ECE685 – Fall 2023
Project Proposal

Teach Staff

C1: Few-shot Learning for Object Detection (Cap 3)

- Objective:
  - Develop object detection system to detect American Sign Language (ASL), which used one hand.
  - Consider the British Sign Language (BSL), which used two hands.
  - Apply few-shot learning and/or continual learning methods to adapt the existing ASL model to BSL.
- Dataset:
  - ASL Dataset: https://www.kaggle.com/datasets/grassknoted/asl-alphabet
  - BSL Dataset: https://www.robots.ox.ac.uk/~vgg/data/bobsl/
- Object detection model
  - YOLOv4: https://medium.com/@eacabrera3/real-time-sign-language-detection-system-e3d6cf49121a

C1: Few-shot Learning for Object Detection (Cap 3)

- Outline:
  - Literature Review: few-shot learning, continual learning
    - https://arxiv.org/abs/1703.03400
    - https://arxiv.org/abs/2110.02399
    - https://arxiv.org/abs/2205.15445
  - Define the few-shot problem as **episodically finetuning** or **pretraining approaches**.
    - Few-shot learning:
      - MAML: https://arxiv.org/abs/1703.03400
      - TAS: https://arxiv.org/abs/2110.02399
    - Continual learning:
      - https://arxiv.org/abs/2302.00487#:~:text=To%20cope%20with%20real%2Dworld,systems%20to%20develop%20themselves%20adaptively.
      - https://arxiv.org/abs/2205.15445

- Responsible TA: Cat Le

C2: Continual Learning for Text-to-Image Generation (Cap 4)

- Objective :
  - Develop text-to-image generative model for flowers
  - Adapt this generator for the new task (birds), without forgetting the existing tasks (flowers).
- Dataset:
  - Flower Dataset: https://paperswithcode.com/sota/text-to-image-generation-on-oxford-102
  - CUB Bird Dataset: https://paperswithcode.com/sota/text-to-image-generation-on-cub
- Text-to-Image GAN model:
  - https://www.kaggle.com/code/msripooja/text2image-gan
  - https://towardsdatascience.com/generating-synthetic-images-from-textual-description-using-gans-e5963bae0df4

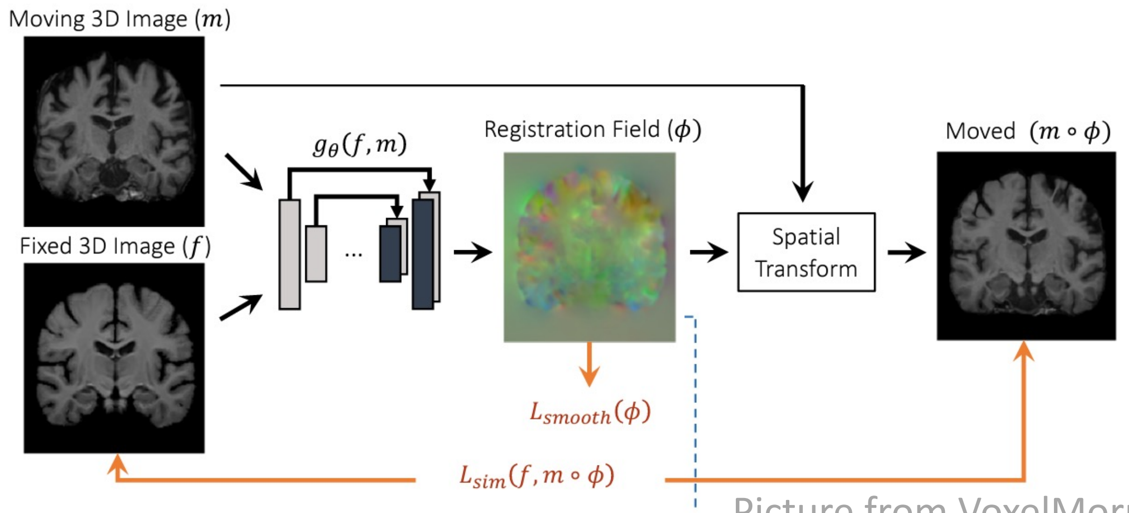
C2: Continual Learning for Text-to-Image Generation (Cap 4)

