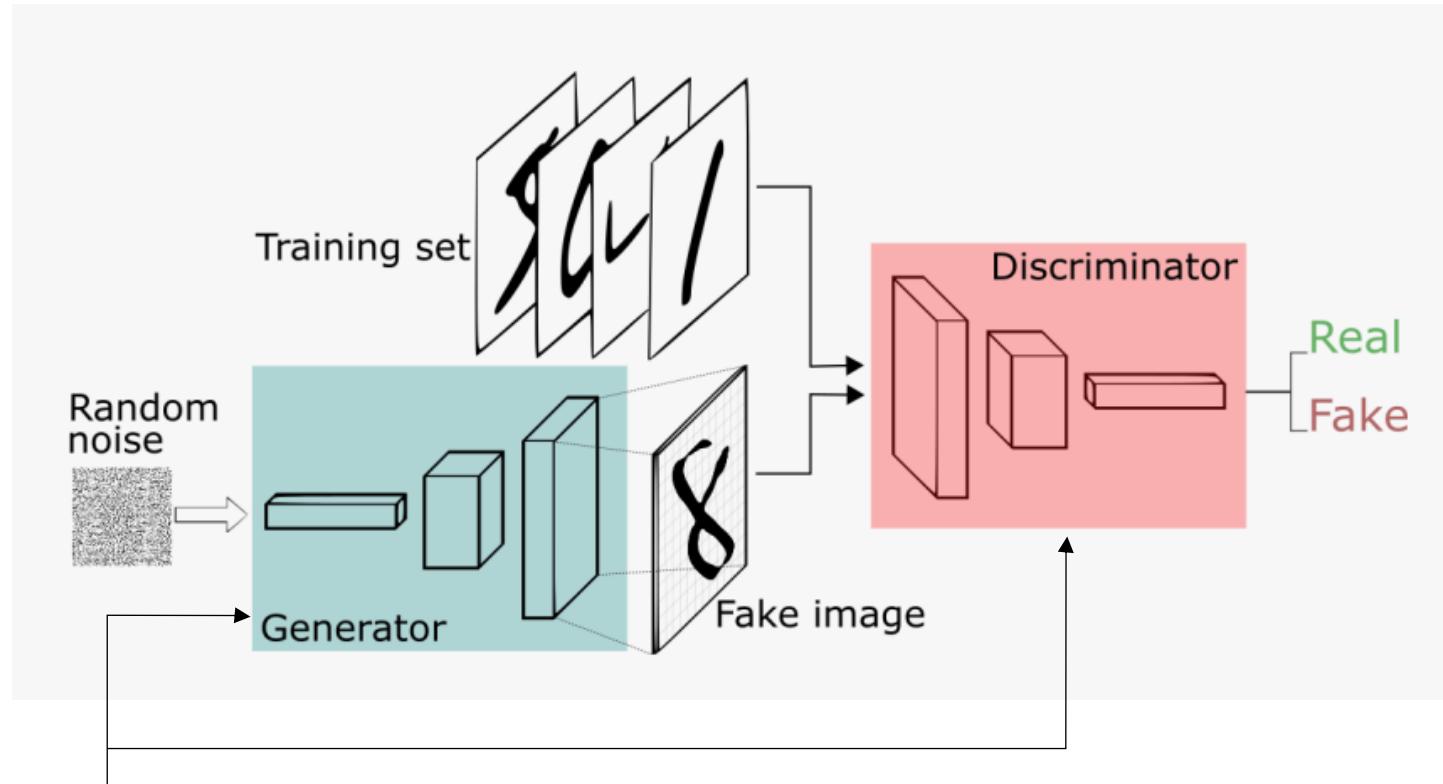
Condition
“8”

Conditional GANs



Condition
“8”

Conditional GANs



Condition
“8”

Both generator and discriminator look at the condition

cSCGAN

Realistic in silico generation and augmentation of single-cell RNA-seq data using generative adversarial networks

Mohamed Marouf, Pierre Machart, Vikas Bansal, Christoph Kilian, Daniel S. Magruder, Christian F. Krebs & Stefan Bonn 

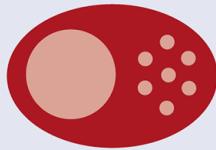
Nature Communications **11**, Article number: 166 (2020) | [Cite this article](#)

cscGAN

- Maps from noise to single cell measurements
- Conditions correspond to cell types in PBMCs



