# Contrastive Learning Based MRI Segmentation Under Arbitrary Distortion

By
Santhosh Holla Prakash
Student ID : 2357503


Supervisor : Dr. Hyung Jin Chang

A thesis submitted to the University of Birmingham
For the degree of MSc in Advanced Computer Science

School of Computer Science
University of Birmingham
Birmingham, UK

September 2022

# Table of Contents

# Abstract

MRI segmentation is particularly beneficial for many clinical applications such as organ volumetric calculation, radiotherapy planning, and so on. The first and most important need for effective MRI segmentation is a clean/proper MR image free of distortions. But data corruption during MR image capture is common due to various reasons such as patient movements, metallic implants, etc. In the present methodologies, two networks are employed in two stages to handle these distorted images. Correction models were applied in the first stage to correct the corrupted image, while segmentation models are utilized in the second stage to segment the MR images. In this study, a novel optimal strategy that does both correction and segmentation in a single stage is proposed. The proposed single network is a GAN-based architecture that can segment MRI with variable distortion parameters. Furthermore, Pixel-wise contrastive learning is employed during model training for effective feature extraction. The study presents methods for generating synthetic distorted MR images, investigates several architectures relevant to the proposed network, and a thorough evaluation of the proposed network with respect to varying degrees of distortions using validation metrics such as SSIM[7], dice etc. Finally, a comparison to the traditional approach is provided, and the proposed network is claimed to perform as accurately as a conventional strategy while having less model complexities.

## Keywords

GAN (Generative Adversarial network), MRI (Magnetic Resonance Imaging), Contrastive Learning, SSIM (Structural Similarity Index Measure), Dice, Convolutional Neural Network[16], Deconvolutional Neural Network, Supervised Learning, Unsupervised Learning, Segmentation.

# Acknowledgment

I'd like to thank my supervisor, Dr. Hyung Jin Chang, for his guidance and assistance during the research. This study would not have been possible without his supervision and advice during the thesis. Special thanks to my inspector, Dr Masoumeh (Iran) Mansouri, for providing feedback and valuable suggestions. In addition, I'd like to thank my friends and family for their ongoing encouragement, support, and patience. They are my primary source of inspiration.

# CHAPTER 1: Overview

## 1.1 Introduction

The major objective of this work is to optimize the task of MRI segmentation under arbitrary distortion. In this section we will understand what is the problem that we are going to solve, why it is required to solve, and an overview of how it is solved currently and our contributions.

Understanding what arbitrary distortion is and how the distortion is introduced in MRI images in the first place is important. Basically, magnetic resonance imaging (MRI) is a non-invasive imaging technique that generates three-dimensional detailed anatomical images of organs. These images are frequently used in disease detection, diagnosis, and therapy monitoring. Since the application of these images are meant for medical examinations it's very important to capture these data in high quality standards for better accuracy. The high-quality clean MRI images are then used for medical tests by segmenting specific regions of interest using image processing techniques or deep learning models based on the application. But there are few problems that can occur in the data acquisition process and we might end up getting few distorted/corrupted images. Some of the distortions commonly observed are motion distortion, viewing distortion, missing distortion etc. Since the MR device is sensitive to motion, if the patient moves during scanning then motion distortion will be captured. Sometimes if the viewing is not proper then viewing distortion will be introduced. Also people with implants, particularly those containing iron, — pacemakers, vagus nerve stimulators will introduce the missing region distortions. Under these circumstances, with distorted images it's highly unlikely to get an accurate MRI segmentation for medical examinations.

With the recent updates in computer vision technology there are several algorithms which have been introduced to handle these kinds of distorted images. Currently the problem of MRI segmentation under arbitrary distortion, is solved in two stages as a traditional approach. Firstly, GAN based architectures are used as a correction model to translate the distorted image to distortion free images. So far these architectures are majorly focused in correction of one particular distortion but in this work, we focus on the correction of arbitrary distortions. Once the correction step is done then the corrected images are further taken for segmentation. In the segmentation stage the standard segmentation architectures such as Unet, Mask-RCNN[14], DeepLabV3[13] etc are used. By combining these cascading networks the solution is obtained for MRI segmentation under the distortion parameter.

This research work pitches in as an optimisation solution for traditional approaches by bringing down two stage solutions to single stage solutions. The proposed model takes a distorted image as an input and outputs the clean segmented image by performing simultaneous correction and segmentation tasks. The network uses GAN as a base line model to bring the segmentation solution along with integrating several other components like contrastive block and style block for better performance. Further in this report, we majorly discuss the contribution of this work like synthetic data generation module, experiments done to reproduce the results of traditional approaches, Implementation details and experiments of proposed networks and finally evaluation strategy and accuracy comparisons with traditional approaches.

## 1.2 Motivation

In this section we touch upon what made us take up this research activity.

Efficiency and performance is one of the most critical factors that matters in MRI segmentation. The distorted images were being discarded during the initial days of medical examinations. While the technology evolved, many algorithms were introduced to consider these distorted images instead of discarding. The images were corrected first via these programs for segmentation. It's impressive that even the corrupted images are now being used but the disadvantage/problem here is the complexity in the process. That is two cascading architectures are required for a solution which makes the overall structure complex in terms of space and execution time.

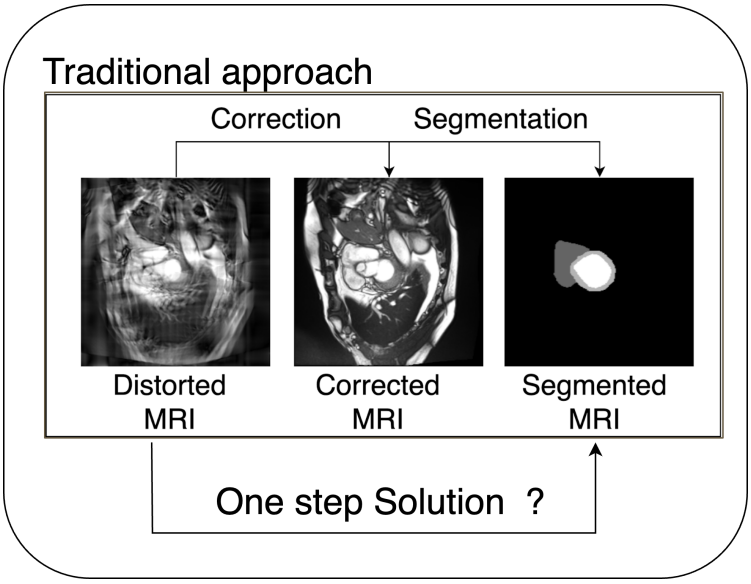


*Figure 1: Represents traditional way of solving Vs idea to solve efficiently.*

As a research question few things could be asked -

1. Can we optimize the method to a single stage which solves in lesser complexity?
2. What kind of model/algorithm is capable of doing both correction and segmentation tasks simultaneously?

Since optimizing this solution brings a breakthrough to this problem, this research activity focuses on finding the solutions for the above questions. While answering these questions this work proposes an optimal solution by proposing GAN based architecture which does both correction and segmentation simultaneously.

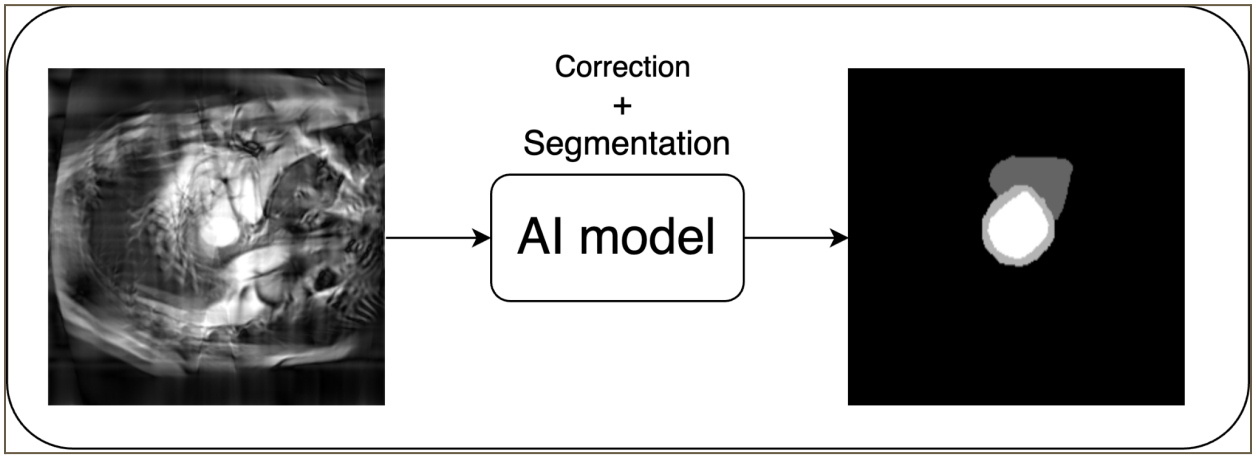


*Figure 2: Overview of a basic approach on how to solve the problem*

# CHAPTER 2: Background

In this section, few of the mathematical concepts and the deep learning model's working principle is discussed at a high level which are essential for understanding the traditional approach and proposed methodology. Some of the concepts will again be discussed in detail at particular sections. The below topics are preliminary for understanding this research activity.

## 2.1 Fourier transform

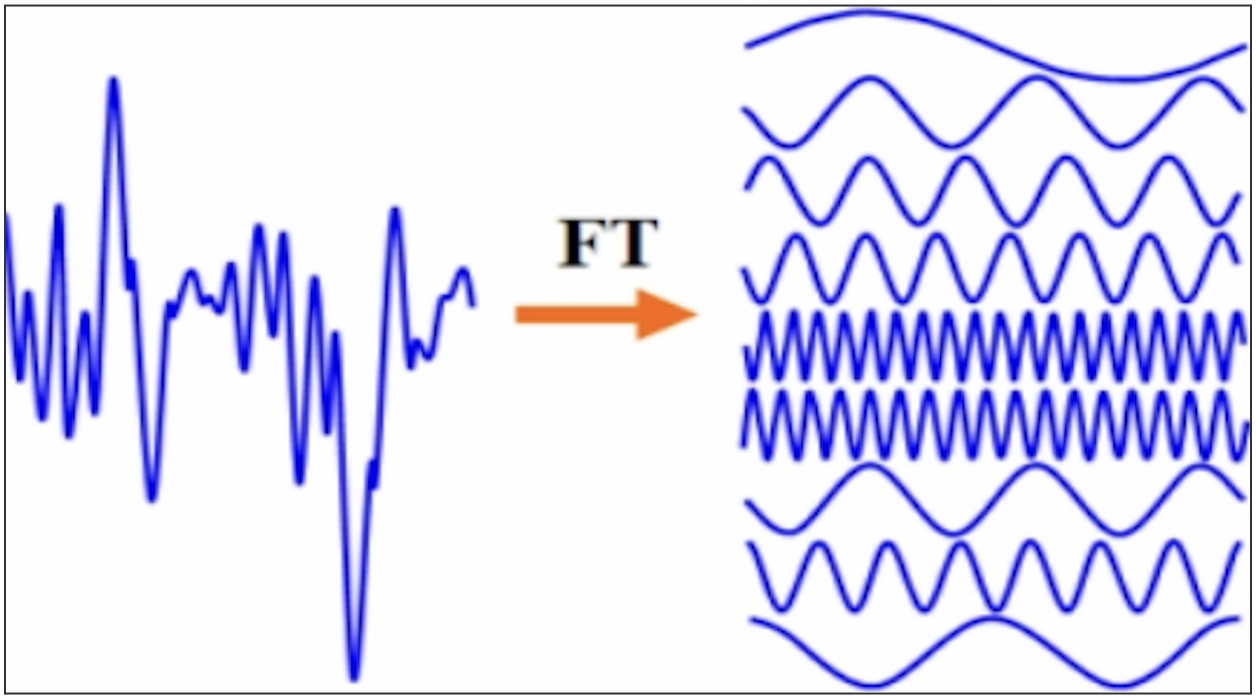


*Figure 3: Represents decomposition of complex waves to simple waves through fourier transform.*

This concept will be used in a data generation module for introducing the motion distortion in MRI. Basically the fourier transform[1] is used for decomposition of the waveform into frequency waves that make it up. The result produced by the fourier transform is a complex valued function of frequency. As we can see in the above image the complex waveform is decomposed into multiple sin/cos ways using fourier operation.

The formula for performing fourier transform is →

$$f(x) = \int_{-\infty}^{\infty} F(k)\, e^{2\pi i k x}\, dk$$

$$F(k) = \int_{-\infty}^{\infty} f(x)\, e^{-2\pi i k x}\, dx$$

Where F(k) can be obtained using the inverse Fourier transform.

Some of the standard properties of Fourier transform include (as per open source):

- *It is a linear transform* – If g(t) and h(t) are two Fourier transforms given by G(f) and H(f) respectively, then the Fourier transform of the linear combination of g and t can be easily calculated.
- *Time shift property* – The Fourier transform of g(t–a) where a is a real number that shifts the original function has the same amount of shift in the magnitude of the spectrum.
- *Modulation property* – A function is modulated by another function when it is multiplied in time.
- *Parseval's theorem* – Fourier transform is unitary, i.e The sum of squares of a function g(t) equals the sum of the squares of its Fourier transform, G(f).
- *Duality* – If g(t) has the Fourier transform G(f), then the Fourier transform of G(t) is g(-f).

Using this concept, we introduce motion distortion. That is, we convert MRI images to K-space and distort the k-space to get the motion distortion. We discuss the motion distortion in the data generation module in detail.

## 2.2 K-Space

The k-space is an expansion of the well-known Fourier space idea in MR imaging. The k-space[15] represents an object's spatial frequency information in two or three dimensions. The space spanned by the phase and frequency encoding data defines the k-space.