- Outline:
  - Literature Review: continual learning GAN and text-to-image generation
    - https://arxiv.org/pdf/1907.10107.pdf
    - https://arxiv.org/pdf/1811.11083.pdf
    - https://arxiv.org/pdf/1605.05396.pdf
    - https://proceedings.neurips.cc/paper_files/paper/2019/file/1d72310edc006dadf2190caad5802983-Paper.pdf
  - Text-to-Image learning:
    - https://arxiv.org/pdf/1605.05396.pdf
    - https://proceedings.neurips.cc/paper_files/paper/2019/file/1d72310edc006dadf2190caad5802983-Paper.pdf
  - Continual learning GAN:
    - https://arxiv.org/pdf/1907.10107.pdf
    - https://arxiv.org/pdf/1811.11083.pdf
- Responsible TA: Cat Le

Z1&Z2: Unsupervised Deformable Image Registration

- **Introduction**: Image registration is an important technique in image analysis. The goal of image registration is to finding a geometric transformation between two/multiple sets of images. Diffeomorphic registration aims to establish the non-linear correspondence between images.

- **Expectation**: In this project, the minimum requirement is to follow the method in VoxelMorph paper and define your own network to deal with different dataset (depends on the group size). You are also encouraged to try other related methods in this field.



Picture from VoxelMorph

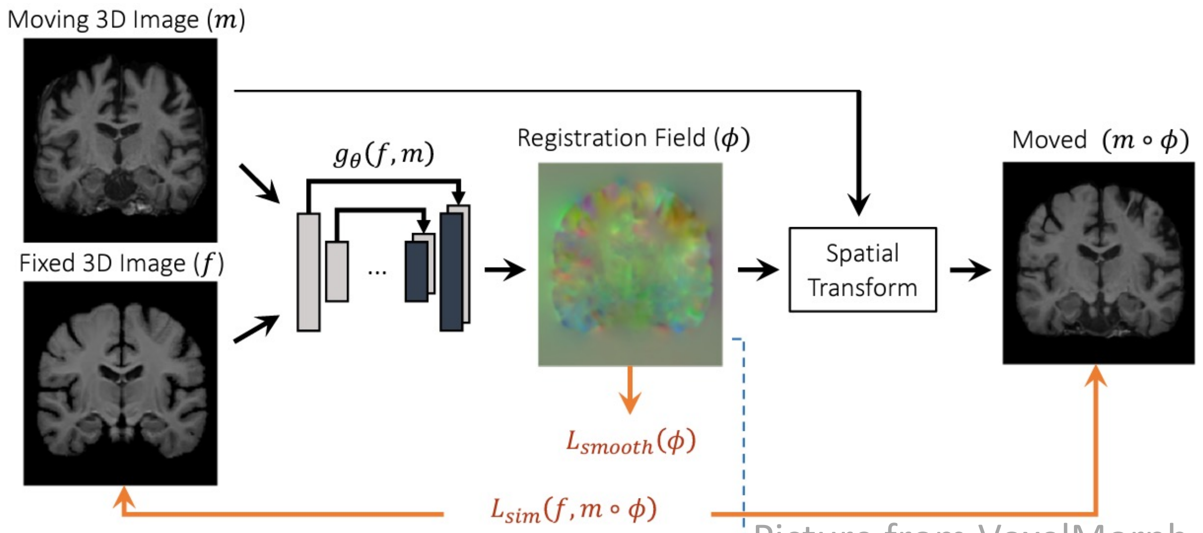
Z1: Unsupervised Deformable Image Registration (Cap 1)

- **Dataset**: MNIST

- **Pipeline**:
  - Use one class of the MNIST dataset (e.g., "7") as your dataset.
  - Define your own network g_theta (note: you won't need a complex architecture like U-Net since your task is relatively easy, please design your own network)
  - Train your registration network and report the results with different metrics.
  - Repeat this process with different classes too.