CD4⁺
T cells
25–60%



CD8⁺
T cells
5–30%



CD19⁺
B cells
5–10%



CD56⁺ CD3⁻
NK cells
10–30%



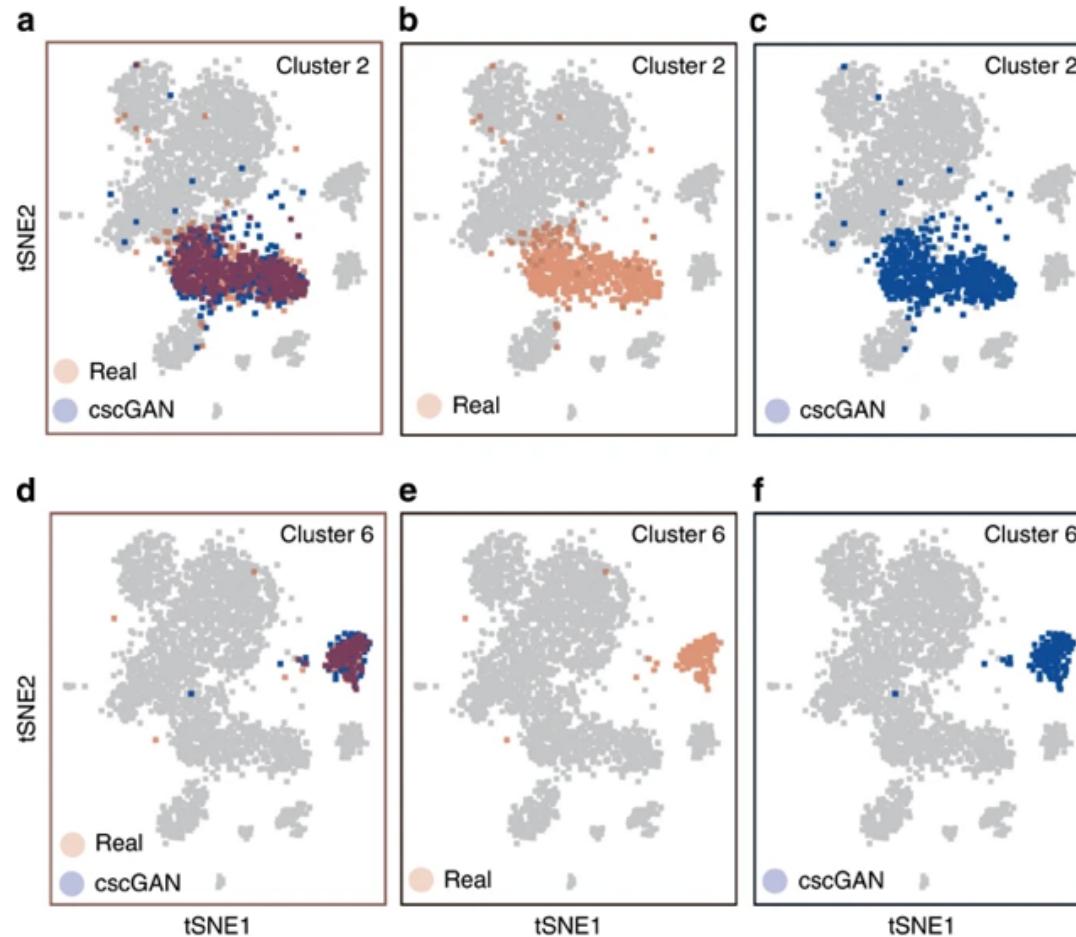
CD14⁺
monocytes
5–10%



Dendritic
cells
1–2%

Some cell types are naturally rare, may want to augment measurements with “in silico” cell generation

cscGAN



Generated cells visualized on tSNE

Domain Adaptation and Cycle GANs

Domain Adaptation

- GANs and adversarial AEs sample from one domain
- What if we have two domains, A and B?
 - A=single cell RNAseq, B= single cell ATACseq
 - A=healthy, B=diseased
 - A=control, B=perturbed
 - A=batch1, B=batch 2?



Unsupervised matching

- Can we learn to turn a point in A into a point that looks like its from B?
- We don't have the “answer” or corresponding measurement in the other modality



