Software Engineering: OOP illustrated through Density Estimating Neural Networks

Albert Montillo
UTSouthwestern
Departments: Lyda Hill Department of Bioinformatics, Radiology, and Advanced Imaging Research Center

May 8-12, 2023

SWE course

Lyda Hill Department of Bioinformatics

Outline

1. **Monday**
   1. **Review OOP, Image I/O, Keras**
   2. **New topic: Object Oriented Variational Autoencoders (VAEs)**

2. **Tuesday**
   1. **Review observations on VAE**
   2. **New topic: Symbolic debugger: cond breakpoints and call stack traversal**
   3. **New topic: Object Oriented Generative Adversarial Networks (GANs)**

3. **Wednesday**
   1. **Review GAN observations**
   2. **New topic: Object Oriented *conditional* VAEs (cVAE) and *Auxillary Classifier* GANs (AKA cGAN or acGAN)**
   3. **New topic: Motivate a possible combination of cVAE and cGAN**

4. **Thursday**
   1. **Review cVAE, cGAN observations**
   2. **Review cVAE-cGAN code**
   3. **New topic: Hyperparameter optimization**
   4. **New topic: training curve and latent space traversal and visualization**

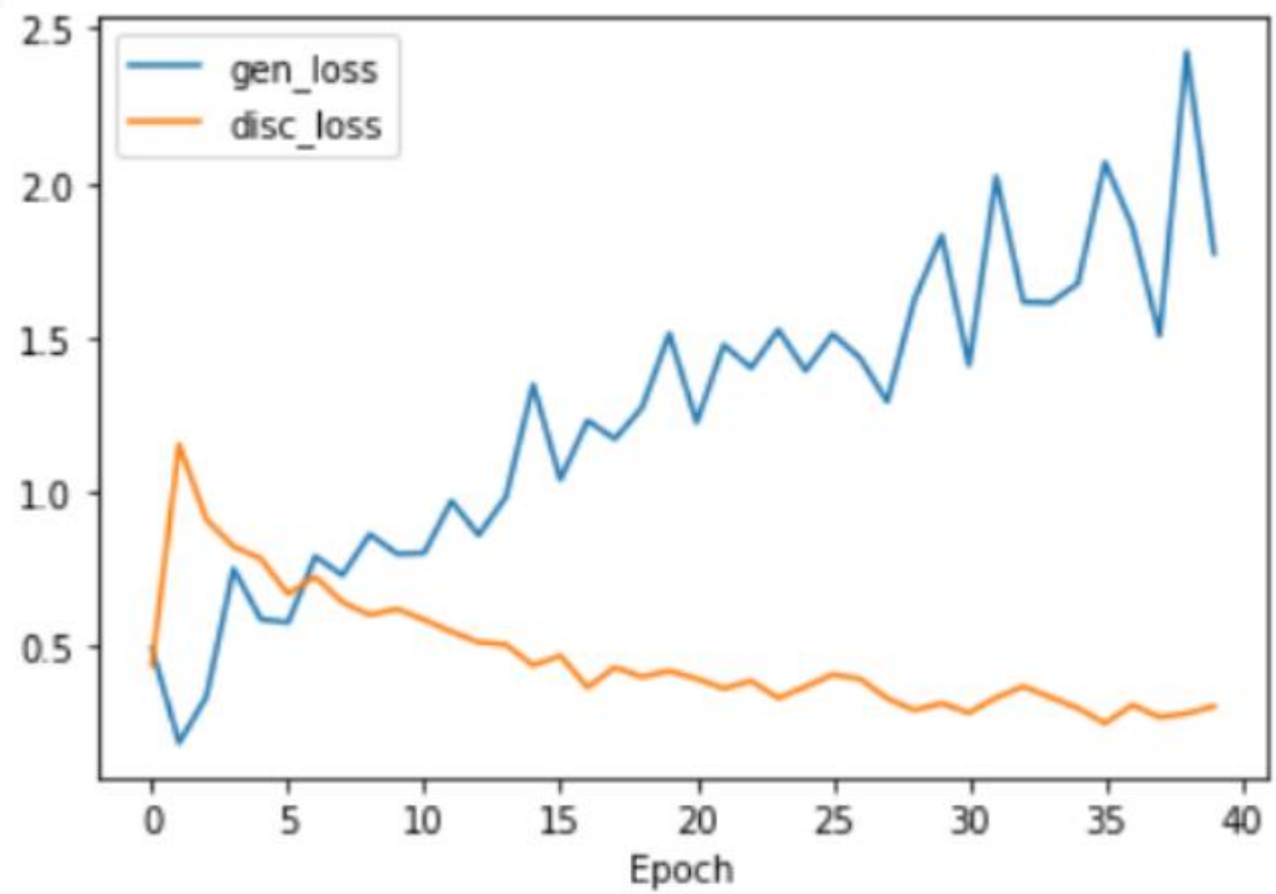
Observations from the Hands on exercise: GANs