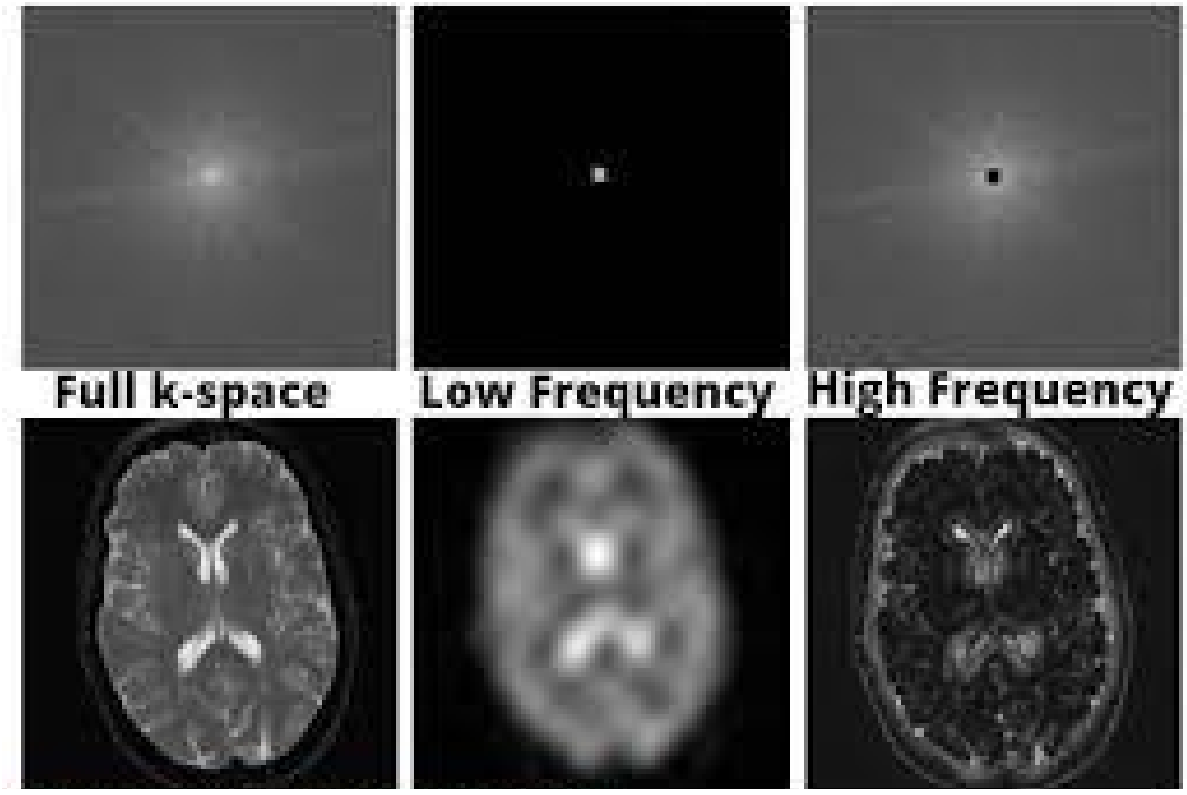


*Figure 4 : Represents how modifying certain values of k-space results in final MRI creation*

In simple words, while data acquisition, the MRI device employs and transmits magnetic and radiofrequency waves into the patient's body to capture an image. Because of these radiations the molecules in the organs start spinning in a particular direction. Magnetic field generated due to the spin will be captured by the sensors of the MR device. These signals are saved in the form of K-Space using phase encoding and frequency encoding techniques. To obtain the MRI image from this raw format (K-space) inverse fourier transform will be used.

This technique is being used in our work to introduce motion distortion. In our work, we access this k-space and corrupt the information related to spatial frequency. By this we could introduce the motion distortion for our work.

## 2.3 Image translation

Image translation is converting/translating the image of one particular domain into another domain. For example, an image of an MRI translating to the image of PET-CT where the translated image looks like a realistic image. GAN and VAE are commonly known methods for performing the image to image translation. There are two types of

image translation - paired and unpaired. Paired translation is supervised learning where there is mapping from one to one, whereas unpaired is unsupervised learning. In this work we will be designing our proposed model using GAN as a baseline architecture so we will understand its working principle.

### 2.3.1 GAN (Generative Adversarial Networks)

Here we speak about the GAN architecture in terms of image translation rather than image generation. The difference between generation and translation is that generation is generating the new image of a particular domain by taking noise vectors as input, whereas translation is converting one form of image to another. Hence translation is also often referred to as a conditional adversarial network since it is conditioned on input image.

Basically GAN architecture has 2 major components such as generator and discriminator . A generator block encompases encoder and decoder modules to perform the image translation task. Encoder could be of any backbone CNN network such as resnet, inception, efficient net and so on. In GAN, the responsibility of the encoder is to extract the important features of an input image. Then the decoder will be used to perform deconvolution operation on extracted feature maps to output the final resultant image. Basically, the decoder is responsible for upsampling the feature maps to obtain a translated output image. The objective of the discriminator is to detect whether the incoming image is fake (generated image) or real (ground truth).

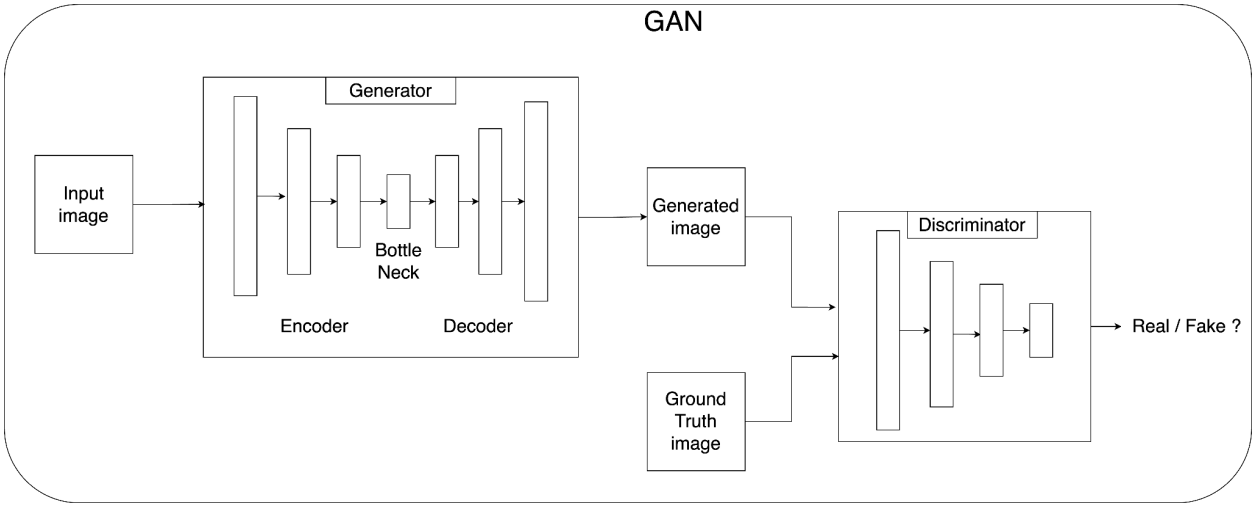


*Figure 5: A high level block diagram of GAN architecture for image translation task*

With the generator module *G* and given input image *x* and random noise *z*, the objective of the generator is to translate *x* to *y*. i.e, $G : \{ x, z \} \rightarrow y$. The generator *G* is trained to produce outputs that cannot be distinguished from “real” images by an adversarially trained discriminator, *D*, which is trained to do as well as possible at detecting the generator’s “fakes”. It is a MinMax problem where the generator is trying to optimize its parameter to output the image that fools the discriminator. Whereas, the discriminator is learning to differentiate between fake and real effectively. The overall objective function is described as :

$$\mathcal{L}_{GAN}(G, D) = \mathbb{E}_y[\log D(y)] + \mathbb{E}_{x,z}[\log(1 - D(G(x, z))].$$

where *G* tries to minimize this objective against an adversarial *D* that tries to maximize it, i.e. $G^* = arg\, min_G max_D L_{cgan}(G, D)$

## 2.4 Contrastive learning

This topic is a recent advancement for efficient extraction of features in convolutional neural networks. The applications of contrastive learning is efficient feature extraction for better classification, building pre-trained models for transfer learning etc. The objective of this module is to extract the feature such that, same class objects will cluster in compact and fall apart from other class clusters in feature embedding space. This helps the model for getting overall improvement in performance. There are two types here - supervised and unsupervised. Some of the well known unsupervised contrastive methods are Simclr[17], MoCo[18] etc.The Unsupervised is mostly used for transfer learning / few shot learning. In unsupervised, the workflow is, the image is taken and its augmented version is considered to be the positive sample and rest other images are considered to be negative sample. Now the objective is to learn the feature such that all positive samples will cluster compactly and separated from negative samples. But for our work we use a supervised version of contrastive learning. Here we dont use it as transfer learning, rather we use contrastive blocks with contrastive loss while training the model for its objective. In supervised, we know the label of each image so images with the same class belong to the positive group and the images with different classes belong to the negative group. On the details of how it is implemented and how loss will be calculated, we can understand in the methodology section. Just to know what will be the result of contrastive learning, we can see the below image.

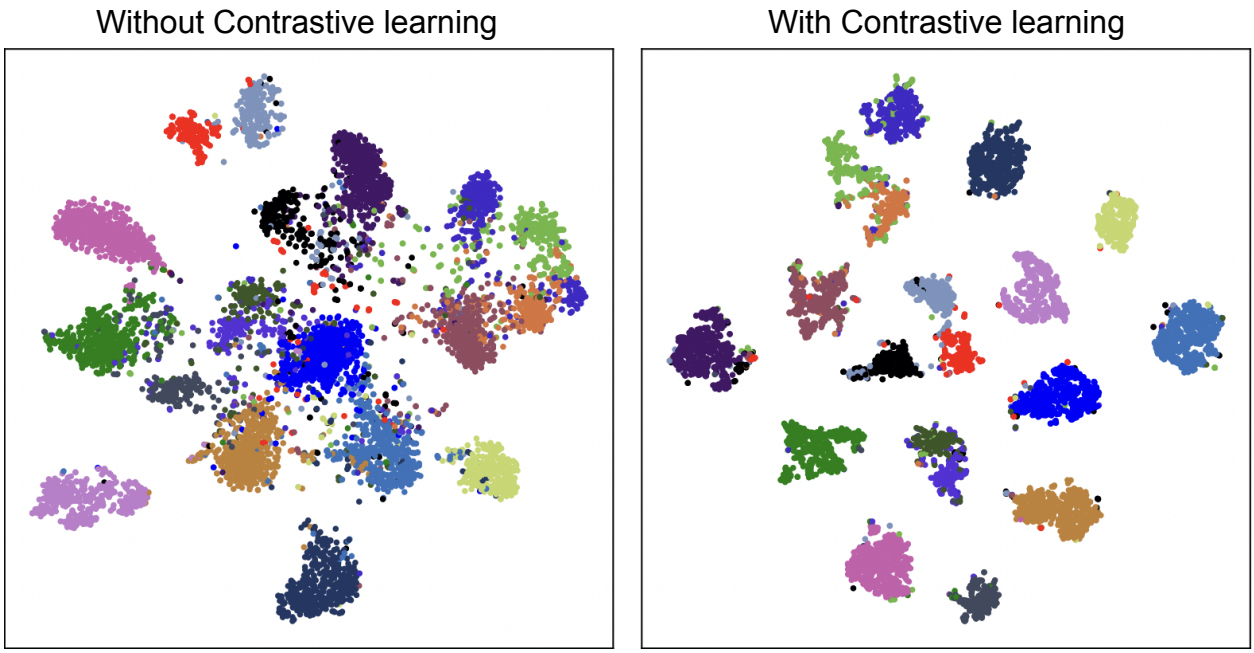


*Figure 6 : An example collected from open source to present how feature space gets affected while using contrastive learning and while not using contrastive learning.*

# CHAPTER 3: Related work and literature review

In this section we will discuss some of the prior work that has been done to solve the problem of MRI segmentation under arbitrary distortion. This section is divided into two sub-section because in the traditional approach the problem is being solved in two stages - correction and segmentation, so we will discuss the related work relevant to the particular stage. Also through the literature review we will understand why all the state of the art solutions use the deep learning approach to solve the problem. The objective of this section is to understand the current approaches, and identify the problems in the current method and come up with novel solution methods.

## 3.1 Correction stage

To do the distortion correction some of the existing architectures are GAN, Med-GAN, IPA-MedGAN etc. Most of the related work prefers to use the GAN based architecture to solve the problem to get the efficient results. Normal image processing algorithms or other linear models are not capable of solving this issue because a lot of nonlinearity will be present in corrupted images. If there is a shift in one direction or blur in some linear passion, then just using the image registration process or de-noising would have resulted in getting back the proper image. But when we see the distortion like ghosting and motion distortion, the pixel values are not shifted/impacted linearly and to correct those we need non-linear models. Furthermore, to get better results on the correction, both local features and global features matter a lot to reconstruct back. To learn these complex features and to perform the correction operation we need a deep learning model.

GAN is majorly used because of its translation capability. This problem is treated as image to image translation where distorted image is being translated to corrected image. From the paper [2] “Respiratory Motion Correction in Abdominal MRI using a Densely Connected U-Net with GAN-guided Training “ through the experimental results, the author has explained that the segmenter model is only capable of removing bulk ghosting outside of the central area of images. However, the organ details are still blobby and blurry. So it is important to train a model in GAN mode such that the correction can be done accurately. Now, let us see a couple of architecture’s working principles that performs the correction on distorted images.

### *3.1.1 MED-GAN*

In 2019, A paper[3] called “Medical Image Translation using GANs” was published addressing the problem of image to image translation in the medical domain. The work has 3 major objectives and one of them is motion correction in MRI images. This work has addressed the issue of just one type of distortion correction that is motion distortion. The architecture uses cascading Unet blocks as a generator model which encompases 3 Unet architecures. The output of one Unet is fed to the following Unet as input. Since the medical image translation poses several challenges in translation tasks, compared to other image domains, this work utilizes the casnet method to solve robustly. Along with the generator network, Med-GAN has a discriminator network and pretrained model. The responsibility of the discriminator is to tell whether the input image is real or fake. To achieve this the network is trained along with the generator with adversarial loss. And to extract the rich feature in Unet encoder, the style and content loss is calculated using vgg pretrained network. Using these 3 components (generator, discriminator and the pretrained network) the model was able to achieve 0.83 SSIM score in doing correction.

### *3.1.2 IPA-MED-GAN*

In 2020, the improved version of med-gan was published addressing the different distortion setting. Med-GAN addressed the motion distortion, but this work addressed the missing region distortion. In this work[4], a slight change in the architecture is done compared to MED-GAN, that is, instead of using a pretrained network, this work uses two discriminators. One discriminator is used to get global adversarial loss and the other to get the local. Which means, patch discriminator is used such that the adversarial loss is calculated on a 70x70 patch region where the final loss will be the sum of all patches. By doing this, the discriminator can’t be fooled by class imbalance or other such issues because it looks at each local region to tell whether it's real or fake instead of looking at an image as a whole. Also another modification is, the input to the discriminators is not only either generated image or fake, instead its the concatenation of generated image with generator input or ground truth with the generator input. By doing this, discriminator can learn the features with respect to translations which helps in getting better translation results from segmenter. This work has claimed to obtain a 0.98 SSIM score for arbitrary missing patch distortion correction.

## 3.2 Segmentation

Once the correction step is done, the next step in the traditional approach is segmentation. The corrected images will be the input for the segmentation models to perform the segmentation. Some of the common segmentation architecture is Unet, DeeplabV3, MaskRCNN etc. we shall discuss about the Unet architecture since we use that model as a segmenter network in our proposed architecture.

### *3.2.1 Unet*

The Unet paper[5] was published in 2015 to perform biomedical image segmentation efficiently. U-Net is an architecture for semantic segmentation. The basic Unet architecture consists of a contracting path and an expansive path called encoder and decoder. The contracting path follows the typical architecture of a convolutional network. It consists of the repeated application of two 3x3 convolutions (unpadded convolutions), each followed by a rectified linear unit (ReLU) and a 2x2 max pooling operation with stride 2 for downsampling. At each downsampling step we double the number of feature channels. Every step in the expansive path begins with an upsampling of the feature map, followed by a 2x2 convolution ("up-convolution") that cuts the number of feature channels in half, a concatenation with the correspondingly cropped feature map from the contracting path, and two 3x3 convolutions, each followed by a ReLU. Because of the loss of boundary pixels in each convolution, cropping is required. A 1x1 convolution is employed at the final layer to transfer each 64-component feature vector to the desired number of classes. The network comprises a total of 23 convolutional layers. Given the corrected MRI image of dimension 256x256x1, the output of the Unet architecture would be 256x256xn where n will be the number of classes that needs to be categorized. Based on the application, in MRI segmentation the average accuracy of Unet is claimed to have around 87% dice score.