- CycleGAN frameworks were set up to do this

DiscoGAN: Motivation

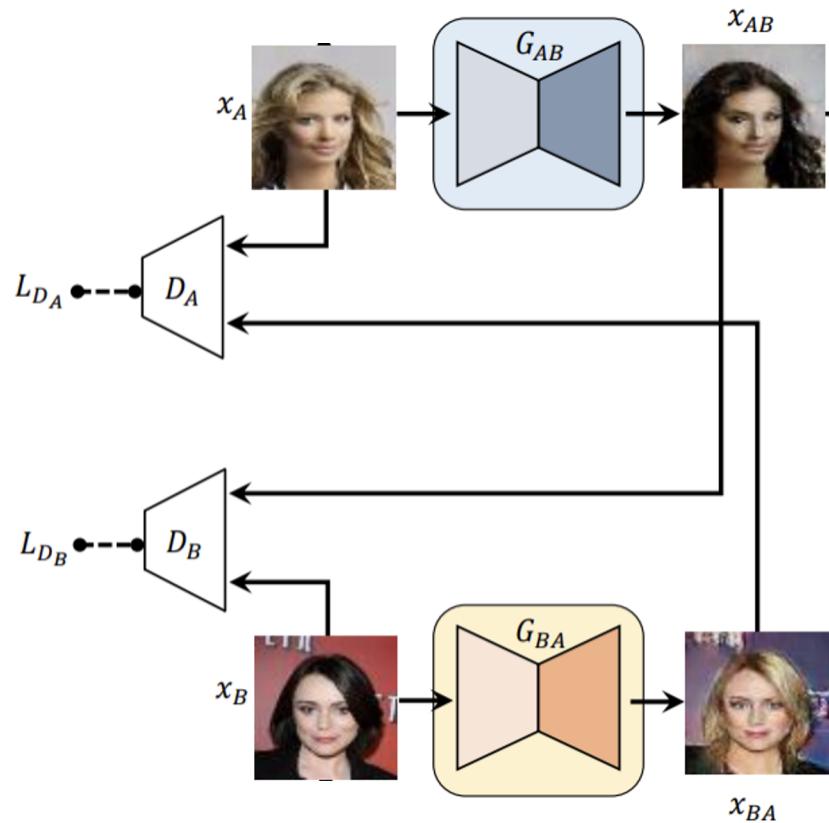
- Domain A: blond hair



- Domain B: black hair



Previous approach

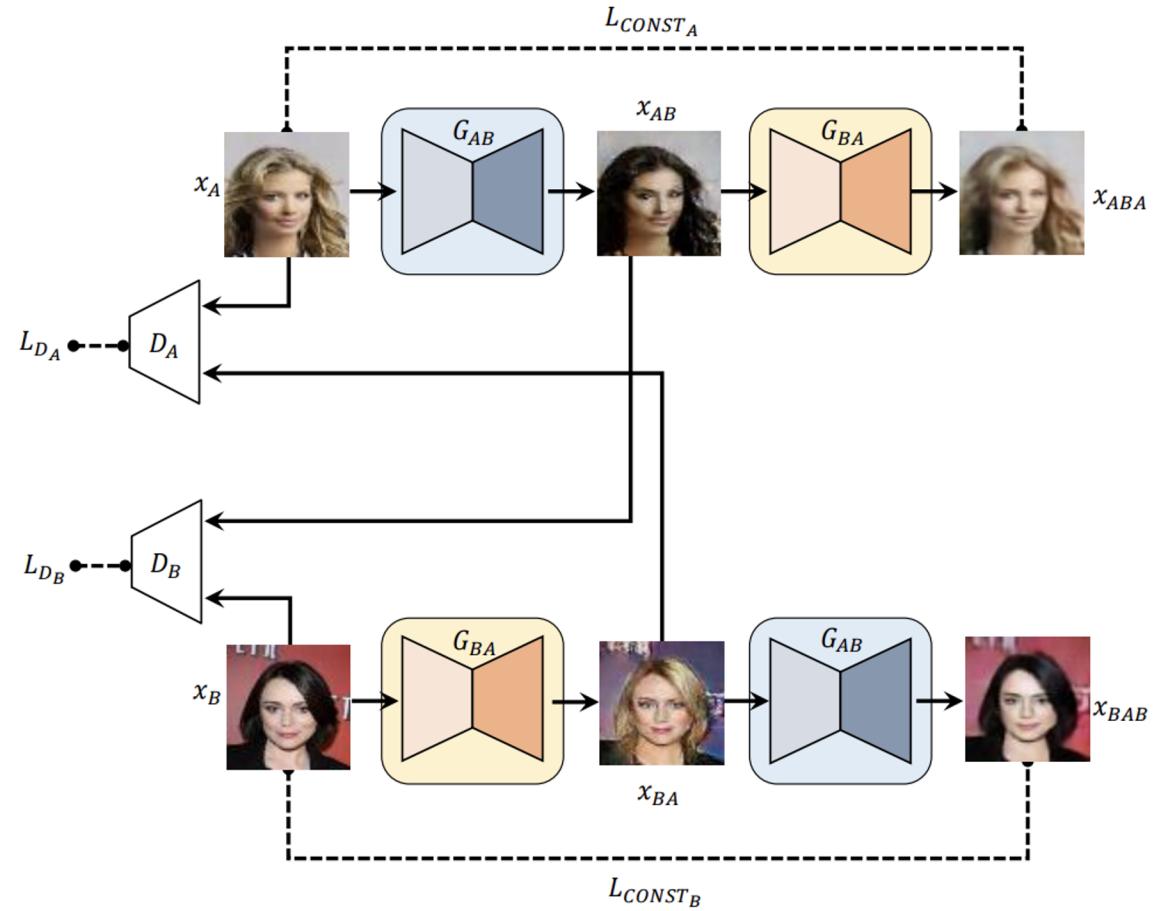


DiscoGAN

- Domain A: blond hair

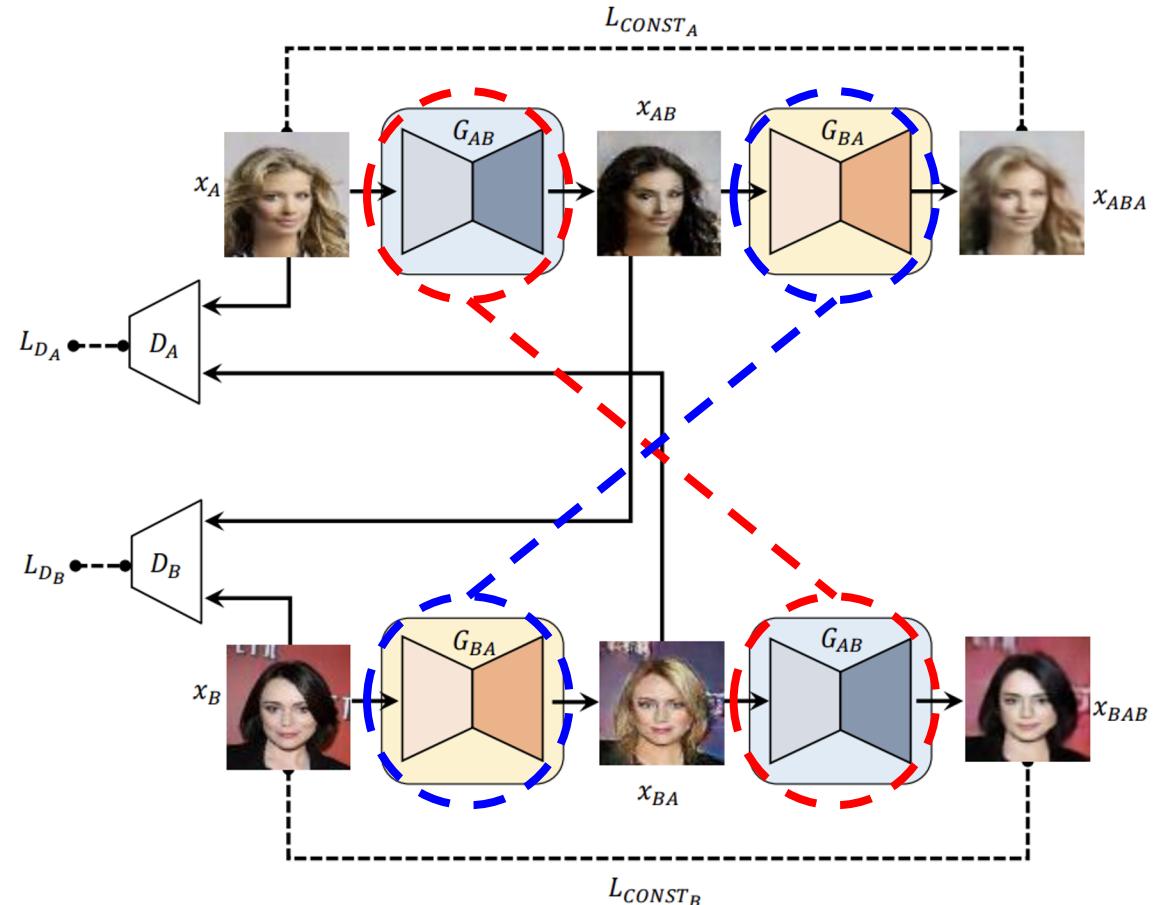


- Domain B: black hair

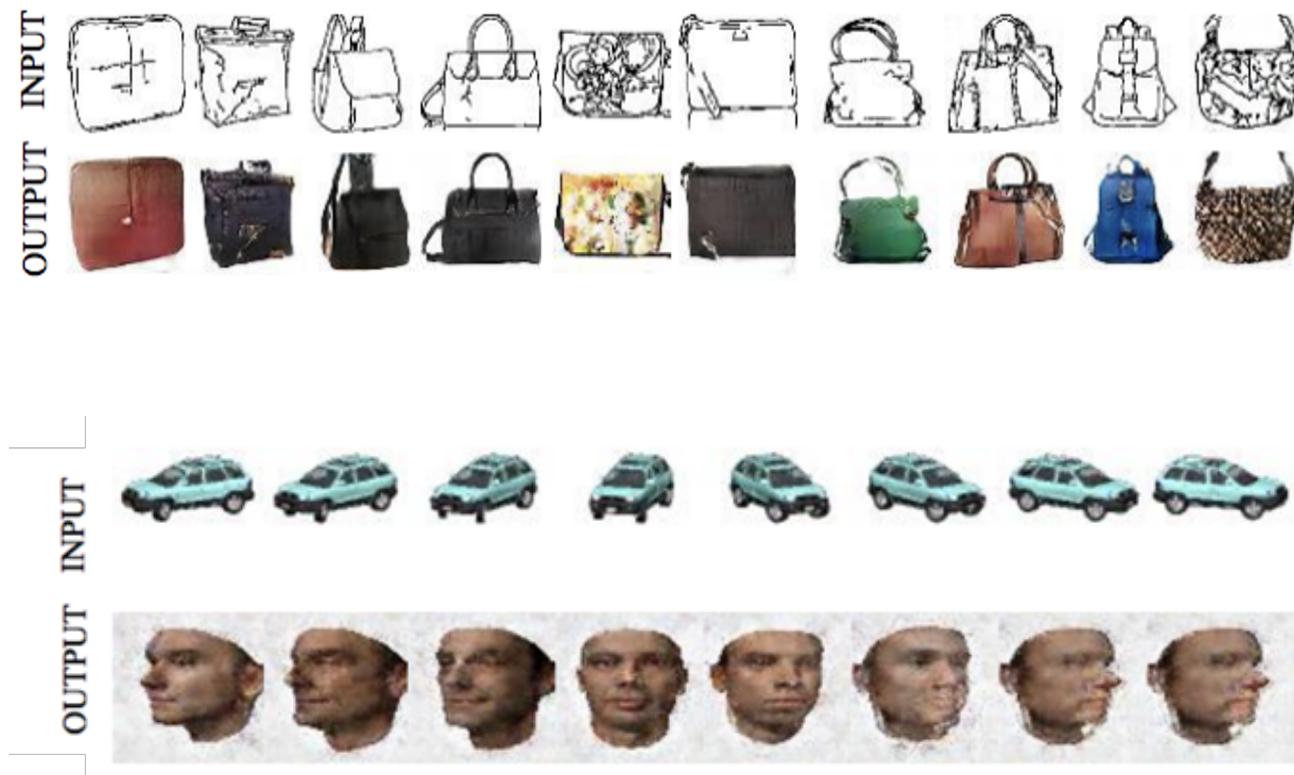


DiscoGAN