3

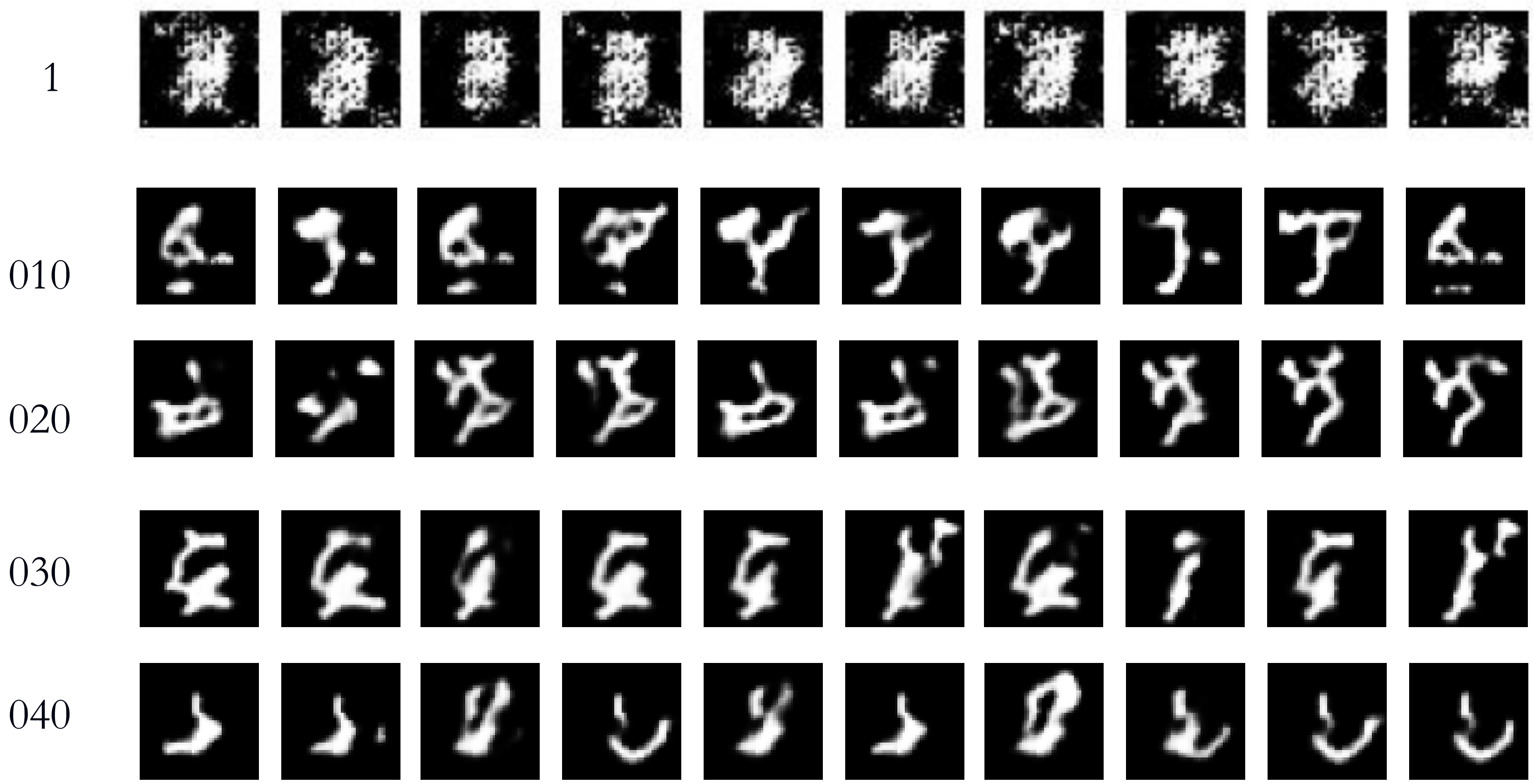
UTSouthwestern
Lyda Hill Department of Bioinformatics

GAN Training curves

1. **Shows evolution of generator and discriminator losses**
2. **We observe some divergence.. What would we do next? Later we will address**



GAN purely synthesized images at epochs 1 … 40



1. **Observations: digits are starting to appear, slowly**
2. **Clearly better at epoch 40 than 1, but more work needed.**
3. **Upshot: it can be challenging to get a simple GAN to converge**

Why Conditional VAEs?