***Note:***

However, from the literature review what we can understand is that, so far the problem of MRI segmentation under arbitrary distortion is solved in two stages. But the disadvantage is no one has attempted to solve it in a single stage. Moreover, correction models address one particular type of distortion and none of them address all distortion settings simultaneously. Also, with the recent advancement in the feature extraction process there are several methods of contrastive learning which we can utilize for getting better results. So our work will be focusing on obtaining the solution in a single stage, for arbitrary distortion settings using contrastive learning for getting better accuracy with less complexity.

# CHAPTER 4: Methodology

## 4.1 Overview

An overview of the proposed network for performing the MRI segmentation under arbitrary distortion as a single stage solution is presented in below fig 10. Before moving to the architecture design we will understand the synthetic data generation part which is also a major contribution in this work. Overall in this section we will discuss the data generation module, loss components, proposed architecture design and its working principle in detail.

## 4.2 Data generation module

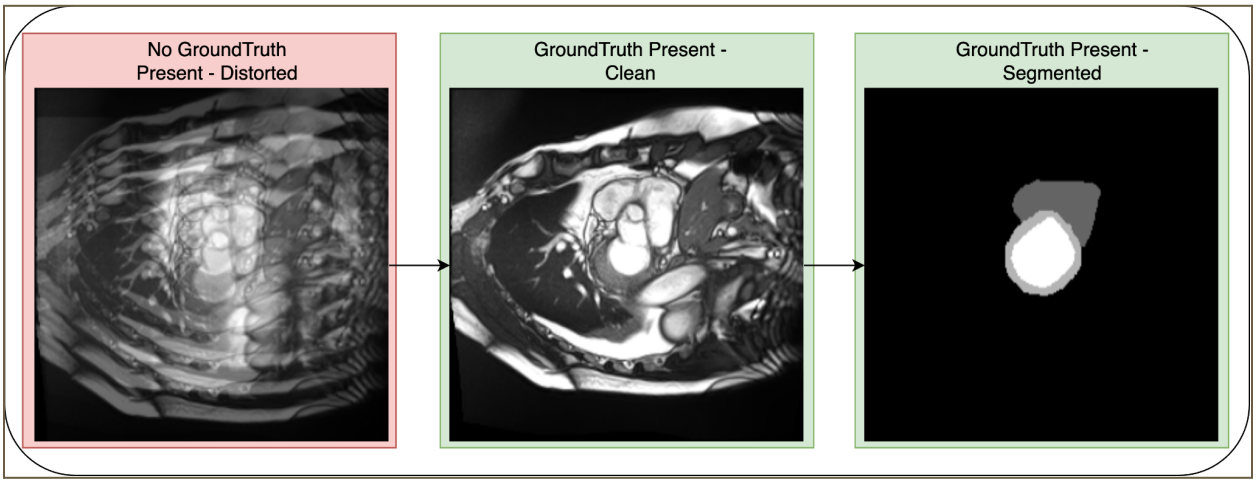


*Figure 7: Ideal combination of datasets. But we don't have distortion images in open source.*

The ideal data set for this research work is the combination of distorted image, clean image and the ground truth segmentation image. Since there is no publicly available dataset which has these three combinations, it is necessary to generate the synthetic distorted images. In this work 6 different distortion settings were implemented which led to generation of several distortion images with varying degrees of distortion. The various distortions are introduced to MR images as presented in the below fig.

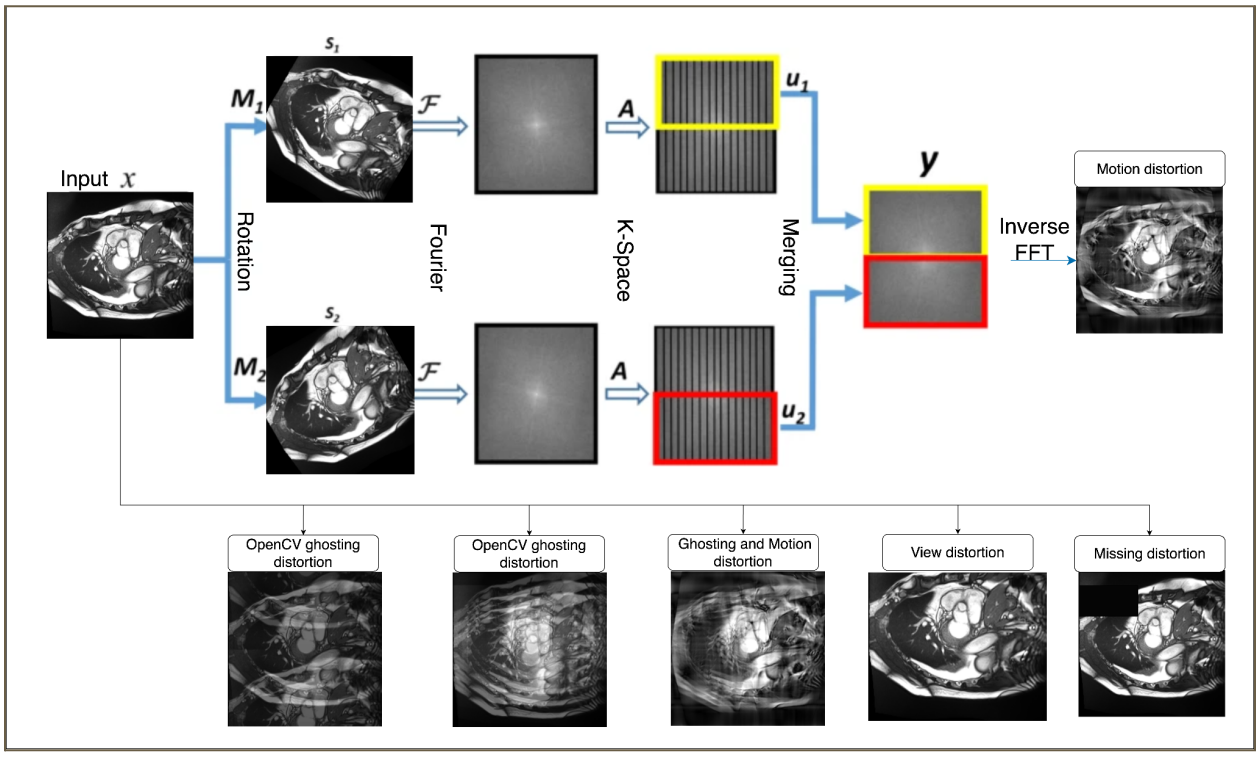


*Figure 8 : An overview of how distortion is introduced in MRI using various techniques.*

### 4.2.1 Motion distortion

The motion distortion[8] is introduced for MRI in following steps. The image x is first rotated with a variable degree of angle to get two versions of augmented images s1 and s2. These two images, s1 and s2, are then converted to K-space by obtaining their spatial and frequency information via fourier transform. A segment of k space which is obtained from image s1 is merged with the k-space of s2. This way the distortion is introduced because in the real time as well when the patient does motion, the k-space information is the one which will be distorted. Then the distorted K-space is converted back to the MR image using inverse fourier transform. This approach is chosen because the introduced distortion should represent a realistic distorted image. And since we have access to k-space and our final distorted image is built by corrupting the k-space, it brings solid evidence for the assumption that synthetic data will look like real time data.

### 4.2.2 Missing patch distortion

Local deformations in medical modalities are common phenomena due to a multitude of factors such as metallic implants in magnetic resonance imaging (MRI). Due to these implants sometimes the metals come as a missing region in the MRI which

makes the segmentation task difficult. In our work to simulate this scenario, we use a [5x5] or [10x10] or [15x15] window which randomly chooses the image location to generate a black patch on it. In this work we fix the constant size of the square black patch. These black patches on the MRI image look close enough to the missing patch distortion images that come in real world data.

### *4.2.3 More ghosting distortion*

Ghosting is a sort of organized noise that appears in the image as repeating versions of the main object. They happen as a result of signal instabilities between pulse cycle repetitions. Ghosts are typically seen along the phase encode direction and are blurred, smeared, and shifted. To simulate the higher degree of ghosting the following steps are done. A copy of an image x is obtained as image y and z. Then y is shifted upwards 50% from the original image and z is shifted downwards by 50% from original image. Now the shifted y and z are merged with image x with some weight value (In our work we use 80% original and 20% shift) to obtain the final more ghosting distorted image.

### *4.2.4 Viewing angle distortion*

Due to limited view during the data acquisition some MRI images come with viewing distortion. To simulate those scenarios, an MRI image is cropped for 10 to 40 pixels in all directions randomly.

### *4.2.5 Less ghosting distortion*

This distortion is similar to the more ghosting distortion. The difference is that instead of shifting 50% upwards and downwards, the images shifted 10%. These distortion settings simulate the local deformations that occur during MR scans due to signal instabilities.

### *4.2.6 Motion-Ghosting distortion*

This distortion setting simulates both the patient movement and unstable signals captured during the MRI scanning. To generate this distortion the image x is first divided into 4 grids and less ghosting is done on randomly picked 2 grids. Once the less ghosting distortion is introduced on a local region, then motion distortion is introduced via k-space, fourier transform and inverse fourier transform method.

The final combinations of distorted images that could be generated is presented in below images. These are few samples, but we can generate many more representations.

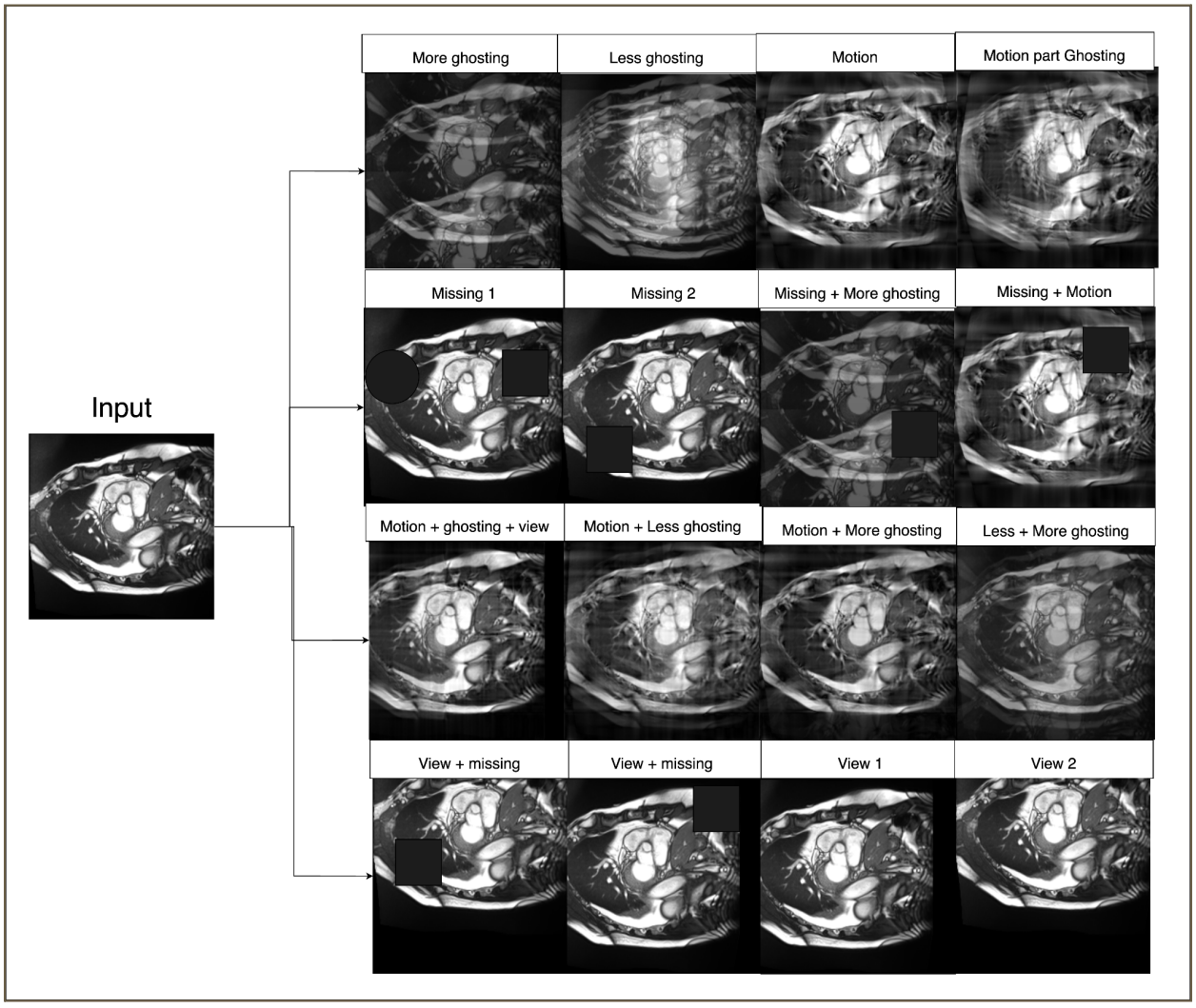


*Figure 9: Few samples of distorted images which are generated by combining different methods.*

## 4.3 Architecture design

The major objective of our proposed network is to perform MRI segmentation under arbitrary distortion as accurately as the traditional/conventional approaches with lesser complexities. The designed network is focusing to meet that objective. The proposed network has majorly 4 components - segmenter, discriminator, pre-trained network and the contrastive block. Each component has its own role to play in bringing the overall performance. Also, one thing to notice is that all 4 components are important during the training of architecture, but only segmenter will be used during the inference time. There will be no contribution from discriminator or pretrained network or contrastive block during inference of the model. This is the major reason for achieving the lesser complexity during the inference mode.