- 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



DiscoGAN



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

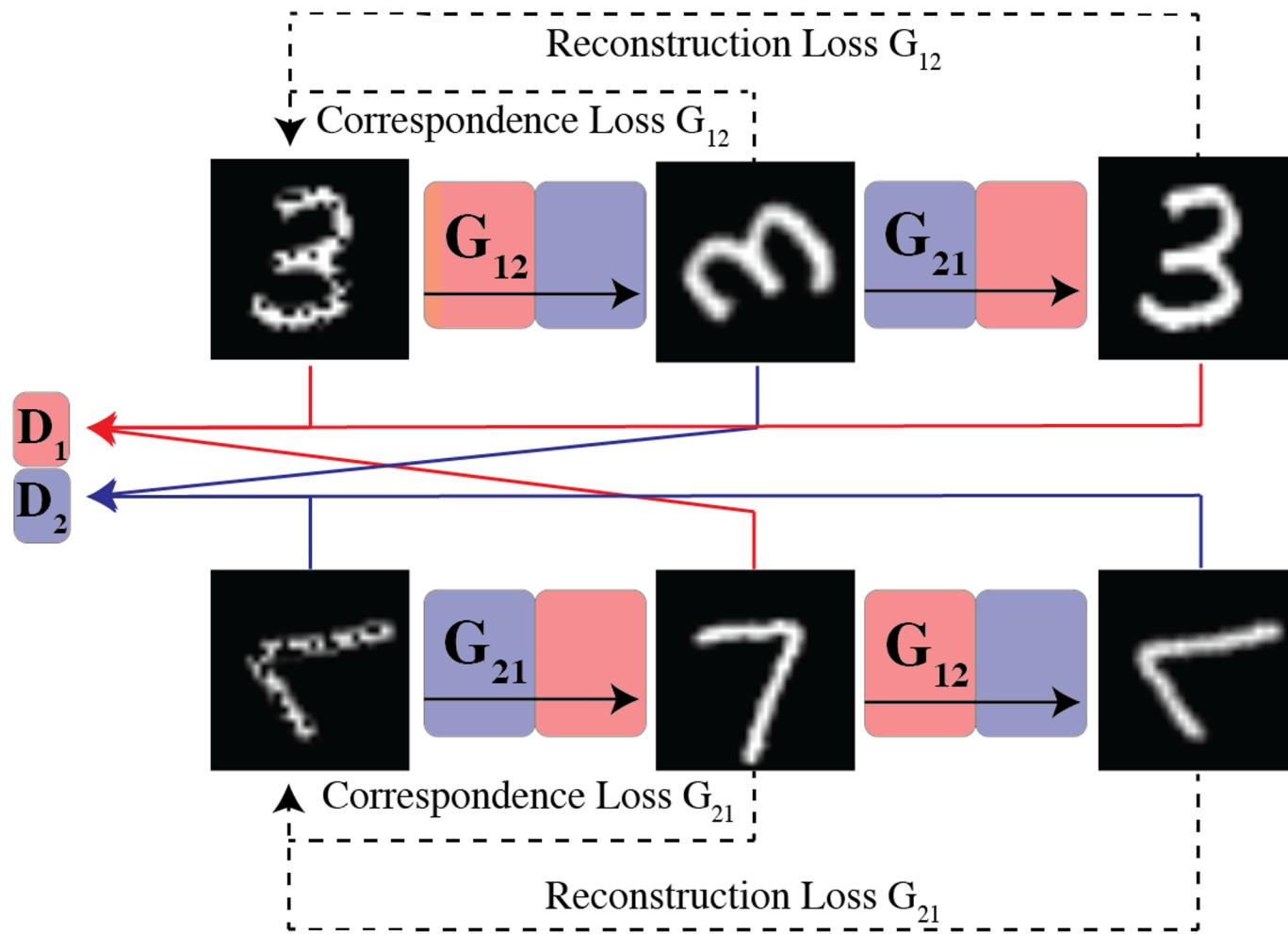


MAGAN

MAGAN: Aligning Biological Manifolds

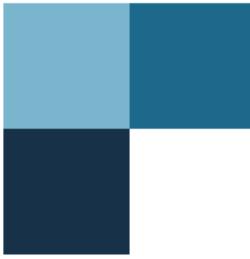
*Matthew Amodio, Smita Krishnaswamy ; Proceedings of the 35th International Conference on Machine Learning,
PMLR 80:215-223, 2018.*

MAGAN Architecture



MAGAN

- Ensures that cells translated from domain A to domain B still correspond in some way
- This is enforced by a correspondence loss
- Correspondence loss formulation requires domain specific knowledge
- How can we tell if a normal cell and a stimulated cell could correspond?
 - If the cell type is the same (stimulated T cells are still T cells)