1. **The VAE condenses latent Z space, making no gaps**

2. **While this is an improvement over AEs, it does permit generation of a specific label of data on demand.**

3. **This causes two problems**
   1. **it makes introspection of the learned latent space (density) more difficult**
   2. **It reduces the usability of the decode to generate specific labels (classes) of data (e.g. a specific digit cannot be guaranteed)**

4. **Conditional VAE remedy those limitations**

Changes to VAE to make it conditional

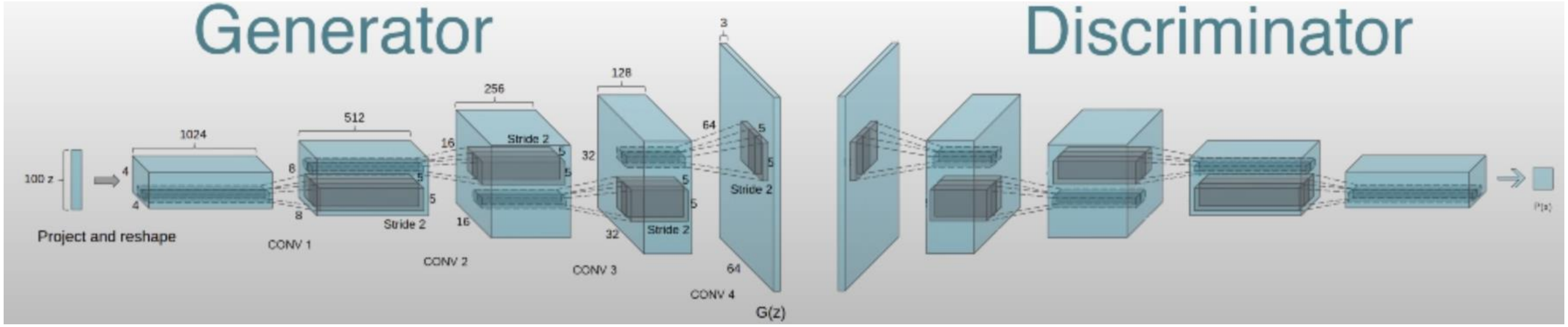
1. **Key idea: concatenate the label to the input vector and encode/decode the now expanded input vector**

2. **Concatenate a the class label when training the encoder**
   For images they are concatenated with a per-pixel one-hot encoded class label.

3. **Concatenate the class label when training the decoder**
   The latent z vector representation is concatenated with the one-hot encoded class label.

Recall: Deep convolutional GANs (DCGANs)

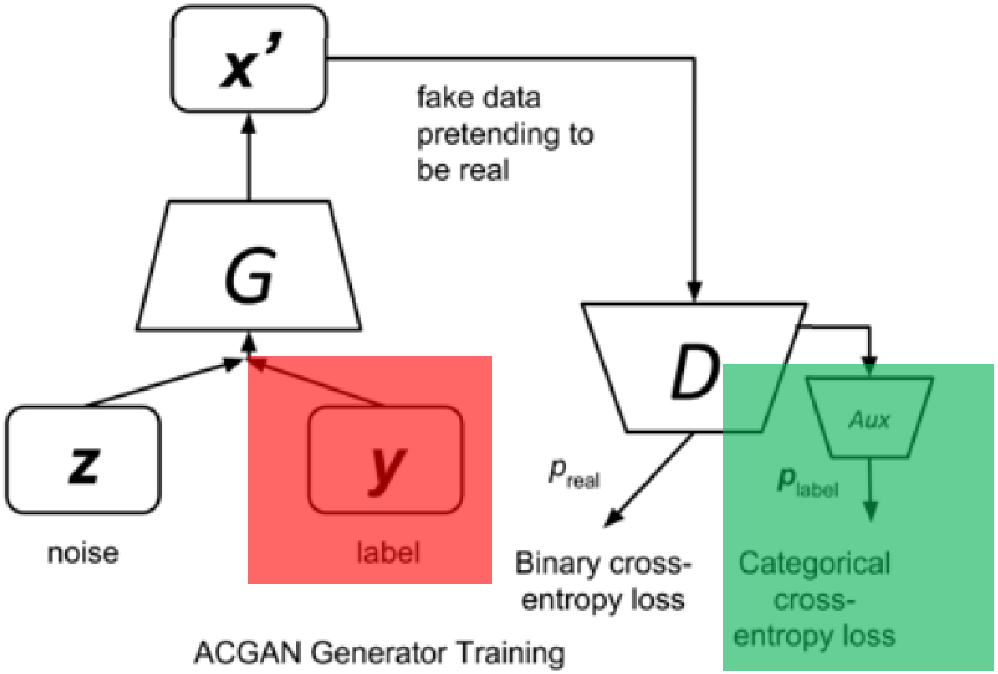
Architecture



Key ideas:

- Generator constructs an image from a random input (Z) with same dimensions as real images
- Discriminator receives images from real data and from generated
- Compatible networks
  - Generator produces images of a given size
  - Discriminator analyzes images of that size
- There is no weight tying.
- Freedom to construct Discriminator with different depth, so long as it is compatible with Generator

Extension: conditional Auxiliary Classifier GAN



- **Generator gets an additional input: class label**
- **Discriminator (now multitasking) produces an additional output: class label**

Why use cGANs for Generation?

All the benefits of GANs

- **Can be trained using back-propagation for Neural Network based Generator/Discriminator functions.**

- **Sharp images can be generated.**

- **Fast to sample from the model distribution: *single* forward pass generates a *single* sample**

Plus we attain the following

- **Generator is now steerable**
- **Discriminator can be used for decision making**

cGANs Reading List

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. and Bengio, Y. Generative adversarial nets, NIPS (2014).