- **Suggested Papers**:
  - Paper 1: [VoxelMorph]()
  - [Paper 2]()

Responsible TA: Ziyun Yang



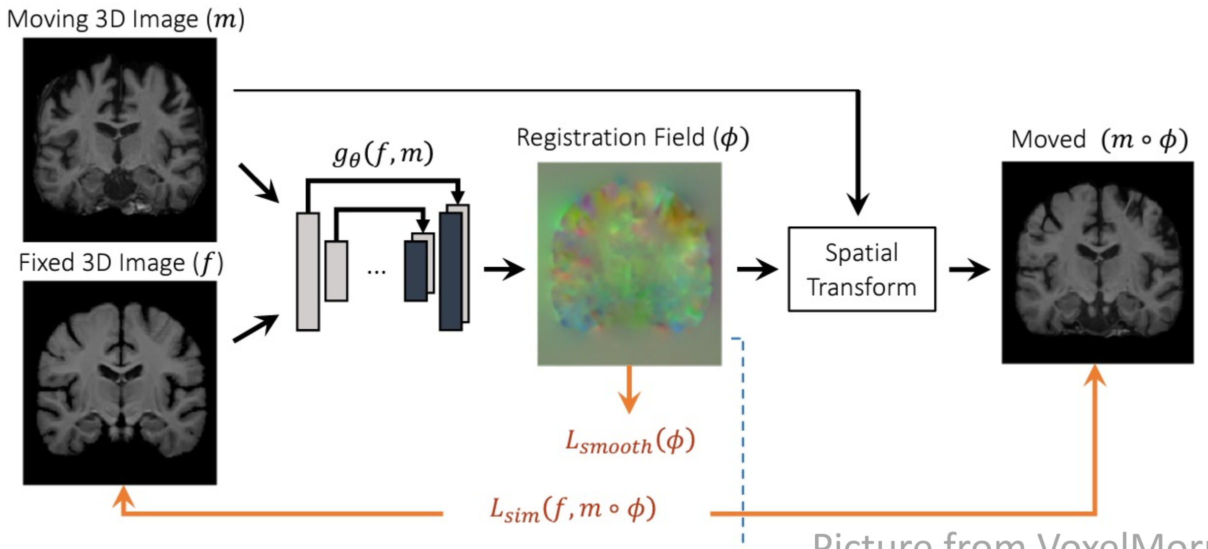
Picture from VoxelMorph

Z2: Unsupervised Deformable Image Registration (Cap 3)

- **Dataset**: Please use any of the following public datasets: OASIS (link), ACDC (link), etc.

- **Pipeline**:
  - Use the slices in a volumetric data as your training dataset.
  - Design your own training strategy and data allocation strategy (i.e., what should be set as static frame and moving?)
  - Try to reduce the size of your model g_theta (e.g., reduce one block in the encoder-decoder) to see what results you can achieve with less parameters.
  - Evaluate your results and make a report
- **Suggested Papers**:
  - Paper 1: VoxelMorph
  - Paper 2

Responsible TA: Ziyun Yang



Picture from VoxelMorph

Z3: Knowledge Distillation Techniques in Salient Object Detection (Cap 3)

- **Introduction**: SOD aims to detect the most eye-attracting parts from a given image. As a fundamental task in computer vision, it plays an essential role in scene understanding. In salient object detection (SOD), speed and computational cost is crucial for a real-time task. Knowledge distillation (KD) is the process of transferring knowledge from a large model to a smaller one without loss of validity. As smaller models are less expensive to evaluate, they can be deployed on less powerful hardware (such as a mobile device).

- **Goals**: Make your compact version of CNN and implement KD algorithms in SOD. Compare the performance, model size, inference speed etc.

SOD: HKU-IS Dataset

Z3: Knowledge Distillation Techniques in Salient Object Detection (Cap 3)

- **Dataset**: DUTS-TR dataset for training and ECSSD, DUT-TE, and DUT-OMRON for testing.