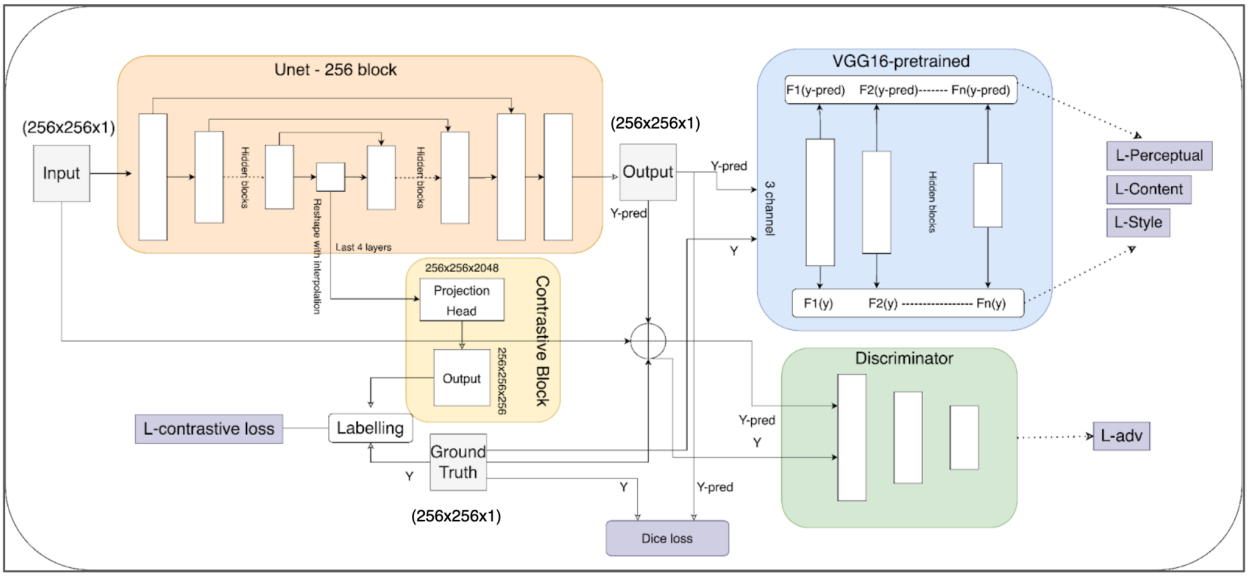


*Figure 10 : Block diagram illustrating the overall components and structure of the proposed network.*

The design encompases VGG-16 as pretrained model, Unet-256 as segmenter, Custom CNN networks as discriminator and contrastive block. The overall workflow of this architecture is, The segmenter will take a distorted image as an input and does the segmentation and correction step and finally outputs the target segmented image. The other blocks are present to make the segmenter to achieve its objective efficiently. In our work we demonstrate the segmentation task on the ACDC dataset, which has 4 regions to be segmented (details of the channel and classes of datasets will be discussed in the dataset section). We shall understand each block by answering these questions - what is that component ?Why is it needed?And how is it constructed?.

### *4.3.1 Segmenter*

Segmenter architecture is the major component of our work. This network is majorly responsible for performing both segmentation and correction operations simultaneously. We use a Unet 256 as our baseline model as segmenter. The Unet-256 network has 2 subcomponents - encoder and decoder. Encoder is constructed with a series of convolutional neural network layers which downsamples the original input image by extracting the necessary features. These features are extracted based on the objective that is meant to be achieved. Once the features are extracted the upsampling is done using the deconvolutional operation to obtain the target image. Our baseline model must identify a straight translation from 2D grayscale MRI image to segmentation picture space. However, with the encoder-decoder architecture, there is an information bottleneck that hinders the flow of low level information in the network. To address this issue, features from the contracting path are concatenated with the upsampled output from the network's expanding path. This skip connection helps the network to learn the spatial information during the upsampling operations.

The Unet 256 model is symmetric which has 7 units of CNN layers in encoder and 7 layers of deconv layers in decoders. Each layer has 4x4 filters with stride value 2. During the contraction the depth will be doubled at each layer. At every layer of encoder, after convolution operation, batch norm is performed and following that Leaky relu activation is used with a slope of 0.2. And at every layer of decoder after decovn operation the depth will be reduced by half, then the mirroring layers of contracting path’s feature map is concatenated. Then batch normalization is done followed by relu activation function. At the final layer, 1 × 1 convolution which is equivalent to cross-channel parametric pooling layer is performed to bring the output segmentation result.

### *4.3.2 Discriminator*

Discriminator is a small encoder network which takes the input of image generated by the generator and ground truth (concatenation with input) and tells whether it's real or fake. The output of the discriminator 0 always indicates that the incoming input image is fake and 1 indicates that the incoming image is real/ground truth. This CNN network is majorly used because it acts as a self-loss function where it extracts the necessary feature to differentiate between the real and fake images. This network is trained in parallel to the training of the segmentation network. Just with the L1/dice loss segmentation Unet architecture will be able to do the segmentation task accurately, but through the experimental observations we found that it is important to

use the adversarial loss for performing the correction step. The network has 3 layers of CNN, where batch norm and leaky relu is used after conv operation. At the final layer, based on the patch region,sigmod is used to give binary output to tell whether it's real or fake.

### *4.3.3 Pretrained network*

Vgg-16 is used as a pretrained network in our design. This pretrained network is trained on imagenet dataset which has around 14 million images with 22,000 classes. The purpose of using this network is to calculate the perceptual, style and content loss at the intermediate layer of CNN between the features of ground truth and the segmenter output image. Using a pertained model and utilizing these losses in training will enable the segmenter to extract deep rich features such that the resultant image will be equivalent to target in terms of style and content.

The structure of Vgg-16 pretrained network is, it has 13 convolutional layers, 5 max-pooling layers, and 3 fully connected layers. In our work we eliminate the dense connected layers and make use of only the CNN network. Since the model takes the 3 channel input, we reshape our 4 channel output to single channel and then to 3 channel. Number of filters in the first block is 64, then this number is doubled in the later blocks until it reaches 512. In this network relu activation function is used and max pooling is performed in the pooling layer. With this we calculate the style loss, perceptual loss and the content loss between the ground truth image features and segmenter output image features.

### *4.3.4 Contrastive Block*

The contrastive block[9] is constructed with a small CNN network which has just one convolution layer with filter size 1x1. The number of filters used in this layer is 256 to get the final output image dimension 256x256x256. There is no activation function or normalization or pooling operations performed after the convolution operation. The purpose of using the contrastive block is to extract the features efficiently in the Unet encoder. Because the accuracy of the segmentation completely depends on the efficiency of the feature extraction in the encoder. In this work, we adopt a supervised method of contrastive learning that is pixel wise contrastive learning approach. Since we know the ground truth label and the operations are done at pixel level, we call it pixel wise contrastive.

The working principle of this block is, the feature maps are taken from the encoder of Unet architecture and perform the interpolation operation to equivalent their dimension (in terms of height and width) to ground truth image. Then these feature maps are processed through the projection head/contrastive block to reduce the depth of the feature map. Finally labeling is done with respect to ground truth image to calculate the contrastive loss.

In traditional contrastive learning approaches, the final layer encoder's output is taken and interpolation is done so that feature map obtained will have the dimension equal to ground truth image dimension. In our case we do not take the last layer feature map only for contrastive loss calculation. Instead we take the last 4 layers of information from the encoder to perform better. Because the output of the encoder at the bottleneck layer is just 2x2 in height and width which might not contain all the necessary information for our problem. So in this work, we take a bottleneck layer feature map 'i' and do the interpolation to obtain a 4x4 dimension which is equivalent to the previous layer's (i-1) height and width. In the next steps, we concatenate the interpolated feature map to the feature map present at (i-1) and do the interpolation to get the height and width dimension of (i-2) feature map (i.e 8x8). We do the same steps until our feature map reaches the dimension 16x16 (i.e at the 4th layer from reverse). For every interpolation and concatenation, the dimension of our feature map increases as a result of the 16x16x2048 feature map. Then finally the feature map is obtained through the last round of interpolation to obtain a dimension of 256x256x2048 (where the height and width is equal to ground truth).

The depth of 2048 is then reduced to 256 via contrastive block. Here each pixel on the feature map (1x1x256) is a feature representation corresponding to each pixel on the ground truth image. Based on the ground truth, each pixel on feature maps is assigned with the labels. Then using the objective function (i,e contrastive loss function), we will be able to achieve efficient feature extraction in Unet encoder. Efficient means the same class feature is grouped closely in feature embedding space and that falls apart from the cluster of other class features. It's also called intra-class compactness and inter-class dispersion.

## 4.4 Loss components

To train our model we used majorly 6 different loss functions - Dice loss, Adversarial loss, Style loss, content loss, perceptual loss and contrastive loss. Each of these objective functions are contributing to our final results. We shall understand how each loss function is used and help us in obtaining the final results.

### *4.4.1 Segmentation loss*

Although all the loss function is contributing for final segmentation results, we use Dice loss as a basic segmentation loss. The loss function works on the basis of dice coefficient. Function measures the overlapping between the groundtruth mask and the output mask. It works similar to IOU but the difference is, in IOU it is an intersection over union, whereas in dice its, twice the intersection divided by total area. Also this function helps the model to train better when there is a class imbalance issue because it makes use of precision and recall. Basically, dice loss is F1, where its harmonic mean of precision and recall. This loss function is basically used to calculate the segmentation difference between segmenter output and the ground truth segmentation.

The dice loss (has to minimize the loss) is calculated using the formula →

$$Dice\ loss\ =\ 1\ -\ (\ \ (2\ *\ (pred\ \cap\ ground\ truth)\ )\ /\ (pred\ \cap\ ground\ truth)\ )$$

### *4.4.2 Adversarial loss*

The loss is MinMax optimization, where the generator is trying to minimize the loss, and the discriminator is trying to maximize the loss of the segmenter. This loss function working principle is the same as we discussed in the image translation section (in background). We train both discriminator and segmenter using this loss. The objective is to make segmenter to output the segmented image such that it looks realistic where dicriminator can be fooled. Where the discriminator is trying to learn the difference between fake and real. This loss is essential to perform the correction step in the segmenter. The loss function is defined as L-adv / L-GAN,

$$\mathcal{L}_{GAN}(G, D) = \mathbb{E}_y[\log D(y)] + \mathbb{E}_{x,z}[\log(1 - D(G(x, z))].$$

Where G is segmeter and D is discriminator. Considering our work, y is actual ground-truth segmentation and x is a distorted image with z noise.

### *4.4.3 Non- Adversarial loss*

Here we use style loss, content loss and the perceptual loss to bring the segmentation results close to the ground truth in terms of content and style. The content and the perpetual loss works almost the same. In fact we can ignore either one of them. The only difference is, perceptual is L1 difference and content is L2 difference between the features. And to calculate the style loss, the feature maps are converted to gram

matrices and MSE/L2 difference is taken between them. The content, perceptual and style loss is given by -

$$L_{content} = \sum_{i=0}^{n} \lambda \left| F_i(y_{pred}) - F_i(y) \right|$$

where λ is weight associated with this loss.

$$L_{perceptual} = \sum_{i=0}^{n} \lambda \left( F_i(y_{pred}) - F_i(y) \right)^2$$

where λ is weight associated with this loss.

$$L_{style} = \sum_{i=0}^{n} \lambda / 4d_i^2 * \left( Gr_i(y_{pred}) - Gr_i(y) \right)^2$$

where λ is weight associated with this loss.

### *4.4.4 Contrastive loss*

In this work we use pixel wise contrastive loss, the pixels belonging to the same class would be the positive sample and rest will be the negative sample. This loss function helps in getting the better feature extraction from Unet encoder. The loss will be used to train the Unet encoder. The loss function is given by :

$$\mathcal{L}_i^{\text{NCE}} = \frac{1}{|\mathcal{P}_i|} \sum_{\boldsymbol{i}^+ \in \mathcal{P}_i} -\log \frac{\exp(\boldsymbol{i} \cdot \boldsymbol{i}^+ / \tau)}{\exp(\boldsymbol{i} \cdot \boldsymbol{i}^+ / \tau) + \sum_{\boldsymbol{i}^- \in \mathcal{N}_i} \exp(\boldsymbol{i} \cdot \boldsymbol{i}^- / \tau)}$$

Where i is a voxel in a feature map which belongs to a particular class. $P_i$ $and$ $N_i$ denote pixel embedding collections of the positive and negative samples in that particular feature map.

***Summary :***

The overall loss is weighted sum of all the loss component that is :

$$overall\ loss = (\lambda 1 * DiceLoss) + (\lambda 2 * L - adv) + (\lambda 3 * L - perceptual) + (\lambda 4 * L - content) + (\lambda 5 * L - style) + (\lambda 6 * L - contrastive)$$

# CHAPTER 5: Experimental evaluation

In this section we will discuss in details regarding - reproducing the traditional results by exploring a couple of correction models and segmentation models, experiments done with respect to the proposed network and ablation study to understand how each component affects the final outcome.

## 5.1 Traditional Approach experiments

Here we discuss the work done with respect to reproducing the traditional approach. As we know, the traditional approach has 2 major steps - correction and segmentation. In our work, we experimented on three architectures - pix2pix[10], cycleGan[11], cycleMedGan[12]. The reason for experimenting on these networks is because pix2pix and cycleGan were the most commonly used for correction before any state-of -the-art architectures published like MedGan. And from the state of the art architecture, cycleMedGan is implemented and experimented, which is an unsupervised version of MedGan to get the correction results.

### *5.1.1 Correction Model*

We shall understand this section by answering what is the basic workflow?How are these networks constructed? And what are final specs/hyperparameters used to obtain the results.

#### *5.1.1.1 Pix2Pix*

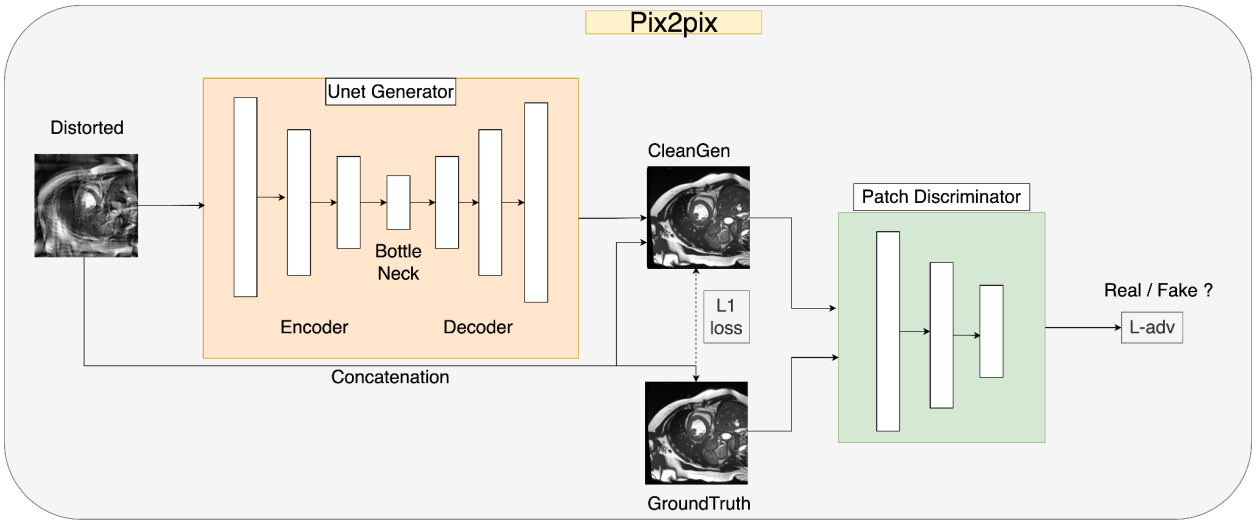


*Figure 11 : An overview of pix2pix architecture*

Pix2Pix is a Generative Adversarial Network, or GAN, model designed for general purpose image-to-image translation. The network works in a supervised fashion where the image translation is done between the paired set of images between two domains. Relating to our work, the image translation is from distorted image to clean image. The approach was presented by Phillip Isola, et al. in their 2016 paper titled “Image-to-Image Translation with Conditional Adversarial Networks” and presented at CVPR in 2017.