Goodfellow, Ian NIPS 2016 Tutorial: Generative Adversarial Networks, NIPS (2016).

Radford, A., Metz, L. and Chintala, S., Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434. (2015).

Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., & Chen, X. Improved techniques for training gans. NIPS (2016).

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

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

Mirza, Mehdi, and Simon Osindero. Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784 (2014).

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

Denton, E.L., Chintala, S. and Fergus, R., 2015. Deep Generative Image Models using a Laplacian Pyramid of Adversarial Networks. NIPS (2015)

Dumoulin, V., Belghazi, I., Poole, B., Lamb, A., Arjovsky, M., Mastropietro, O., & Courville, A. Adversarially learned inference. arXiv preprint arXiv:1606.00704 (2016).

Applications:

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

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

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

Exercise 5
Part 1: implement cVAE
Part 2: experiment with cGAN

12

Walk through of Exercise cVAEs and cGANs

Introduce the code:

1. **Point out where the cVAE is to be implemented in vae.py**
   class ConditionalVAE(VAE):

2. **Point out the provided cGAN in gan.py**
   Play with this fully implemented code.
   class ConditionalGAN(GAN):

3. **Walk through vaegan\conditional\callbacks.py**

4. **Walk through today's caller programs:**
   1. **code/train_cvae_mnist.ipynb**
   2. **code/train_cgan_mnist.ipynb**
   3. **code/AD/train_cgan_ad.py**

5. **Tomorrow: fully implemented cVAE-cGAN to experiment with.**

Walk through of the cVAE Exercise

1. **Introduce the cVAE code:**

    1. **Walk through the structure of /code**
        1. **At the level of /code are the *caller programs* (jupyter notebooks)**

    2. **Walk through today's caller program: code/train_cvae_mnist.ipynb**
        1. **It uses MNIST data as the training set.**
        2. **Purpose is to apply the cVAE class you complete to estimate the density of MNIST images and be able to reconstruct new digit images.**
        3. **It outputs reconstructed results throughout and at the end of training here:**
           code/outputs/mnist_cvae
           Folder contains reconstructed real digits and synthesized digits (next slides)
        4. **Note: HINTS_SWE.ipynb contains helpful hints for this course.**

    3. **Class hierarchy (illustrating inheritance) is in code/vaegan**
        1. **Walk through code/vaegan/vae.py**
            1. **Overall structure, use of code folding (Ctrl+K,J and Ctrl+K,1 or 2)**
            2. **Use VSCode to edit .py use Firefox to edit Jupyter**

    4. **Note: /code/AD contains caller programs for applying what we build to AD**

Deep Learning for Precision Health Lab
www.UTSouthwestern.edu/labs/Montillo
UTSouthwestern
Lyda Hill Department of Bioinformatics

… Walk through of the VAE Exercise

1. **Walk through exercise in Syllabus**
2. **Complete the Exercise 3 “ToImplement” occurrences in vae.py**

   Show this in VSCode..

   1. **ToImplement Exercise5a**

      def make_conditional_input(self, images, labels):

   2. **ToImplement Exercise5b**

      def train_step(self, data):

Overview of the train_cvae notebook

- **Given a complete notebook to run a cVAE (train_cvae_mnist.ipynb)**
- **Given a *partially* complete cVAE (vae.py)**
- **Your job:**
  - **Understand the calling code in the notebook (.ipynb)**
  - **Understand the partial vae.py**
  - **We will review together what is missing and you will work on it together.**
- **Once you are done, test your cVAE.**
- **Write down what you observe.**

Structure of the provided cGAN in the file: gan.py

Standard GAN

- **Generator module**
  - **class Generator(tf.keras.Model)**
- **Discriminator module**
  - **class Discriminator(ft.keras.Model)**
- **GAN model**
  - **Contains one Generator and one Discriminator**