- **Pipeline:**
  - Train a GCPANet on the above dataset.
  - Make a compact version of GCPANet by reducing the size of it (e.g., by removing blocks from the Encoder/Decoder or changing backbone) and retrain it.
  - Implement one KD algorithm on the two models.
  - Compare the performance of the compact network before and after the KD.

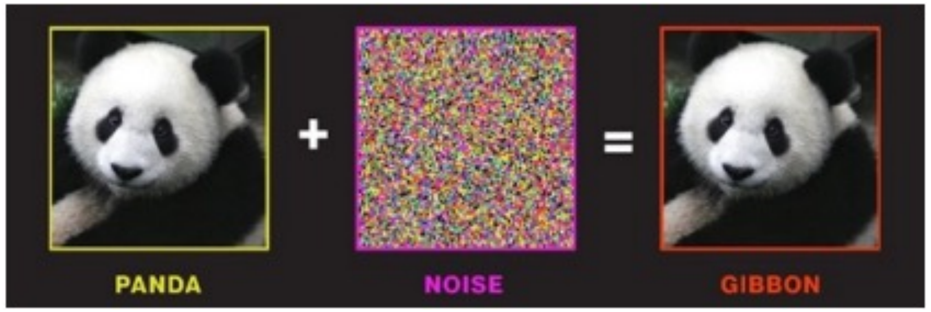
- **Suggested Papers**:
  - SOD: GCPANet
  - KD: Structure KD

SOD: HKU-IS Dataset

Responsible TA: Ziyun Yang

Y1: Black Box Adversarial Attack (Cap 4)

- Background
  - In Deep Learning, adversarial attack is a process of generating deceptive data that lead to the malfunction of a neural network. However, such data should be hardly able to fool humans. A classic example of this is the panda to gibbon case as described in the image below.



  - There are two primary types of adversarial attacks to neural networks: white-box attack and black-box attack. White box attack allows access to model weights. Direct gradient optimization is allowed in this case. However, black box attack allows access to more limited information.

Y1: Black Box Adversarial Attack (Cap 4)

- Objective
  - In this project, you will read three literature relevant to black box attack/defense and implement the algorithms listed. As an attacker, you have access to at most the probability vector in model outputs, but not the gradient nor its architecture. As a defender, you can arbitrarily modify your model architecture and parameters.
  - You need to perform box attack/defense on ImageNet 1k in addition to one of the dataset listed.
  - You should report the attack success rate and query count for the two attack algorithms when a defense mechanism exists/not exist. You need to also briefly discuss why or why not the defense algorithm is effective.
- Literature
  - https://arxiv.org/pdf/1804.08598.pdf (Partial information attack with NES Gradient Estimate)
  - https://arxiv.org/pdf/1904.02144.pdf (HopSkipJumpAttack)
  - https://arxiv.org/pdf/2205.12134.pdf (Adversarial attack on attackers)
- Dataset
  - Butterfly & Moths Image Classification 100 species (kaggle.com)
  - TIME -Image Dataset-Classification (kaggle.com)

Responsible TA: Yixin Zhang

Y2: Weakly Supervised Semantic Segmentation (Cap 4)

- Background
  - The semantic segmentation task is to assign a label from a label set to each pixel in an image. In the case of fully supervised setting, the dataset consists of images and their corresponding pixel-level class-specific annotations (expensive pixel-level annotations). However, in the weakly-supervised setting, the dataset consists of images and corresponding annotations that are relatively easy to obtain, such as image-level annotation, bounding box, points and scribbles.
- Project Objective:
  - Review a few literatures related to weak-supervised learning
  - Training segmentation models with point annotation or image-level annotation
  - Compare the performance disparity between SAM-assisted (preprocessing-based) and Iterative Self-Improved (iterative update) model training with weak annotations.