The architecture encompases generator and discriminator network, where in our work we experimented with resnet-9 blocks, Unet-128 and unet-256 encoder-decoder as our generatore module and three layer convolutional neural network encoder as discriminator module. The workflow of the network is, the generator takes a single channel distorted image (considering our work) as an input and gives a single channel corrected image as an output. The L1 loss is calculated between the generated image (corrected) and the ground truth (clean) image to get the reconstruction loss. Then two sets of images will be given as an input to the discriminator - ground-truth concatenated to input (real) and generated is concatenated to input (fake). Now the objective of the discriminator is to tell whether the incoming image is real or fake. This network is trained along with the generator module. The adversarial loss is calculated considering the 70x70 pixels patch. The objective of the discriminator is to tell whether each patch is real/fake. The overall loss for this architecture is, weighted sum of L1 loss and adversarial loss where the parameters of the discriminators are tuned by calculating the gradient on the basis of adversarial loss and generator on the basis of L1 + adversarial loss.

The architecture is trained with hyperparameters - Unet-256 generator module, learning rate 0.0002, batch size 4, 1300 number of epochs, normal weight initialisation with mean 0 and standard deviation 0.02. During the inference only the Unet-256 is used to get the correction results. With this environment, we were able to train the pix2pix model to better perform the correction step.

#### *5.1.1.2 CycleGan*

The Cycle Generative Adversarial Network, or CycleGAN, is an approach to training a deep convolutional neural network for image-to-image translation tasks. The Network learns mapping between input and output images using unpaired dataset. That is, this architecture works in unsupervised fashion where there is no mapping between the input and output.

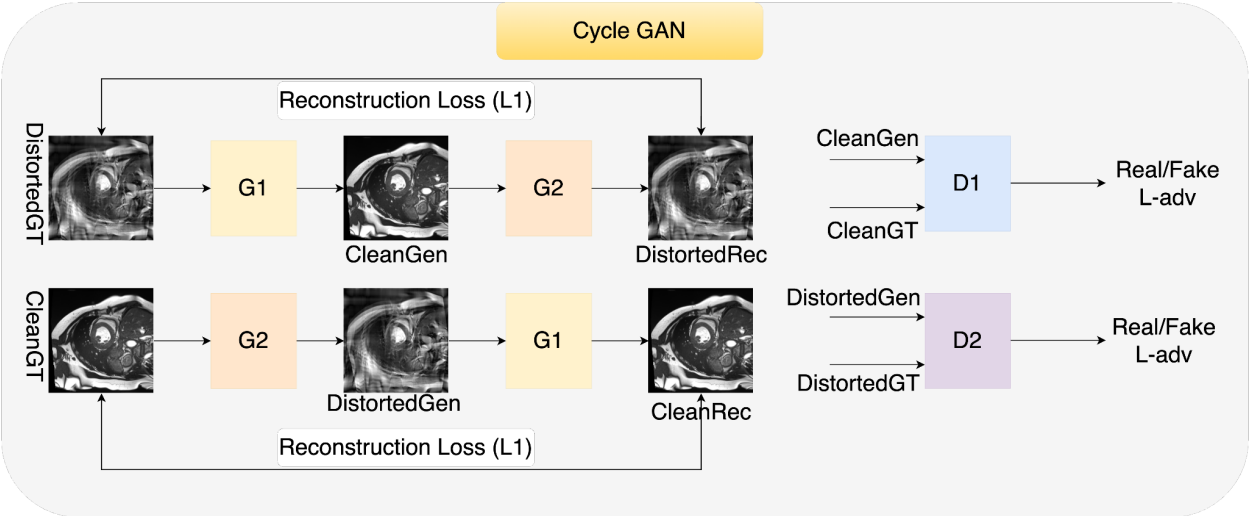


*Figure 12: An overview of cycleGAN architecture*

The architecture encompases 2 generator modules and 2 discriminator modules. Generator 1 is responsible for translating the image from domain1 (distortion) to domain2 (clean) where generator 2 does the opposite. Also there are 2 discriminators, where discriminator 1 takes domain 1 input and tells whether it's real or generated, discriminator 2 takes domain 2 as input and tells whether it's real or generated. Considering our work, the overall workflow is, a distorted image will be given as input to generator 1 and generator 1 will do the correction step. Then the generator 2 will take a corrected image and translate it back to a distorted image. Reconstruction loss is calculated between input distorted image and reconstructed distortion image. In parallel to this, generator 2 takes a clean MRI image and translates it to the distorted image and generator 1 takes the output of generator 2 and translates it back to a clean image. Along with this, discriminator 1 takes ground-truth clean image and generates clean image by generator 1 and differentiate between real and fake, whereas discriminator 2 takes ground-truth distorted image and generated distorted image by generator 2 and differentiate between real and fake. In the deployment only generator 1 is used. The total loss function for this architecture is the weighted sum of reconstruction loss and adversarial loss.

The architecture is trained with hyperparameters - resnet9 as both generator modules, learning rate 0.0002, batch size 4, 2700 number of epochs, normal weight initialisation with mean 0 and standard deviation 0.02. During the inference only the Unet-128 (generator 1) is used to get the correction results. With this environment, we were able to train the cycleGan model to better perform the correction step.

### *5.1.1.3 CycleMedGan*

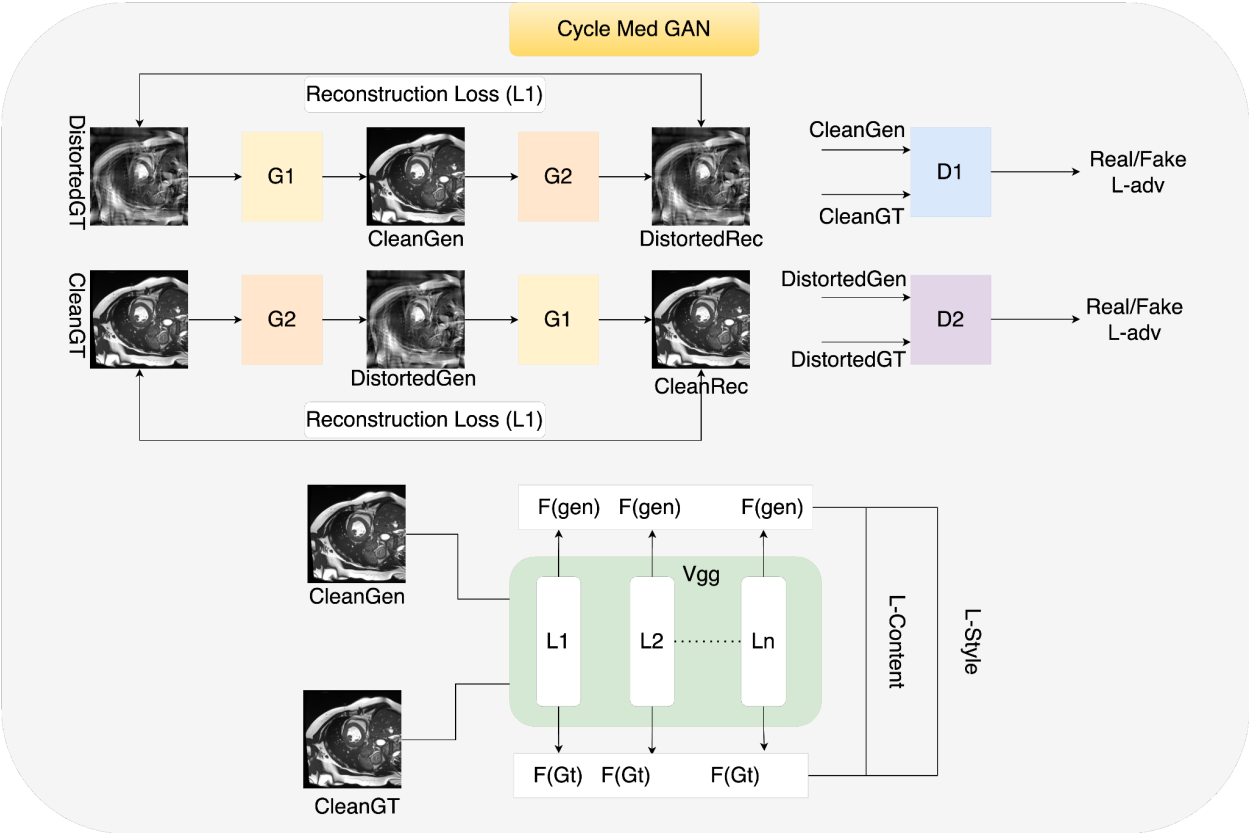


*Figure 13: An overview of cycleMedGAN architecture*

This architecture works similar to the cycleGan but slightly modified in its architecture to perform well for medical applications. The paper is published as an unsupervised version of a medGAN, where image to image translation is done with unpaired set of data between 2 domains.

Similar to cycleGAN, this architecture also has two generators and the two discriminators with the same working principle and objective. The additional component in this network is the pretrained model which is used to calculate the non-adversarial loss. The objective of using the pretrained model is that the generator learns the rich and dense texture level feature which helps in getting better translation results. Hence, this architecture used vgg-19 as a pre-trained network for calculating style and perceptual loss. The objective is to make the generator to generate the image such that the content and style of that generated one will match the ground truth image. The overall loss function used in this architecture is, weighted sum of adversarial loss, style loss, perceptual loss and reconstruction loss.

The architecture is trained with hyperparameters - resnet9 as both generator modules, learning rate 0.0002, batch size 4, 2300 number of epochs, normal weight initialisation with mean 0 and standard deviation 0.02. During the inference only the resnet9 (generator 1) is used to get the correction results. With this environment, we were able to train the cycleMedGan model to better perform the correction step.

### *5.1.2 Segmentation*

#### *5.1.2.1 Unet*

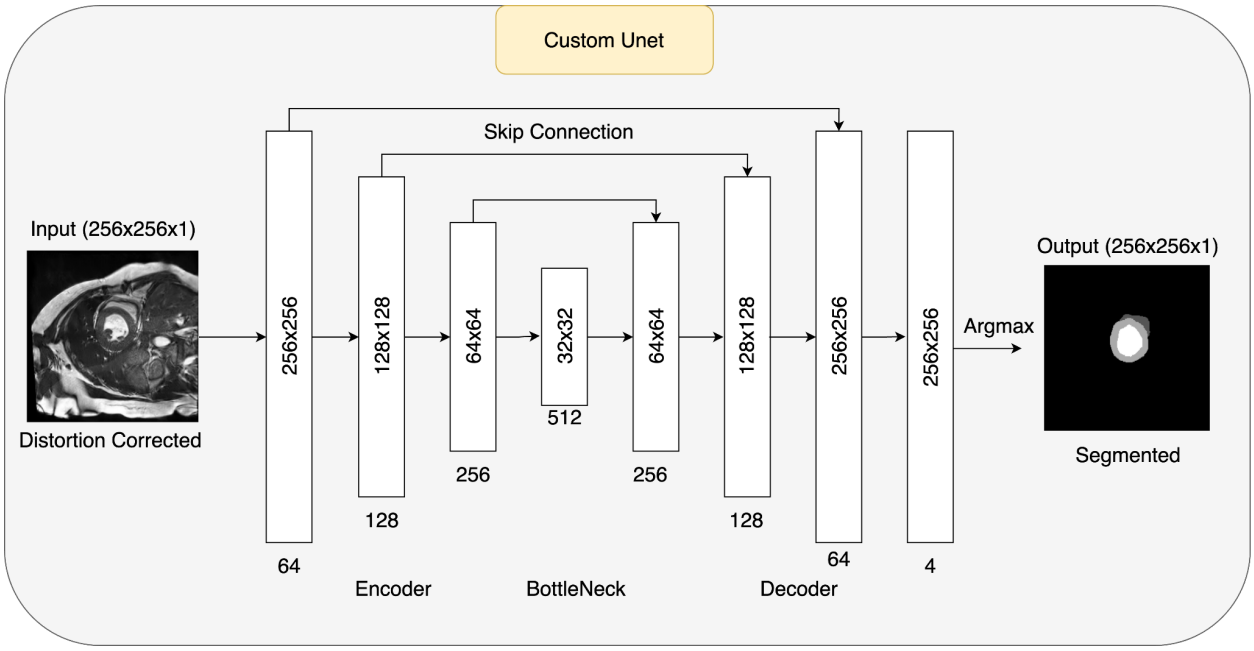


*Figure 13: An overview of cycleMedGAN architecture*

As discussed in the methodology section, U-Net is an architecture for semantic segmentation. It consists of a contracting path and an expansive path. The contracting path follows the typical architecture of a convolutional network whereas an expansive path performs deconvolution operations. In this encoder decoder architecture, skip connection is something which is important which does feature flow from contracting path to expansive path. This helps in sharing the spatial feature information during the upsampling. Apart, most of the working principles of this architecture are the same as we discussed in the methodology section.

In our work, we explored different versions of Unet architecture - custom Unet, Unet-128 and unet-256. Since it's not that complex of an objective at this moment, even custom UNet architecture with just 3 encoder layers, a bottleneck layer and 3 decoder

layers are sufficient to obtain the segmentation results. Here we pass the corrected images as an input to this Unet model and get the segmentation results. Since we are using an ACDC dataset, the output will be the segmentation of 4 classes (data related information will be discussed in the dataset section). The loss function used to train this model is Dice and IoU, since the dice coefficient performs well when there is class imbalance, it is used as the final loss function. Since Unet is the most commonly used network for medical image segmentation, in the traditional approach we stick on to experimenting with different versions of Unet architectures. Other architectures like DeepLabV3 or maskRCNN could also be explored.

The architecture is trained with hyperparameters - learning rate - 0.003, Adam optimiser with regulariser parameter 0.003, batch size 32 and number of epochs 610. With this environment, we were able to train the Unet model to better perform the segmentation step.

## 5.2 Traditional approach comparison and evaluation

In this section we validate and compare the results of different architecture we explored in the traditional approach to solve MRI segmentation under distortion. Here we will understand what type of evaluation matrix is being used, why that is relevant, what is the accuracy we are able to obtain.