Advanced: Auxillary Classifier GAN

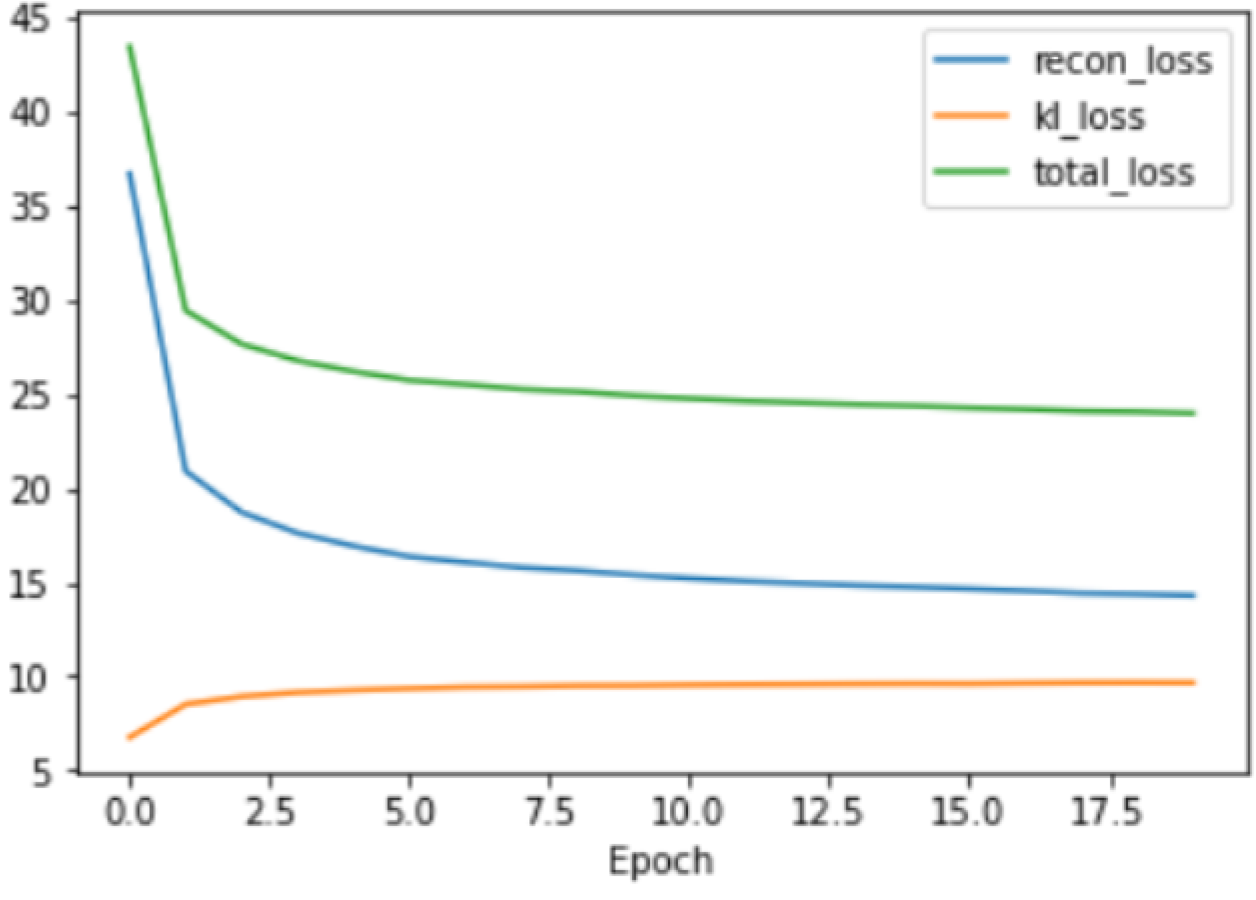
- **MultitaskDiscriminator module**
- **conditonalGAN model … an auxillary classifier GAN.**
  - **Generates images and predicts their class label**