Y2: Weakly Supervised Semantic Segmentation (Cap 4)

- Related literature:
  - arxiv.org/pdf/2211.12455.pdf
  - https://github.com/facebookresearch/segment-anything
  - 1506.02106.pdf (arxiv.org)
- Dataset
  - PASCAL VOC (official)
  - PASCAL VOC Aug (https://www.sun11.me/blog/2018/how-to-use-10582-trainaug-images-on-DeeplabV3-code/)

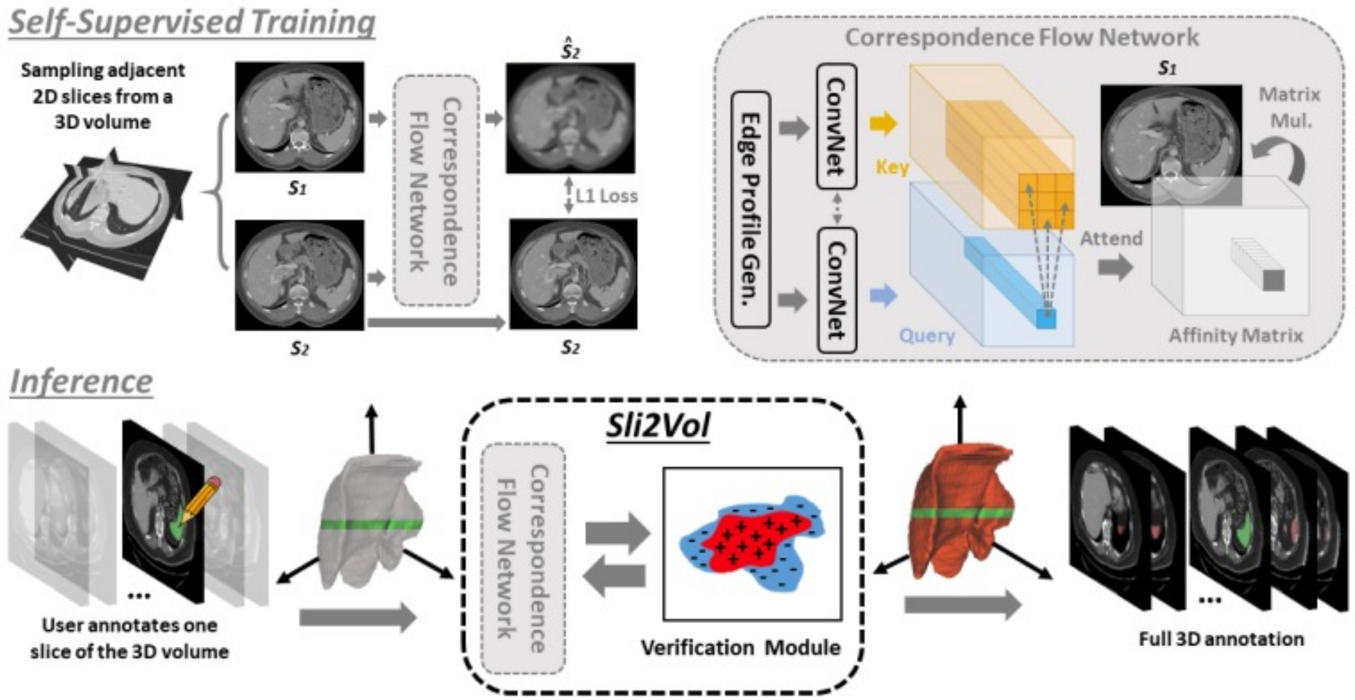
*** Please consider using devices with RTX3070 or better. You may adapt from the Segment Anything Model (SAM) GitHub site for an example of inference with SAM on Jupyter notebook.

Responsible TA: Yixin Zhang

Y3: Self-supervised One-slice Annotation (Cap 3)

- Background

  Images generated from CT and MRI scans typically come as 3D volumes. Each 3D volume may contain hundreds of slices, where each pair of adjacent slices contain similar contents. Conventionally, each of the slices need to be annotated if we want to use the 3D volumes to train a segmentation network. However, since adjacent slices contains similar content, this conventional practice may not be efficient when annotation resources are restricted.