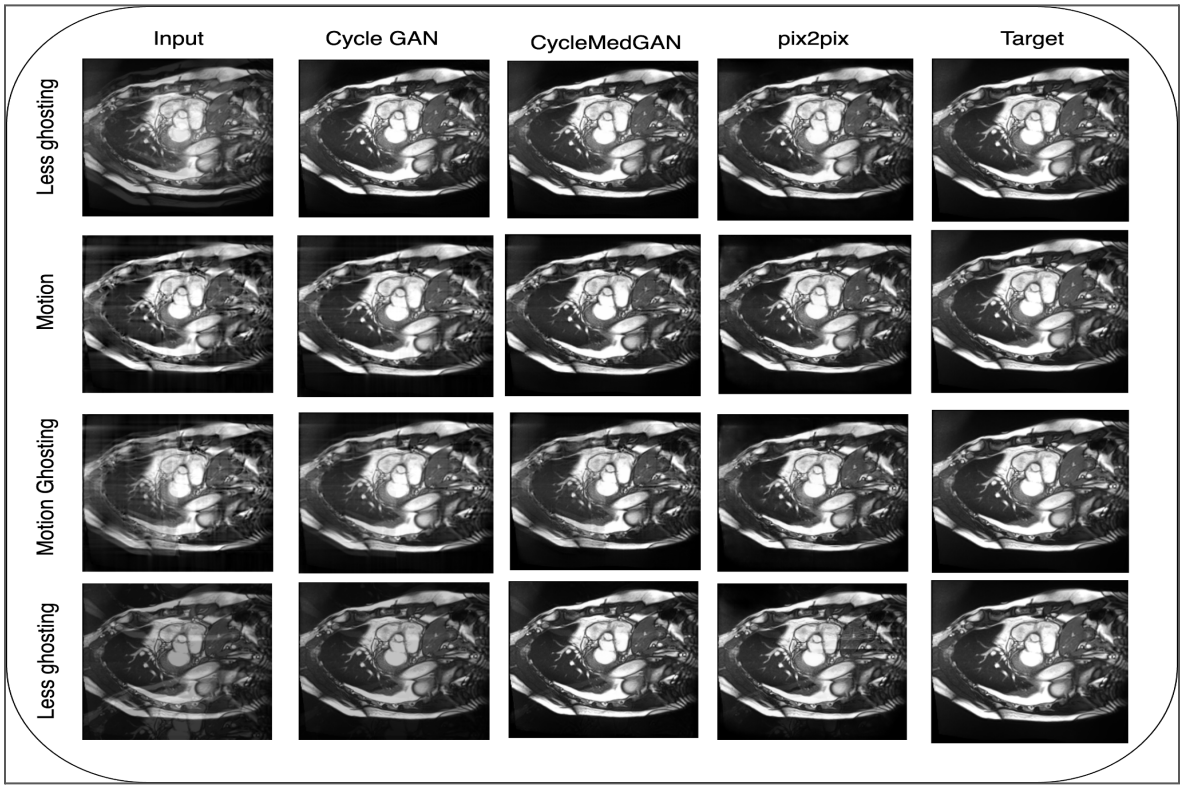


*Figure 14 : Sample outputs of different correction models for various distortion settings*

To evaluate our correction models, we used the SSIM validation metrics. SSIM stands for Structural Similarity Index and is a perceptual metric to measure similarity of two images. This is the most commonly used metric to evaluate image reconstruction algorithms. Given the two images, the structural similarity score is calculated between them - 1 score represents the images are highly identical and 0 represents they are not similar. Between the experimented architectures pix2pix, cyclegan and cycleMedGan the SSIM score obtained for corrected images is presented below.

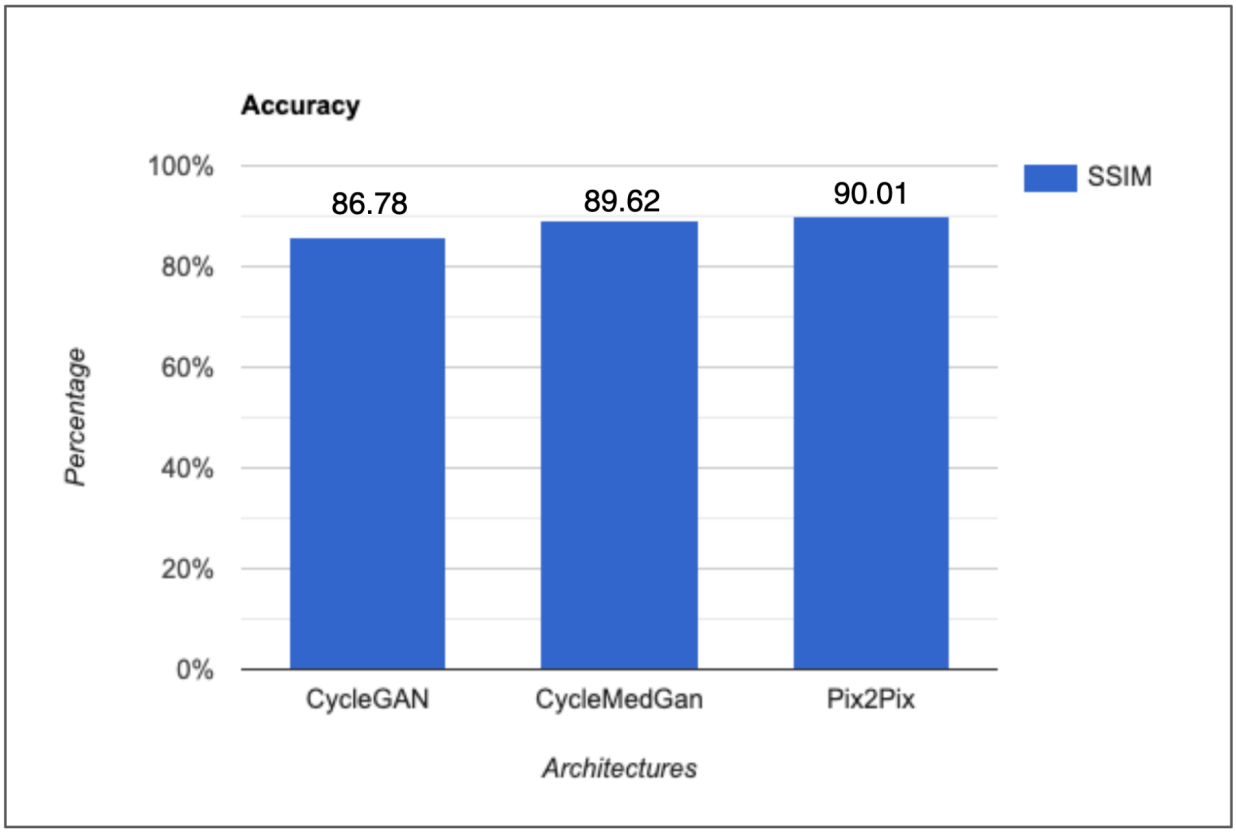


*Figure 15: Quantitative accuracy comparison over different correction models*

The observation is, pix2pix is outperforming all the other correction models by obtaining a 0.90 SSIM score. CycleMedGan is performing almost the same as the pix2pix model by obtaining 0.89 SSIM. Even when we observe visually, the results between pix2pix and cycleMedGan are almost the same. The difference in accuracy is because there is slight change in the pixel intensity and contrast. But cycleGan is not giving us very good results compared to others. The reason for this, cyclegan is an unsupervised learning method and it just has 2 loss function which is making it perform a little lower than the other models. Even though cycleMedGan is unsupervised learning, the pretrained network is helping it to extract deep features and result in better correction results. Anyhow, the final model that we pick from this stage is, pix2pix

because it gives better results and more-over its one to one connection, that is obvious in giving better results.

Once we obtained the corrected images, we did the segmentation using Unet and we used dice loss in our work to evaluate the model accuracy. Dice loss/ F1 score is a harmonic mean of precision and recall. The reason for using the dice loss is, since it makes use of precision and recall, it handles the class imbalance issues. The results that we are obtaining for the segmentation of the corrected image are presented below.

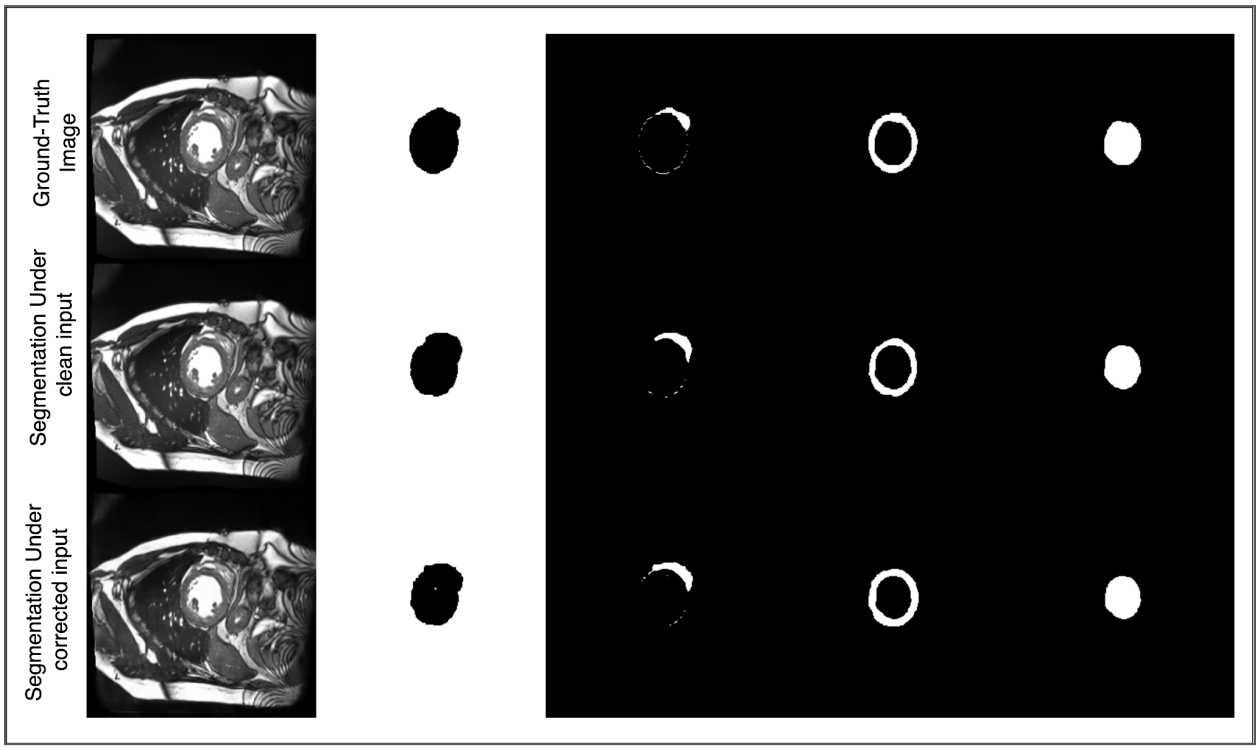


*Figure 16: Segmentation output comparison between corrected and clean input data*

Comparison should be given between the segmentation results of corrected images and clean images. From this experiment we can understand that, we are getting almost the same segmentation results between the clean images and corrected images. This indicates that the correction is happening accurately, the feature level information is constructed as in clean image, so that the segmentation is done efficiently as it does in case of clean images. The dice score of corrected images segmentation is ~0.85 and for clean images it's ~0.86. The segmentation results which we obtained for corrected images i.e ~0.85 will be considered as our final traditional approach result.

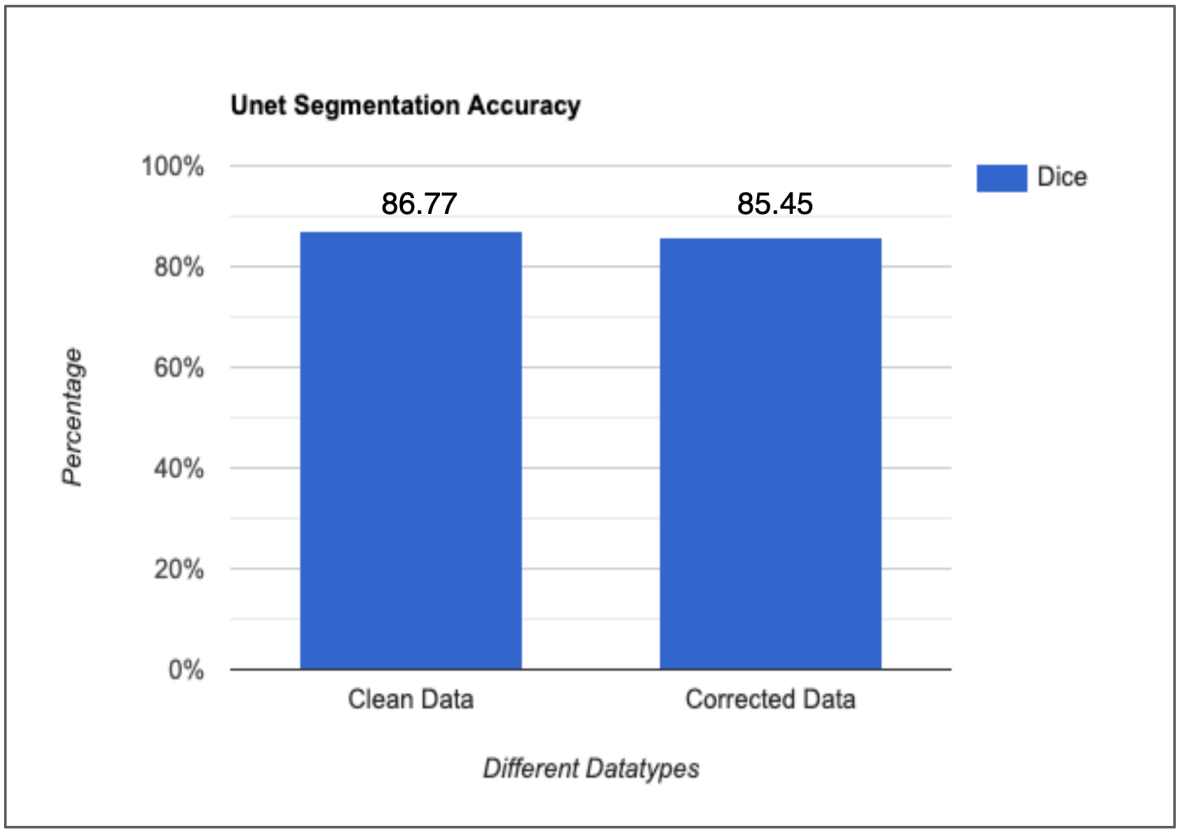


*Figure 17: Quantitative accuracy comparison over different input type for Unet segmentation*

## 5.3 Proposed network experiments & evaluation - Ablation study

In this section, we will discuss the experiments that we performed to construct the proposed network and evaluation strategies used to validate our model performance. We will try to understand this section, by understanding different challenges and solutions that are obtained. Also answering - why are each component required? How is that being affected in overall performance and what is the final accuracy we are obtaining?

As we know our proposed network has 4 major components. Segmenter, discriminator, contrastive and pretrained. Since many components are predefined, in our work all the experiments that are performed are with respect to exploring multiple components and architectures rather than changing hyperparameters and modifying the layers. With respect to segmenter, experimented with different architectures like Unet128, resnet 9 block, Unet256. Both Unet-128 and Resnet 9 blocks didn't give the

expected results. It makes sense though, because our segmenter has to perform both correction and segmentation simultaneously, we need a bit of a complex network.

Lets understand why to use the discriminator ? - From the experimental results, we understood that just segmentation can be done accurately with the dice/IOU/L1 loss function. But when correction comes into picture using a discriminator gives better results. That makes sense because our model has to perform correction as well with segmentation. Using adversarial loss our model was able to do the correction and segmentation simultaneously. We can understand this way - dice loss was helping in the segmentation and adversarial loss was helping in getting better correction/clean segmentation results. Although just using segmenter and discriminator were achieving objectives, It was not satisfactory to see the final results. The problem that we were facing is, due to data imbalance in the ground truth the discriminator was not able to perform upto the mark. To handle this scenario, a pretrained network is used in addition to the discriminator to calculate style, perceptual and content loss. This vgg16 pertained model helped segmenter to generate the segmentation images which matches with the ground truth images in terms of style and content. This method is inspired from the MedGan paper. Changing the vgg16 pretrained to vgg19 or resnet doesn't really matter as it's not a trainable model in our case, it's just a model for feature extraction and loss calculation. In the below result we can see how pretrained networks are helping in improving the accuracy. Without pre-training the model was able to achieve ~0.77 dice score, but having vgg16 it raised to ~0.84 dice score.