HINTS for
Hands on cVAE exercise

24

Conditional VAE (cVAE) Training curves

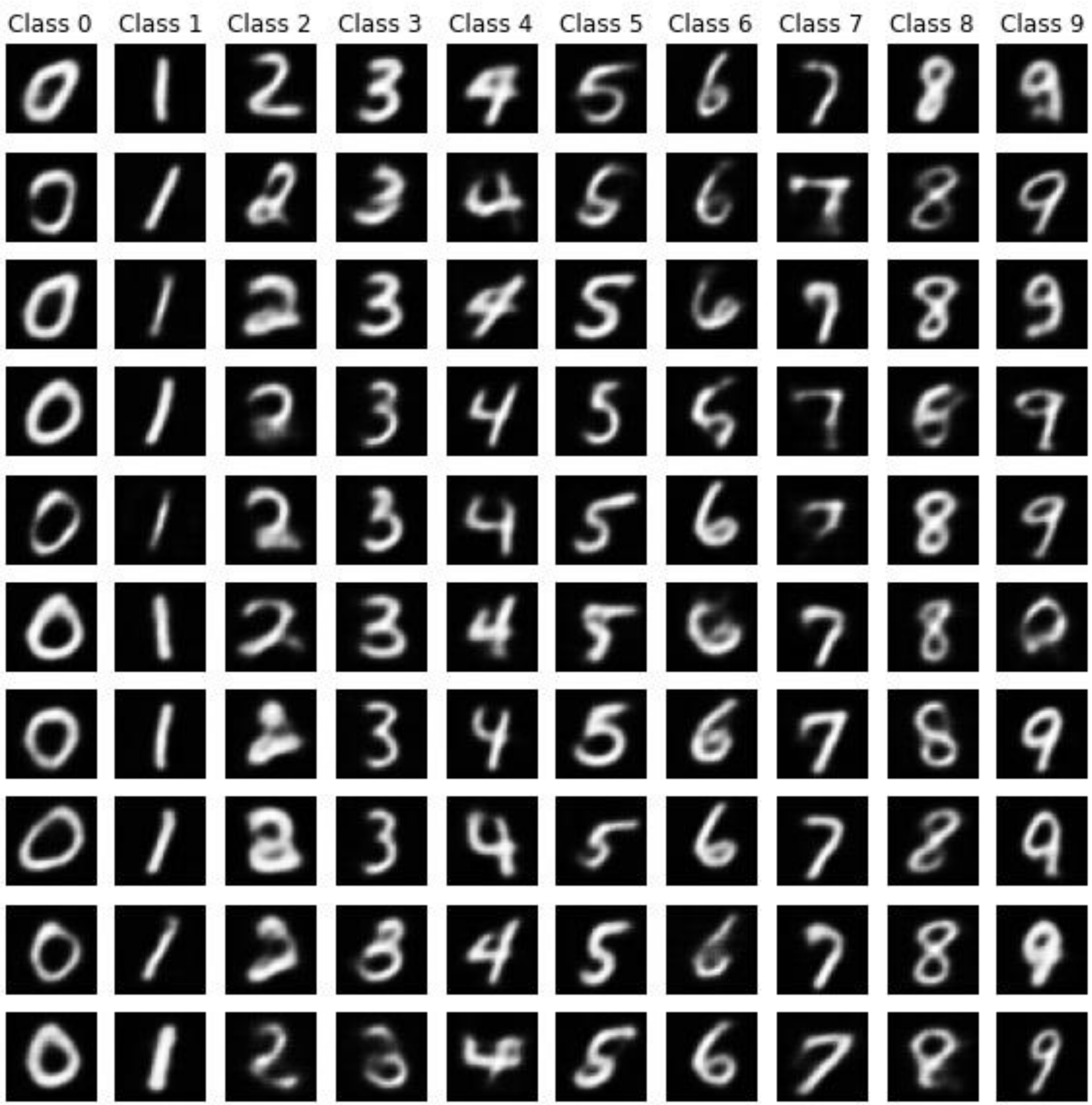
1. Shows evolution of reconstruction and regularizing prior (D_{KL}) loss as well as the total loss (their sum)
2. We observe that the majority of the learning took place in the first 10 epochs. We could train longer but diminishing returns



Conditional VAE Reconstructed images at epoch 1

Conditional VAE Purely synthesized images at epoch 1

Note: Steerable class label (column), 100 different random z's



Think about these questions:

- **When you train longer, Are good digits are produced?**

- **What do you observe about the appearance of the strokes of the digits?**

- **In the training curves plot, what is happening to the generator and discriminator losses? What does that signify? Practical implication?**

- **What practical value does *conditional* VAE add over VAE?**
- **What practical value does *conditional aux classifier* GAN add over GAN and over cVAE?**

- **Qualitatively compare your generated images across the models you have learned in this week: What pros and cons of the results do you see from VAE, GAN, cVAE, and cGAN?**

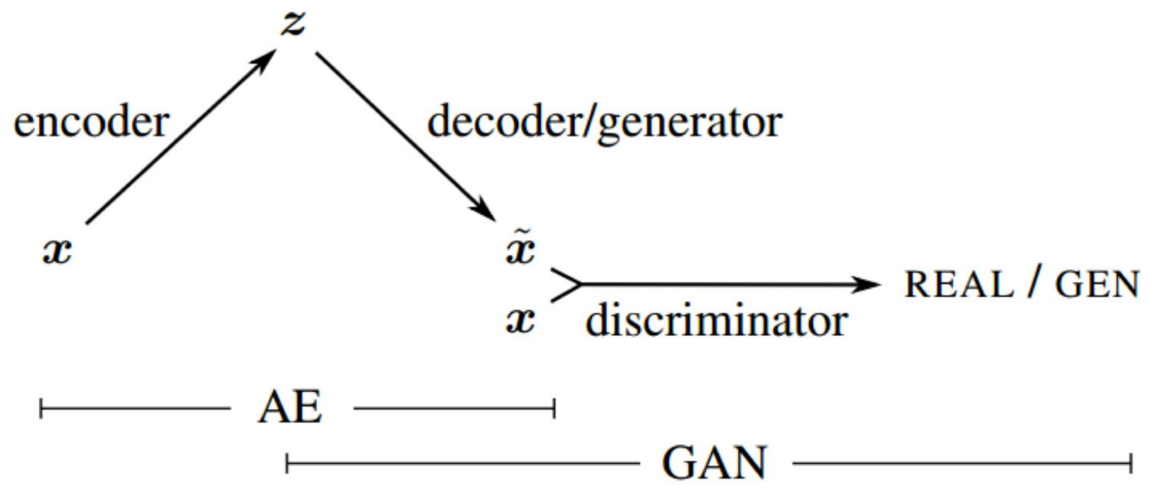
Combining VAEs and GANs: The cVAE~cACGAN

- **Motivation:**
  - **Want to generate images that have sharp edges of the GANs as this is a property of convincing real world images.**
  - **Also want to smoothly traverse the learned space of images to understand better our data**