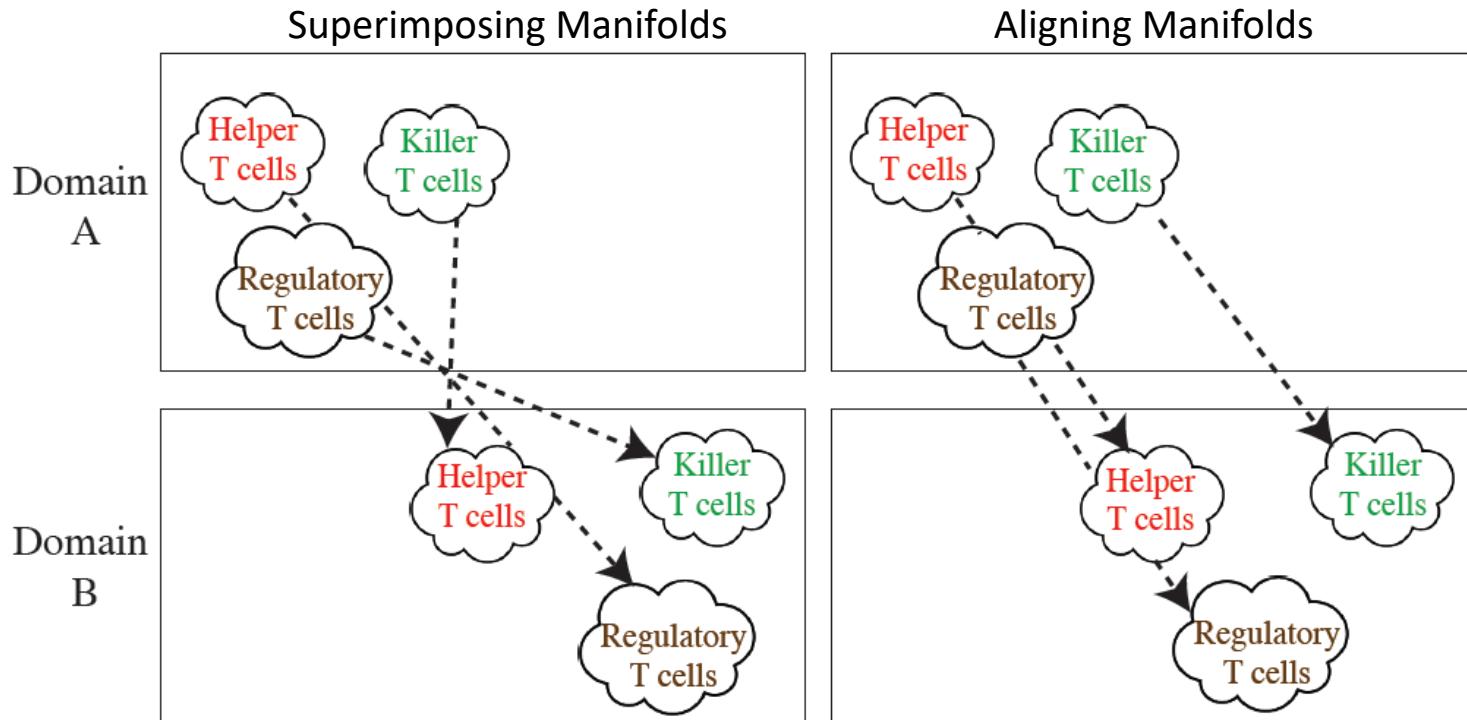


Start the presentation to activate live content

If you see this message in presentation mode, install the add-in or get help at PollEv.com/app