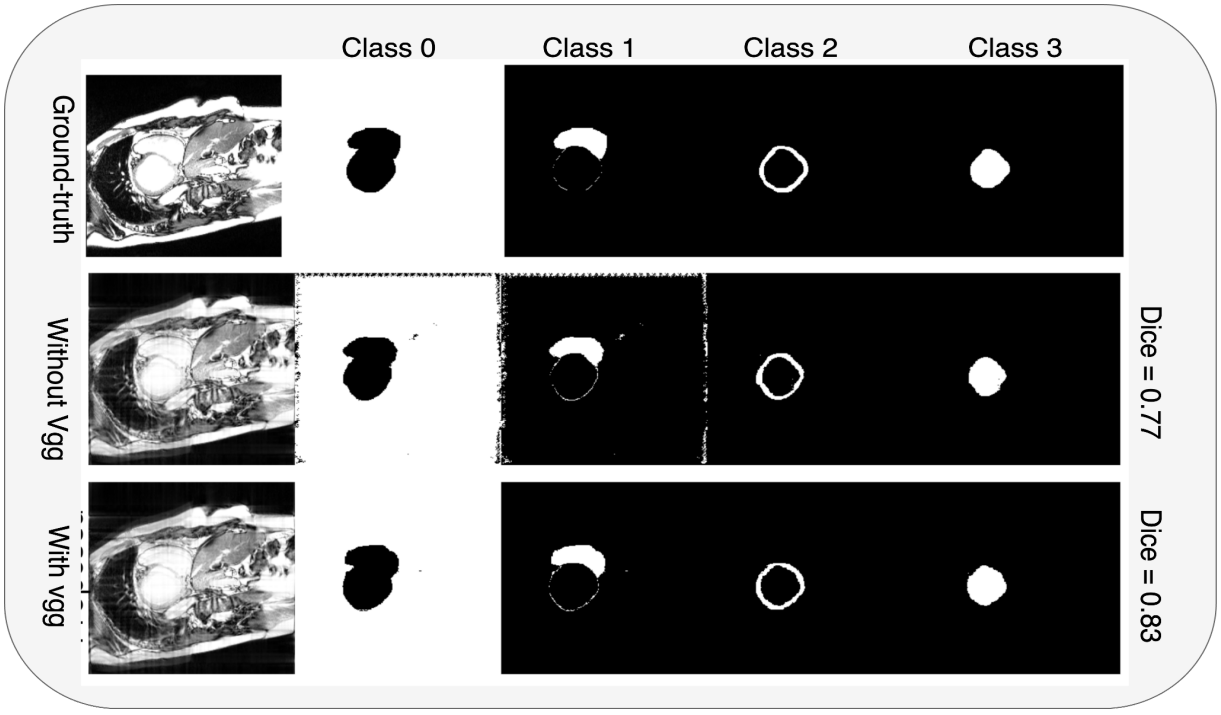


*Figure 18 : Illustrate how the output varies with and without a pre-trained network.*

Also to understand the impact of contrastive learning blocks, the architecture is trained with and without using contrastive blocks. The major objective was that contrastive learning should do better feature extraction i.e intra-class compactness and inter-class dispersion. And with the experiment, we observed that pattern in the feature maps in embedding space, that cluster formation is getting compact while using contrastive learning. Although there is no huge improvement in the accuracy, definitely we can see that the feature extraction is happening efficiently as expected.

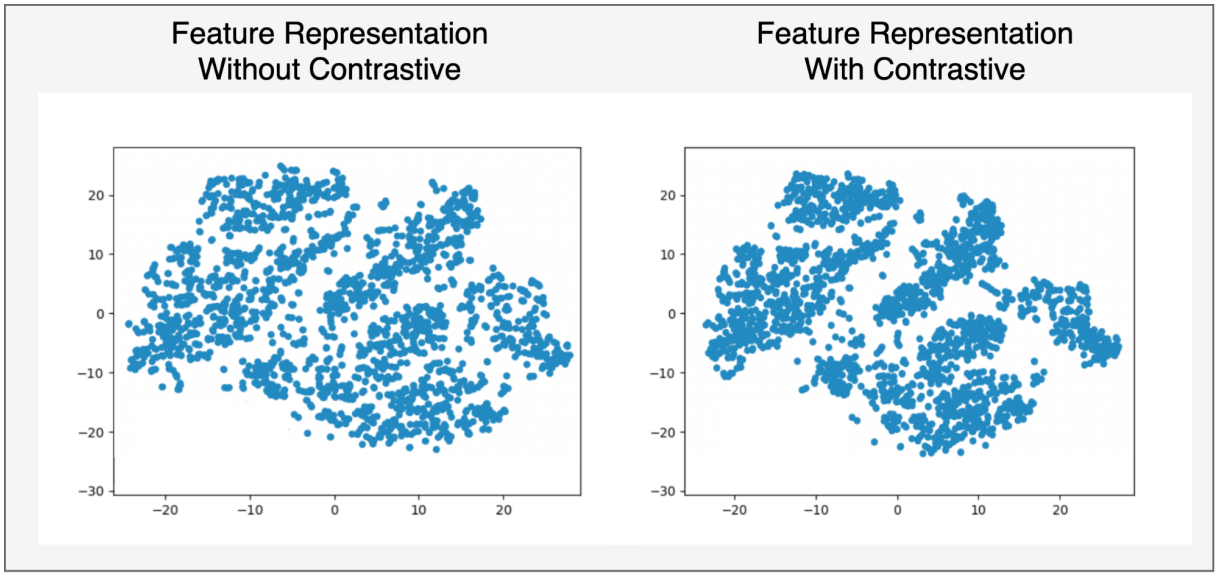


*Figure 19 : The TSNE diagram demonstrates the feature representation while using contrastive learning and without using. Although the cluster class is unknown, we can see that features are getting compact. With this its, evident that compactness is increasing within the class. So we can assume that closer clusters are of the same class.*

Quantitative difference is :

| Mode | Dice |
|---|---|
| Without Contrastive | 83.62 |
| With Contrastive | 84.27 |

With this architectures, the early stopping principle is used to test over different checkpoint and finalize the one (manually) such that the model does not overfit to the network - the final hyperparameters choose are standard values such as learning rate 0.0002, batch size 4, 3100 number of epochs, normal weight initialisation with mean 0 and standard deviation 0.02. And the weight given for each loss is 100 for dice loss and 1 for adversarial loss, 0.1 for contrastive loss, 0.0001 for style and content loss and 20 for perceptual loss. During the inference only the segmenter is used to get the segmentation results. With this environment, we were able to train the proposed model to better perform the correction and segmentation simultaneously.

# CHAPTER 6: Accuracy comparison

In this we compare our proposed network results with the traditional approach results. The architectures that are chosen for the traditional approach are pix2pix as a correction model and a custom UNet network as a segmentation model.

From the below image we can observe that our proposed network is able to perform as good as the traditional approach with the lesser complexity. The traditional model follows the correction step and then the segmentation step to get the accuracy of 0.85 dice score, whereas our model with the single step obtained 0.84 dice score. Also with the execution time, our proposed network outperforms the traditional approach by completing tasks within 4.5ms.

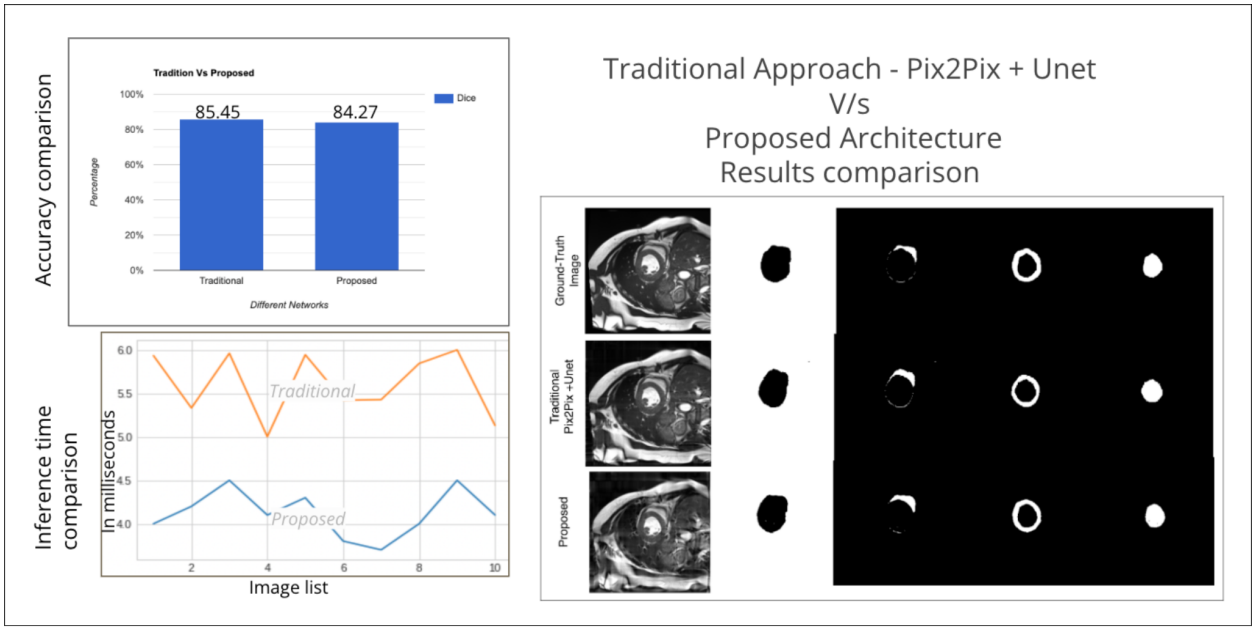


*Figure 20 : Comparison between the traditional approach and proposed network outcome in terms of accuracy and inference.*

# CHAPTER 7: Datasets and assumptions

For our work, since the objective is to optimize the MRI segmentation for distortion settings, the specific dataset would not matter at all. Any MRI images with ground truth annotations could be used. Our work makes use of the ACDC dataset (Automated Cardiac Diagnosis Challenge)[6]. The dataset consists of 150 patient's records, where 100 patient's is for training and 50 patient's is for testing. Each patient's MRI has 10 slices. The objective is to segment 4 classes - 0, 1, 2 and 3 represent voxels located in the background, in the right ventricular cavity, in the myocardium, and in the left ventricular cavity, respectively. The dataset has both MRI and segmentation ground truth. The data preprocessing that we did for our work is, all the image dimensions are resized to 256x256x1. And each image will undergo some distortion settings to output 4 different combinations of distortion. So the total number of final training data is 4000. And the same thing is done for testing images to obtain 1600 images. No augmentation strategy is used because of the memory limits.

The assumption in our work is, we believe that, the distortion which has been introduced is simulating the real world distortions. We could say that the assumption is strong and valid, because the distortion has been introduced in a MR data acquisition way. That means, we had access to K-space through the fourier transform and distortion is introduced in k-space which simulates the real world scenario. Even in the MR data acquisition the distortion happens only when K-space is distorted. The same procedure is replicated here to introduce the motion distortion. So it's theoretically solid enough to say the assumption is strong enough. Furthermore to validate our assumption we have conducted an experiment with respect to statements made in MedGAN. In MedGan on the real time distortion they have observed the accuracy decrease as the degree of distortion increases. So in our work as well, we generated 3 bins of distorted images for the correction step. When we train the pix2pix model with these 3 distortion parameters, the observation is the same, as we can see in the below image. As the degree of distortion increases, the accuracy decreases. This gives the pattern which we observe for the real time. These theoretical understanding and the experimental results gives us confidence that our assumption is valid and strong enough.

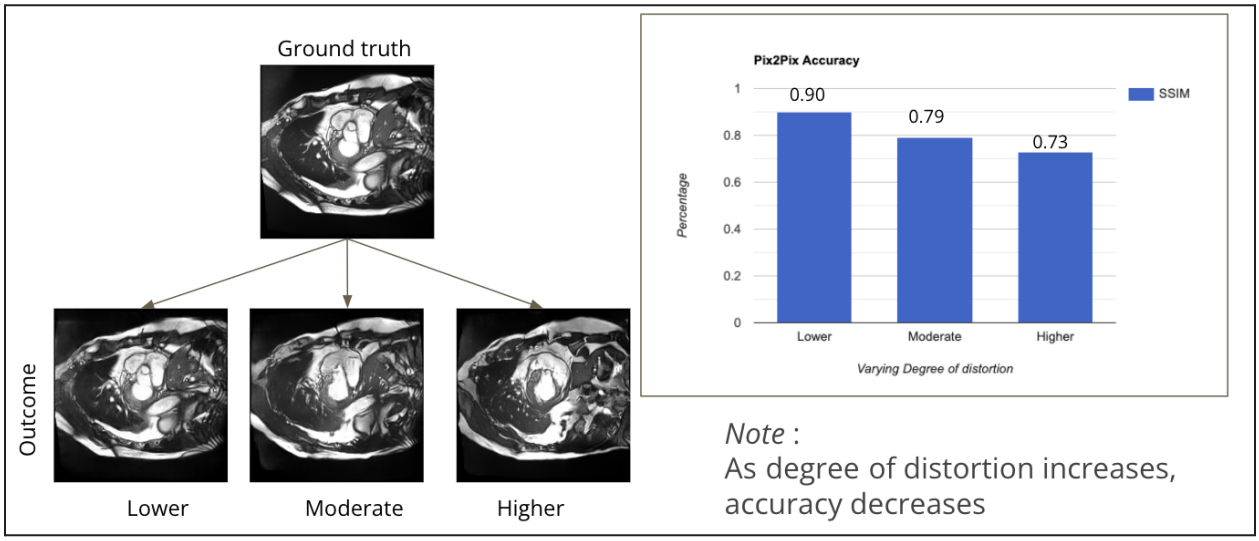


*Figure 21: Represents how the correction model accuracy decreases while increasing the degree of distortion*

# CHAPTER 8: Limitations and Explainability issue discussions

The current limitations are, in both the traditional approach and the proposed network, the model accuracy is not so good at missing patch and viewing distortion. So we had to discard it for further consideration. This is because, for other distortion settings like ghosting or motion, the information just has to be corrected. But for the missing patch or the viewing distortion, the information has to be filled. This distortion settings in something like inpainting rather than correction. Currently both the traditional approaches and the proposed model are not capable of handling this arbitrary distortion. Also, our work doesn't suit all MRI segmentation tasks. That means our work is more relevant to the MRI segmentation for volumetric calculation of organ rather than diagnosis problems.