- **Observation 1: Loss functions are different**
  - **The pixel wise loss from the VAE is to blame for the blurry images.**
  - **The GAN has a semantic loss from the extraction of semantic features through the layers of the Discriminator. This semantic loss is the reason for the sharper (less blurry) images.**

cVAE~cACGAN

- **Observation 2: Compatible/duplicate parts**
  - **Decoder of a VAE performs largely the same job as the generator of the GAN**



VAE-GAN architecture, the discriminator from GAN takes input from VAE's decoder

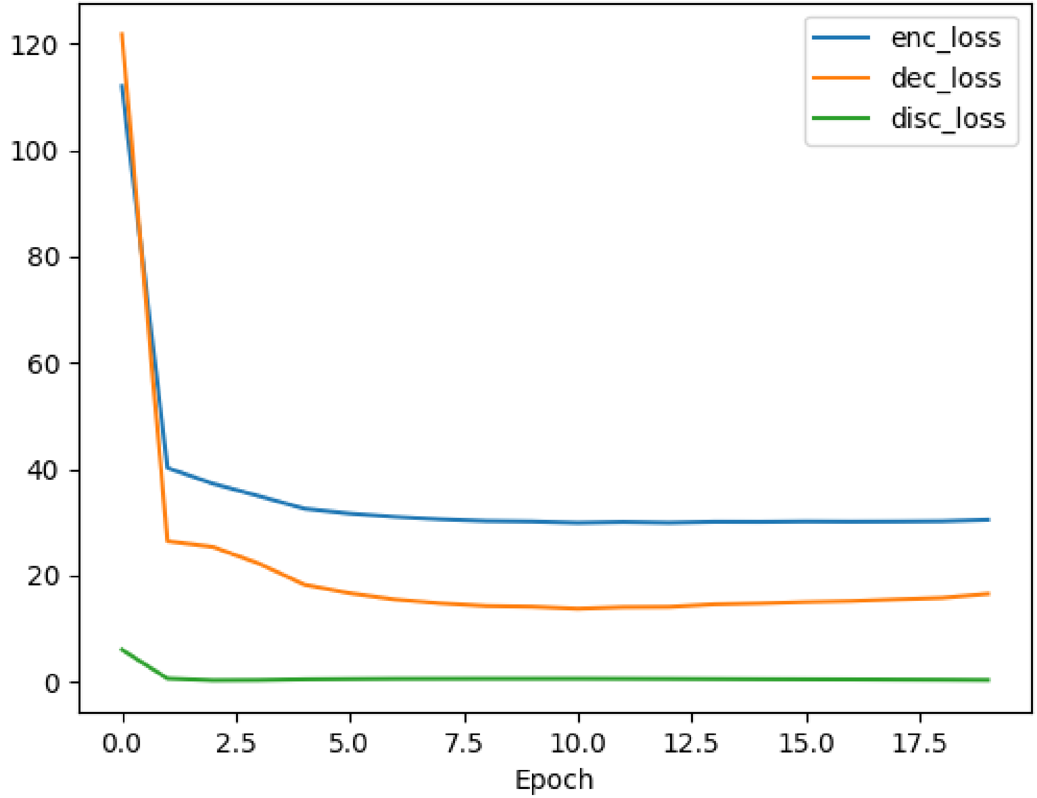
- **Observation 3: value of linkage**
  - **The encoder of the VAE allows us to find the location in the compressed latent space of any input image, this also helps us understand the learned embedding.**
  - **Backpropagating the discriminator loss (and pixel wise recon loss) back to the encoder/decoder allows to retain sharpness and high quality images whose embedding space we can traverse and explore.**

Observations from when you run cVAE-cGAN on MNIST

31

cVAE~cACGAN Training curves

1. Shows evolution of reconstruction and regularizing prior (D_{KL}) loss as well as the total loss (their sum)
2. We observe that the majority of the learning took place in the first 10 epochs. We could train longer but diminishing returns



cVAE~cACGAN Reconstructed images at epoch 20

Real:

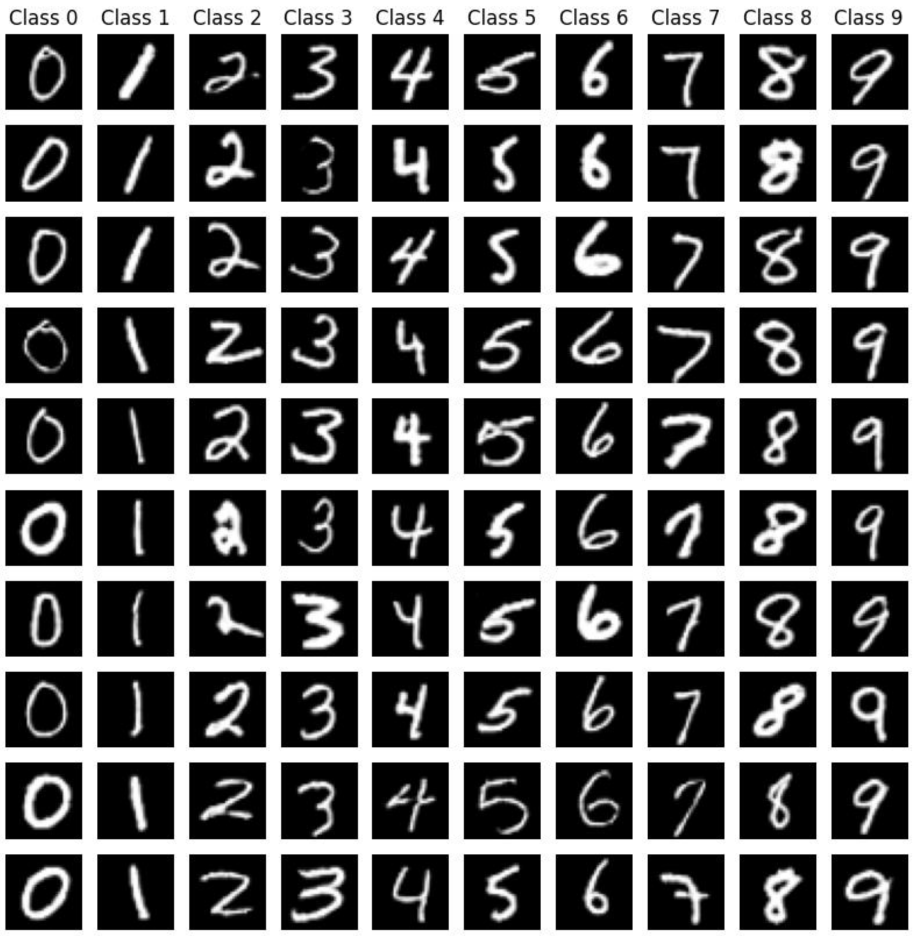
Recon:

We observe sharpness of ACGAN with high digit quality of cVAE.

cVAE~cACGAN purely synthesized images at epoch 20

Note: Steerable class label (column), 100 different random z's

We observe sharpness of ACGAN with high digit quality of cVAE.
We still have the steerable capacity of Conditional VAE and the ability to classify images as well.
Combined strengths of both architectures



cVAE~cACGAN test results

Using test script: python test_cvaecgan_mnist.py
Note: Same latent ("style") per column, but changing the class label
Each column has same z, 10 rand z's

We observe: *<u>disentanglement</u>* of style from digit

Also attains 97.46% digit classification accuracy

though not SoA, not bad for only 20 epochs!

and we get all of the synthesis capabilities and insights

Request (optional)
Run cVAE-cGAN on Alzheimer’s dataset

37

Acknowledgements

Albert Montillo, PhD, PI

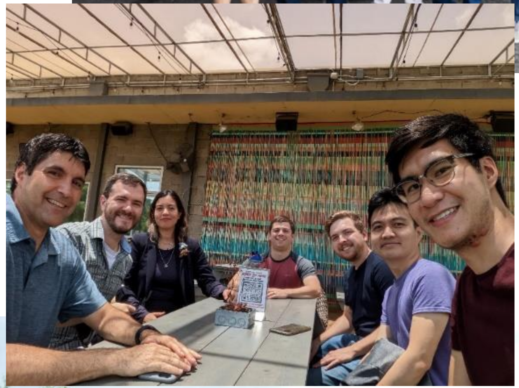
Son Nguyen, PhD
Postdoc

Alex Treacher
PhD student

Aixa Andrade Hernandez,
MS, PhD student

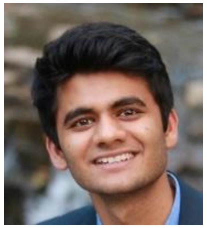
Austin Marckx
PhD student

Krishna Chitta
Res. Sci.



------------ Recent Alumni -------------

Atef Ali
Undergrad



Vyom Raval, BS
MD/PhD

Kevin Nguyen
MD/PhD student

Cooper Mellema
MD/PhD student

Lab Funding

- **NIH/ NIGMS R01** ***Correcting Biases in Deep Learning***
- **King Foundation (PI) : Quantitative AD diagnostics.**
- **Lyda Hill Foundation (PI): Quantitative prognostics of Parkinson's disease**
- **NIH/ NIA R01 Blood Biomarkers for Alzheimer's and Parkinson's**
- **TARCC : Texas Alzheimer's Research and Care Consortium.**
- **NIH / NINDS F31 fellowship : Causal connectivity biomarkers for neurological disorders**

Thank you!

Email: Albert.Montillo@UTSouthwestern.edu
Github: https://github.com/DeepLearningForPrecisionHealthLab

MegNET …. Artifact suppression
BLENDS …. fMRI augmentation
Antidepressant-Reward-fMRI …. response prediction
Parkinson-Severity-rsfMRI … disease trajectory prediction

End of presentation