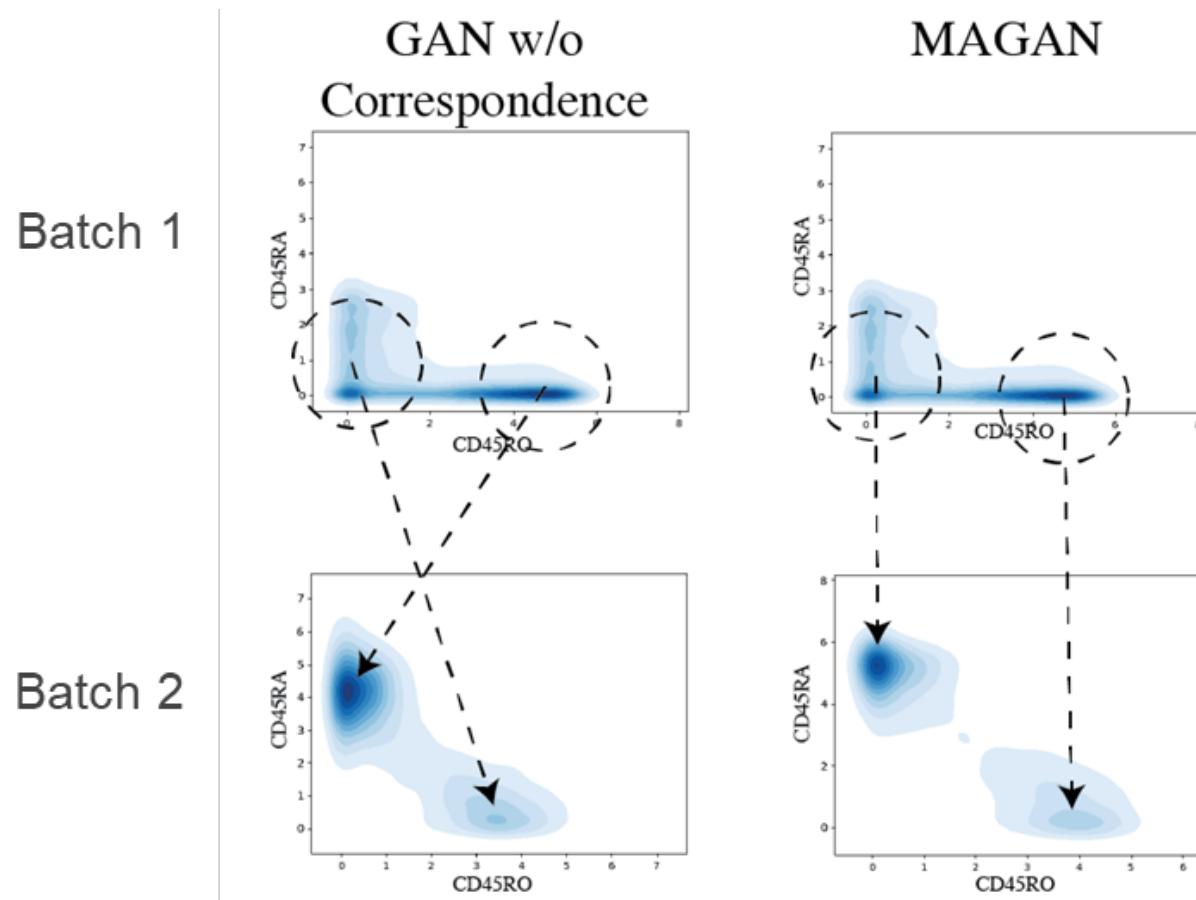
Combining 2 cytof Panels

- 2 Panels A and B
- Each panel has 40 markers, 10 are shared
- Want a dataset with $30+10+30=70$ markers!
- For each cell measured with panel A, generate panel B
- Correspondence loss: difference between shared markers should be small



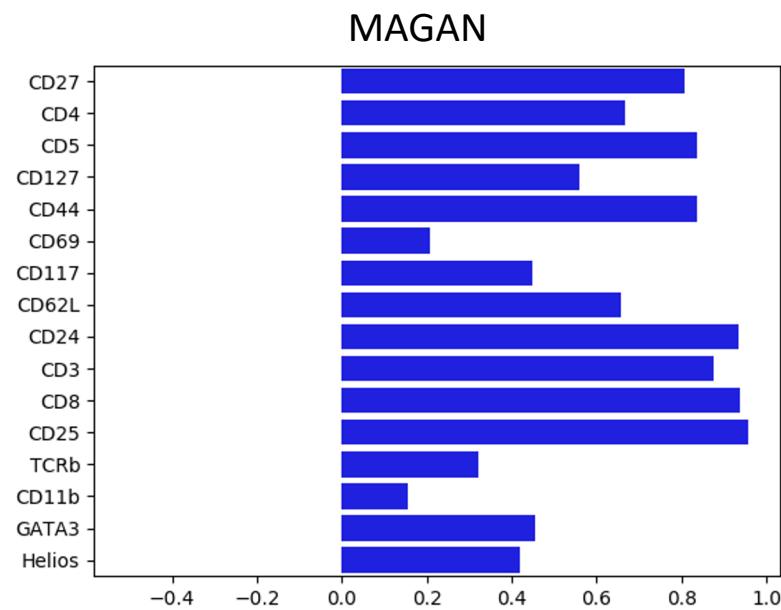
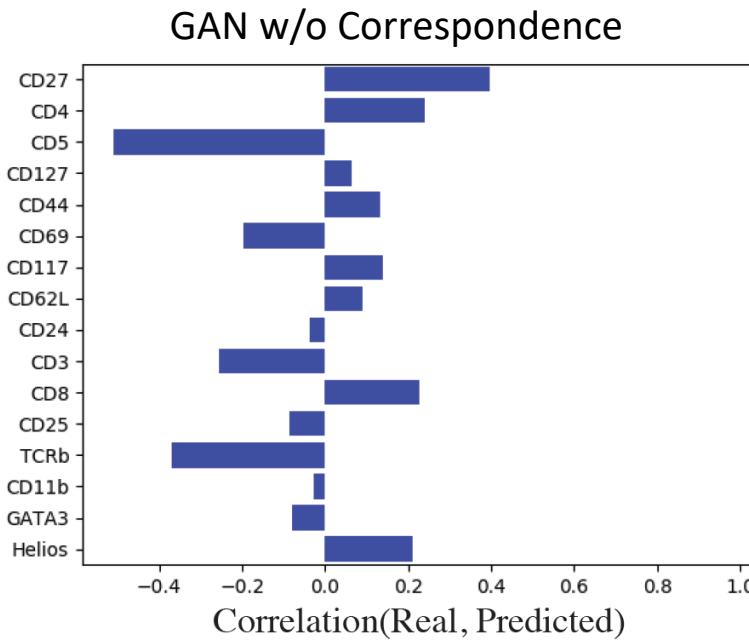
Aligning CyTOF Batches