Self-Supervised Training
Sampling adjacent 2D slices from a 3D volume
S1
S2
Correspondence Flow Network
Ŝ2
L1 Loss
S2
Correspondence Flow Network
Edge Profile Gen.
ConvNet
ConvNet
Key
Query
S1
Matrix Mul.
Attend
Affinity Matrix
Inference
User annotates one slice of the 3D volume
Sli2Vol
Correspondence Flow Network
Verification Module
Full 3D annotation

Y3: Self-supervised One-slice Annotation (Cap 3)

- Related literature:
  - [2105.12722] Sli2Vol: Annotate a 3D Volume from a Single Slice with Self-Supervised Learning (arxiv.org)
  - Self-learning and One-Shot Learning Based Single-Slice Annotation for 3D Medical Image Segmentation | SpringerLink
- Dataset:
  - LiTS – Liver Tumor Segmentation Challenge (LiTS17) - Academic Torrents

- Project Objective
  - 1. Get familiar with the selected literature and self-supervised learning
  - 2. Learn to work with medical datasets (You may use TorchIO or MONAI library)
  - 3. Make your own implementation of the code with comparable performance to the selected literature. The two literatures have different implementation for solving similar problems. You may propose your own solution, if any.
  - 4. Experiment and report model performance by define the encoder architecture as
    - 1. Some variants of Resnet (e.g., ResNet18)
    - 2. Some variants or derivatives of Vision Transformers (e.g., ViT).

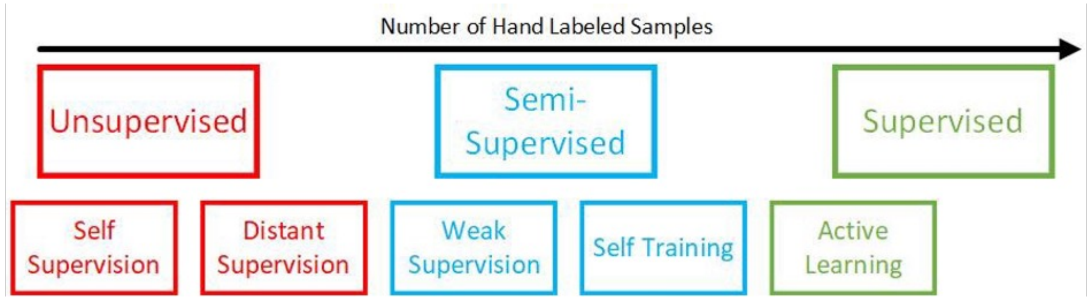
***You many changes any part of the code you feel necessary. However, you must properly select one annotated slice and propagate the annotation with your model. Please consider using devices with RTX3070 or better.

***If you have limited computation resources, you may down sample you slices to 128*128 and exclude slices far from objects of interest.

Responsible TA: Yixin Zhang

N1: How Much Do Augmentations Matter for Medical Image Self-Supervised Learning? (Cap 4)

- **Self-supervised learning (SSL)** is a process where the model is trained on an automatically-defined proxy task (i.e., the model self-labels images), rather than on human labels.
- Semi-supervised learning is an approach to machine learning that combines a small amount of labeled data (learned via supervision) with a large amount of unlabeled data (learned via self-supervision) during training.
- In this project, you will evaluate various SSL techniques and determine which is the best for your dataset.



N1: How Much Do Augmentations Matter for Medical Image Self-Supervised Learning?

1. Read the following self-supervised learning (SSL) technique papers, and **choose one for later implementation** https://arxiv.org/abs/2002.05709, https://arxiv.org/abs/2006.07733, https://arxiv.org/abs/1911.05722.

2. Design a CNN/encoder with 3-4 hidden layers that outputs a 32-dimensional feature vector used for contrastive learning.

3. Pre-train your model on each of three diverse **2D** MedMNIST datasets of your choice (https://github.com/MedMNIST/MedMNIST), with the **SSL training procedure of your choice.** Note that you may not be able to use the high batch sizes that some of these papers use.