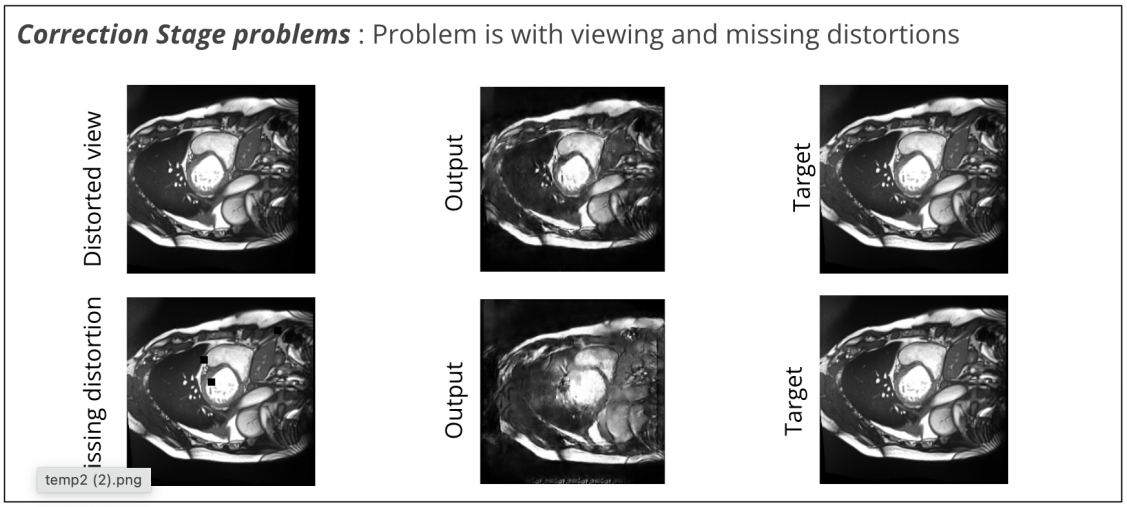


*Figure 22: Illustrate the discrepancies in the correction model output for missing patch and viewing distortion.*

To find the culprit in our work, we need to debug on the basis of distortion. If any problem happens in MRI segmentation under arbitrary distortion, in the traditional approach we can check whether it's a problem in the correction step or its in a segmentation step. But in our case, since the network parameters are getting optimized for both correction and segmentation simultaneously, it's not possible to identify between the stages. So the possible way to find the culprit is, we have to divide the data on the basis of distortion settings and cross check. This way we will get to know the culprit. One thing to understand here is, in most of the cases, the model won't be a problem. So rather than finding the issue on the basis of models or stage, the first step is to debug on the basis of data.

# CHAPTER 9: Conclusion

Finally in this research work, we have addressed a problem of MRI segmentation under arbitrary distortion in a novel way where no one has attempted. That is bringing down the two stage solution to the single stage. This work addressed the correction of several distortions while the traditional approaches were aiming for a particular distortion setting. The proposed network is able to perform the segmentation task under arbitrary distortion with a dice score of ***0.84*** and has achieved comparable results with traditional network of dice scores ***0.85***. While the traditional approach takes 2 networks to achieve the task, our network takes just 1 network and out performs in terms of inference time ***(~4.2ms)***. Also, our work involves and succeeded in employing the pixel-wise contrative block(novel try) for efficient feature extraction which helped in slight improvement in the accurray. Hence, we can conclude that the task can be achieved with a single network with improved inference time and obtaining the par results as traditional approaches.

# CHAPTER 10: Future work

Our work takes two points from the current research work for the future activity. One is definitely to try and solve the issues related to the missing and viewing distortion. As we discussed the current proposed network is not that great in solving these two distortions, we would like to continue this work as a feature enhancement. Along with this, another point is to utilize information between the images in pixel-wise contrastive learning. Our work would like to explore memory bank management in contrastive blocks for furthermore efficient feature extraction. Currently, the pixel wise contrastive loss is calculated from the different classes of one image, but through the memory bank concept, it's possible to collect information from many images to calculate the loss. Coloring the TSNE plot with respect to class is taken to the future work. Also, testing on the real time distortion settings could be taken as a future work.

# References

[1] Bracewell, R. N., & Bracewell, R. N. (1986). The Fourier transform and its applications (Vol. 31999). McGraw-Hill New York.

[2] Respiratory Motion Correction in Abdominal MRI using a Densely Connected U-Net with GAN-guided Training. https://arxiv.org/pdf/1906.09745.pdf.

[3] Karim Armanious, Chenming Jiang, Marc Fischer, Thomas Küstner, Tobias Hepp, Konstantin Nikolaou, Sergios Gatidis, Bin Yang, MedGAN: Medical image translation using GANs, Computerized Medical Imaging and Graphics, Volume 79, 2020, 101684, ISSN 0895-6111,https://doi.org/10.1016/j.compmedimag.2019.101684.

[4] ipA-MedGAN: Inpainting of Arbitrary Regions in Medical Imaging. https://arxiv.org/abs/1910.09230. Submitted to IEEE ICIP 2020.

[5] Ronneberger, O., Fischer, P., Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. In: Navab, N., Hornegger, J., Wells, W., Frangi, A. (eds) Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015. MICCAI 2015. Lecture Notes in Computer Science(), vol 9351. Springer, Cham.

[6] Automated Cardiac Diagnosis Challenge Dataset, MICCAI Challenge 2017 - https://www.creatis.insa-lyon.fr/Challenge/acdc/databases.html

[7] Wang, Zhou; Bovik, A.C.; Sheikh, H.R.; Simoncelli, E.P. (2004-04-01). "Image quality assessment: from error visibility to structural similarity". IEEE Transactions on Image Processing. 13 (4): 600–612. Bibcode:2004ITIP...13..600W. CiteSeerX 10.1.1.2.5689. doi:10.1109/TIP.2003.819861. ISSN 1057-7149. PMID 15376593.

[8] Usman, M., Latif, S., Asim, M. et al. Retrospective Motion Correction in Multishot MRI using Generative Adversarial Network. Sci Rep 10, 4786 (2020). https://doi.org/10.1038/s41598-020-61705-9 -Published in nature.

[9]Exploring Cross-Image Pixel Contrast for Semantic Segmentation. https://arxiv.org/pdf/2101.11939.pdf - submitted to CVPR.

[10]P. Isola, J. Zhu, T. Zhou and A. A. Efros, "Image-to-Image Translation with Conditional Adversarial Networks," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 5967-5976, doi: 10.1109/CVPR.2017.632.

[11]J. Zhu, T. Park, P. Isola and A. A. Efros, "Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks," 2017 IEEE International Conference on Computer Vision (ICCV), 2017, pp. 2242-2251, doi: 10.1109/ICCV.2017.244.

[12] Armanious, Karim & Jiang, Chenming & Abdulatif, Sherif & Küstner, Thomas & Gatidis, Sergios & Yang, Bin. (2019). Unsupervised Medical Image Translation Using Cycle-MedGAN. 1-5. 10.23919/EUSIPCO.2019.8902799.

[13] Chen, Liang-Chieh and Papandreou, George and Schroff, Florian and Adam, Hartwig. (2017). Rethinking atrous convolution for semantic image segmentation.

[14] Kaiming He, Georgia Gkioxari, Piotr Dollár, Ross Girshick: “Mask R-CNN”, 2017; [http://arxiv.org/abs/1703.06870 arXiv:1703.06870].

[15] Moratal D, Vallés-Luch A, Martí-Bonmatí L, Brummer M. k-Space tutorial: an MRI educational tool for a better understanding of k-space. Biomed Imaging Interv J. 2008 Jan;4(1):e15. doi: 10.2349/biij.4.1.e15. Epub 2008 Jan 1. PMID: 21614308; PMCID: PMC3097694.

[16]Keiron O'Shea, Ryan Nash: “An Introduction to Convolutional Neural Networks”, 2015; [http://arxiv.org/abs/1511.08458 arXiv:1511.08458].

[17]Ting Chen, Simon Kornblith, Mohammad Norouzi, Geoffrey Hinton: “A Simple Framework for Contrastive Learning of Visual Representations”, 2020; [http://arxiv.org/abs/2002.05709 arXiv:2002.05709].

[18]Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, Ross Girshick: “Momentum Contrast for Unsupervised Visual Representation Learning”, 2019; [http://arxiv.org/abs/1911.05722 arXiv:1911.05722].

# Appendix A: GitLab Repository

To run the experiments which are done in this research work, please access the git repository using the link.

- https://git-teaching.cs.bham.ac.uk/mod-msc-proj-2021/sxh1373.git
- Or https://github.com/santhoshhollap/MSC_Project.git (back up, - trained model not present)

***Information about the repository:***

The training and the inference code in the repository is configured for executing in google colab and data has to be uploaded to gdrive. This repository has code for pix2pix, cyclegan, cycleMedgan, Custom-Unet and proposed network. Also, this repository has code for a data generation module and validation which has to be executed in the personal systems - python environment. The dependencies and libraries are specified in requirement.txt and environment.yaml.

***Steps to setup the pipeline:***

- Download the git repo into the personal system.
- To generate the distorted images, scripts and the sample images are present in the folder “Data_gen”. please open the terminal under this folder and run the following commands -
    - For view distortion → ***python3 view.py <input_path> <output_path>***
    - For ghosting + motion distortion → ***python3 ghosting_motion.py <input_path> <output_path>***
    - For more ghosting distortion → ***python3 ghosting_more.py <input_path> <output_more_ghosting_path>***
    - For motion distortion → ***python3 motion.py <input_path> <output_path>***
    - *For less ghosting distortion* → ***python3 ghosting_less.py <input_path> <output_less_ghosting_path>***
    - For missing patch distortion → ***python3 missing.py <input_path> <output_path>***
- To train and infer the model, upload the folders called “ACDC_Sample_Data”, “Unet_traditional”, “trained_model” and “Msc.ipynb” file to the gdrive. ACDC_Sample_Data folder has sample of ground truth and distorted images, Unet_traditional has traditional approach unet model and data, trained_model has final trained model (pix2pix, cycleGan, cycleMedGan and proposed) which can be used for inference and Msc.ipynb is a tool from which training and inference can be done.

- Open the “Msc.ipynb” file in the google colab to train and infer the model
- To train the model
    - Install dependencies as the cell indicates
    - Connect to gdrive
    - Specify the path to training source and target. Source will be ACDC_Sample_Data/Distorted and target will be ACDC_Sample_Data/Clean if training is correction model else ACDC_Sample_Data/Segmented_GT for proposed. The folder will be in gdrive.
    - Specify the training mode as per which model needs to be trained :
        - Train_mode → choose from [pix2pix,cyclegan,cyclemedgan,proposed]
        - Model_mode → choose from [pix2pix,cycle_gan,cycle_mgan,proposed]
    - Model name and path is the user's choice. Specify the hyper-parameters
        - Batch size = 2 or 4
        - Learning rate = 0.0002
        - Number of epochs 1500+ for better results
        - Patch size 256
    - If continuing the training then specify the pretrained network path else ignore.
    - Just run the other cells to start the training process.
- To infer the model
    - Install dependencies as the cell indicates
    - Connect to gdrive
    - Directly jump to the “Generate prediction(s) from unseen dataset” cell.
    - Specify the source path “ACDC_Sample_Data/Distorted” which will be in gdrive. The result folder is the user's choice.
    - Specify the training mode as per which model needs to be trained :
        - Train_mode → choose from [pix2pix,cyclegan,cyclemedgan,proposed]
        - Model_mode → choose from [pix2pix,cycle_gan,cycle_mgan,proposed]
    - Specify the prediction folder path as per which model has to be inferred. For example to infer proposed → trained_model/pix2pix, which is in gdrive.
    - Specify the patch as 256, and checkpoint as latest for running inference.
    - To see the output image just execute the “Inspect the predicted output”.
- To train and infer the traditional Unet model : open Unet.ipynb file in google colab. Mount to gdrive and change the directory to the Unet_traditional. Run the

cells to start training. To infer the model, few modification needs to be done in main.py
    - In line 391 - change the saved_params_path to “models/unettrainval.pt”
    - Change load_model_params to true, save_model_params = false, train = false and test = true.
    - Change the output saving directory to user’s choice gdrive path in line 287 under model_test function.
- To validate the model output, some of the output images and ground truth are present in the ACDC_Sample_Data folder. The following commands can be executed in a personal system.
    - To validate correction model → ***python3 SSIM.py <input_path_GT> <Input_Output>***
    - To validate traditional model Unet output → ***python3 DiceScore.py <input_segmentedout> <input_maskGT>***
    - To validate the proposed network → ***python3 proposedvalidation.py <segmentedoutput> <Segmented_GT>***
- To check the compactness of feature map through contrastive, please run the python code which is in ContrastiveVisual folder → ***python3 TSNE.py***

Note: Please follow the steps to get smooth execution. If you need to reproduce the accuracy and results the model must be trained on a full ACDC dataset with specified parameters[important]. Although the detailed steps are given above, if any issues occur, please refresh and reconnect the colab to re-execution and try. Also in repo, there might be few unwanted files and folders. Please ignore.

***Contribution of this work:***

In this work the major contribution is the great project idea. This work has attempted and succeeded in solving the MRI segmentation under arbitrary distortion in a single step. Also this is the first work for using the combination of 4 components - segmenter, pretrained, discriminator and contrastive. No one has tried to solve this before in a single step, for arbitrary distortion and used this combination to solve the problem. The self contribution around the coding part is →

- Implementing the data generation module for introducing various distortions. All files in the data_gen folder are my own work.
- Implementing the validation logics → All files in the validation folder is my own work.
- Modified the pix2pix repository to run cylegan, cycleMedGan and proposed work → integration of several repositories to establish the final pipeline is my work.

- Modified the notebook of pix2pix for running all pix2pix, cyclegan, cyclemedgan and proposed models. → Modification done in MSc.ipynb
- Implementing the TSNE module for feature visualization → Contrastive visual is my own work.
- Modified the folder “models” with integrating and modifying few files in it →
    - Modified the cycleGan to get CyleMedGan architecture by adding vgg network file and adding the extra-lines of code with respect to style, content and perceptual loss. → Modification of cycleMedGan file
    - Modified the pix2pix to get the proposed network by integrating the vgg16 network and adding lines with respect to style, content and perceptual loss. Also implemented the code for interpolation of feature maps for contrastive learning → Modification of proposed file.
    - Modified the network.py files with respect to extracting the last 4 layers of encoders feature map for contrastive loss function. → modified network file
    - The input to the contrastive block is modified from the conventional way. Instead of taking the last layer, our work takes the last 4 → new ideas are my own work.
- The traditional approach to Unet architecture has been used from previous work of my own. So this is considered to be my work to implement and unet architecture.
- All the experiments, evaluations and reports are my own work.

***References for coding:***

Our work majorly concentrated on the research idea, experimentation and solving logic. So code has been referred to some of the open source and modified as per our applications and needs. Few of the syntax and codes are referred from the function’s documentation pages and few from open sources repositories.

- As a baseline repository, our work utilized the repository of https://github.com/junyanz/pytorch-CycleGAN-and-pix2pix.git. And made all the necessary changes which are required for our work.
- To make the training pipeline user friendly, our MSc.ipynb has been modified upon the notebook originally made in work specified in this link.
- Vgg baseline architecture has been referred to by open source blogs.
- For projection head and contrastive loss calculation our work utilized the repository → https://github.com/tfzhou/ContrastiveSeg.git.

Note : Any confusions or concerns in understanding/executing the repository or code or the report, please feel free to contact me on email - santoshhollap@gmail.com