- MAGAN correctly matches the cell types in the two batches



Aligning CyTOF Panels w/ MAGAN

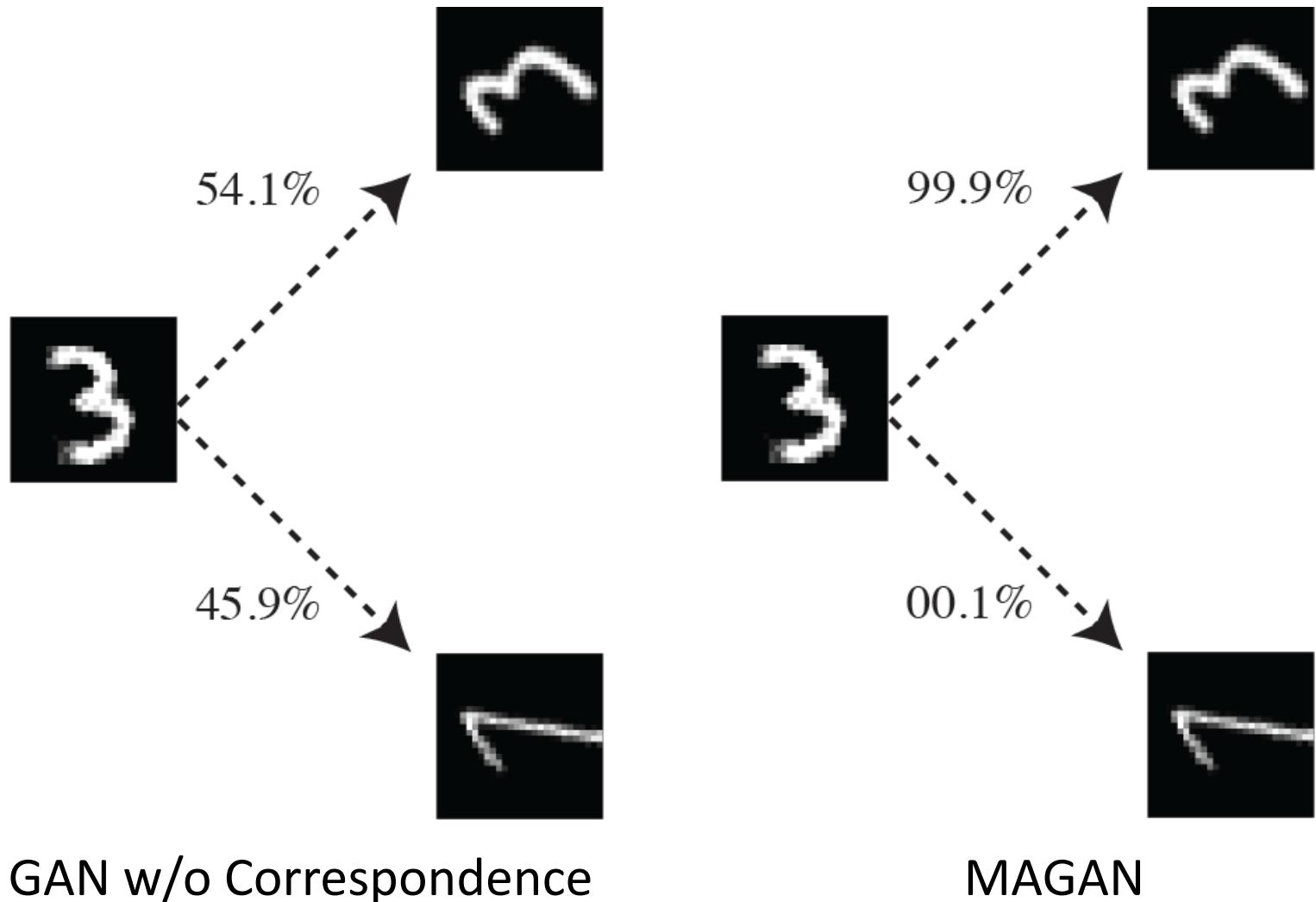
- Validation on publicly available CyTOF data from developing mouse brain*

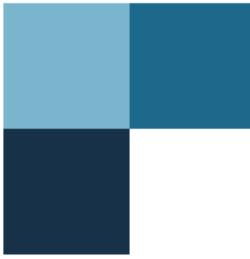


* Setty, Manu et al. "Wishbone identifies bifurcating developmental trajectories from single-cell data". *Nature Biotechnology*.

Matching rotated digits

- 100 simulations of each model





Start the presentation to activate live content

If you see this message in presentation mode, install the add-in or get help at PollEv.com/app

End of course survey

Machine Learning
for
Single Cell Analysis

End of course survey

* Required

Were you in a Beginner or Advanced group? *

Beginner

Advanced

Before the workshop, had you analyzed single cell data before? *

Yes

No

What is one thing you learned today? *

Your answer

! 

https://bit.ly/YaleML_May_PostSurvey