4. For each dataset, use **linear evaluation** to prepare your pretrained network for classifying a labeled test set, using a small "finetuning" training set that only trains a last, appended layer in your network.

5. Evaluate your self-supervised network on a classification test set and compare to the supervised result.

6. Try pretraining your SSL model using an image augmentation set **with one augmentation removed**, for each augmentation, and for each dataset, followed by finetuned linear evaluation. Use this *ablation study* to answer the following questions:

   1. *Which augmentation, overall, was most important for self-supervised pretraining?*

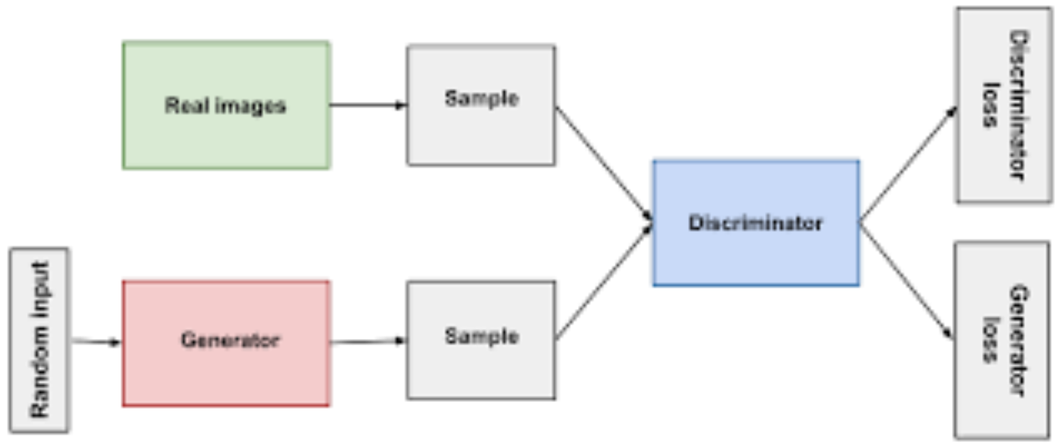
   2. *How did this differ for different dataset? Why do you think that this is the case?*

7. Note: this project will involve a lot of training, so don't save experiments for the last minute.

Responsible TA: Nick Konz

N2: Evaluating generative models for medical images (Cap 4)

- Your goal:
- (1) train a generative model on medical images and
- (2) develop medical image-specific generation fidelity metrics.



N2: Evaluating generative models for medical images (Cap 4)

1. Choose three medical image datasets from MedMNIST https://github.com/MedMNIST/MedMNIST: one full color datasets, and two greyscale datasets (e.g. radiology)
2. Implement and train a **class-conditional** GAN on each dataset.
3. Next, you need to figure out how the quality of your generated images can be evaluated. Please present the following:
    1. Visual examples of generated medical images, for each target dataset, alongside real images.
        1. Show how these both vary by class.
    2. **Develop target dataset FID scores for samples of generated images: The FID score compares the distribution of generated image features to real image features. It was originally introduced in https://arxiv.org/abs/1706.08500, equation 6, but https://arxiv.org/pdf/1706.08500.pdf may be more helpful. To implement this for your medical datasets:**
        1. Train a simple CNN encoder (e.g., resnet-18) on each medical image dataset, and save the checkpoints. You will use the features in the **final hidden layer** (i.e., before the final classification layer) to evaluate FID.
        2. To evaluate the FID of generated images, you'll need to compare the mean vectors and covariance matrices of features that the CNN extracts from real and synthetic images. Use sample sizes (for both real and generated images) that are **computationally reasonable** for your setup. Evaluate the unconditional FID (averaged over all classes) of generated images.
        3. Analyze how FID varies for different generated classes (make sure generated images are only compared to real images of the same class). Are some classes easier to generate than others?
        4. Use your FID scoring to analyze **how similar the distributions of images from different modalities are to each other.** Address if your results are reasonable.

Responsible TA: Nick Konz

G1: VAE on Medical images (Cap 3)

- Using the code from https://github.com/AntixK/PyTorch-VAE, improve the quality of the medical images generated from VAEs
  - e.g. parameter tuning, different architectures, losses …
- You should use at least 2 different VAEs, vary multiple parameters and compare their performance on these datasets. Do they produce good images?
- Mandatory dataset: https://github.com/MicheleDamian/prostate-gleason-dataset

G1: VAE on Medical images (Cap 3)

- Outline:
  - Find another two medical image datasets and repeat for those. You can try looking at medical MNIST
  - Write a discriminator to differentiate between your images and the truth, and wire those VAEs into it (turn them into GANs essentially), train and show your results.

Responsible TA: Gan Lim

Z1: Causal Discovery In Video via DNNs (Cap 4)

Objective: Identify (granger) causal relationships between objects/subjects in videos

Phase 1: Select a object-detection video dataset in which multiple objects/subjects interact, preferably in ways which include object obstruction

- Example Dataset: SportsMOT dataset (https://paperswithcode.com/dataset/sportsmot). Other resources may be found at, for example, https://www.twine.net/blog/top-object-recognition-video-datasets/ .
- Note: Students will likely need to create an augmented version of this dataset in Phase 2 to ensure object obstruction is appropriately handled

Phase 2: Implement 2 Video-based object detection algorithms that handle missing (esp. obstructed) data. These algorithms will need to create a bounding box for each object and assign a unique identifier to each box.

Phase 3: Implement 1-2 Granger causal estimation algorithms for discovering Granger causal relationships using box-center time-series extracted from videos.

- Example algorithm: cMLP / cLSTM (see https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9376668 and code at https://github.com/iancovert/Neural-GC)
- Students will need to adapt trained algorithms from Phase 2 to extract box-center time series from real data
- Example time series (toy) datasets for causal discovery: Synthetic (lorenz model), DREAM4 insilico network data (see https://www.synapse.org/#!Synapse:syn3049712/wiki/74630)

Other Notes:

- Analysis Technique(s): MSE for both networks (video object detection and causal graphs)

Responsible TA: Zac Brown

Z2: Deepfake Detection (Cap 2)

Objective: Study tools and challenges for identifying deep fakes using both classical and modern techniques.

Phase 1: Literature search for various deepfake detection methods

- Primary Papers:
  - "Deepfake Detection: A Systematic Literature Review", IEEE Access 2022, (https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9721302)
  - "Analysis Survey on Deepfake detection and Recognition with Convolutional Neural Networks", IEEE 2022, (https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9799858)
  - "Evading DeepFake Detectors via Adversarial Statistical Consistency", CVPR 2023, (https://openaccess.thecvf.com/content/CVPR2023/papers/Hou_Evading_DeepFake_Detectors_via_Adversarial_Statistical_Consistency_CVPR_2023_paper.pdf)
  - "Intriguing properties of synthetic images: from generative adversarial networks to diffusion models", CVPR 2023, (https://openaccess.thecvf.com/content/CVPR2023W/WMF/papers/Corvi_Intriguing_Properties_of_Synthetic_Images_From_Generative_Adversarial_Networks_to_CVPRW_2023_paper.pdf)

Phase 2: Train 1-2 image-generative GAN models

- Example algorithms
  - See review paper at https://link.springer.com/article/10.1007/s10462-023-10434-2
  - see following link to survey on image synthesis
  - https://www.sciencedirect.com/science/article/abs/pii/S1566253521000385?fr=RR-2&ref=pdf_download&rr=815b7212aba14263
  - Example datasets: Section 3 of https://link.springer.com/article/10.1007/s10462-023-10434-2

Phase 3: Implement 1-2 deepfake detection methods, along with at least 1 classical metric-based detection method for identifying synthetic images from Phase 2, compare results

- Deepfake detection algorithms should be selected from literature search papers
- Example classical metrics: Box-counting dimensionality, general population statistics

Responsible TA: Zac Brown