Synthetic Data Generation for Medical Images Using Generative Adversarial Networks (GANs)

OLLSCOIL NA GAILLIMHE
UNIVERSITY OF GALWAY

Naveen James
School of Computer Science
University of Galway

Supervisor(s)
Dr. Frank Glavin

In partial fulfillment of the requirements for the degree of
MSc in Computer Science (Artificial Intelligence)
August 21st 2024

DECLARATION I, Naveen James, hereby declare that this thesis, titled "Synthetic Data Generation for Medical Images Using Generative Adversarial Networks (GANs)", and the work presented in it are entirely my own except where explicitly stated otherwise in the text, and that this work has not been previously submitted, in part or whole, to any university or institution for any degree, diploma, or other qualification.

Signature: Naveen James

Abstract

This thesis investigates the use of Generative Adversarial Networks (GANs) to generate synthetic medical images for cancer diagnosis, with a focus on addressing the challenges of class imbalance and the complexity of generating high-quality, diverse images. By exploring various architectures of GANs such as Pix2Pix, DCGANs, CatGAN, StyleGAN, this work demonstrates the potential of GANs to significantly enhance diagnostic accuracy. However, the challenges such as maintaining the realism of synthetic images, preventing mode collapse and ensuring training stability remain critical issues that this research aims to address. Through systematic experimentation and analysis, this thesis seeks to highlight the effectiveness of GANs in addressing these challenges.

Keywords: Generative Adversarial Network, CycleGANs, Unsupervised Domain Adaptation, Data Augmentation

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The field of medical imaging has seen remarkable advancements in the recent years, fueled by the emergence of advanced imaging methodologies. However, these advancements are frequently hampered by the scarcity of extensive, diverse and annotated datasets [4]. This challenge is particularly prominent in the field of cancer imaging, where the obtainment of high volume of data isn't always possible. Synthetic Data Generation has become a feasible solution to overcome these limitations, providing a means to enhance existing datasets and boost the performance of the machine learning algorithms effectively. This thesis centers on the creation of Synthetic Data Generation using Generative Adversarial Networks (GANs) with a specific focus on cancer imaging. A significant hurdle in medical imaging is the bias in the datasets, which is often called as the "Majority Class Dominance" [4]. This occurs when the class disproportionately is represented in the dataset, resulting in a biased model performance [4]. This is a crucial problem to tackle especially in the field of medical imaging. Furthermore, issues related to data privacy and regulations, such as the General Data Protection Regulation (GDPR), further complicate the process of securing data . Synthetic Data Generation presents a solution to these challenges by producing diverse and

anonymised datasets that can be used for research applications.

1.0.1 Motivation

The motivation for this research roots from the need to enhance the quality and diversity of the medical imaging datasets, with particular emphasis on the cancer domain. Cancer continues to be one of the leading causes of global mortality, making early detection predominant for improving patient prognoses. Despite advances in technology, the scarcity of labeled medical images availability for training robust machine learning models remains a significant obstacle. This shortage not only hinders the development of effective diagnostic tools but also aggravate issues of data bias, especially the majority class representation problem. By generating high-quality synthetic images, this research aims to augment existing datasets, thereby addressing these critical gaps. Synthetic data generation can help mitigate the biases ensuring a more balanced and comprehensive dataset. Moreover, synthetic images can help in the training of machine learning models in situations where obtaining real images is impractical due to privacy and ethical concerns. The potential benefits of this approach is multifaceted because high quality synthetic data can accelerate the pace of research and innovation of medical imaging by providing sufficient amount of data for model training and validation. Ultimately, this research thrives to leverage the capabilities of Generative Adversarial Networks (GANs) to create synthetic data that enhances the real world datasets, helping in making better cancer diagnosis.

1.0.2 Research Questions

The following questions will be addressed in this research:

1. Can Synthetic Images generated by GANs improve the performance of the

various Machine Learning and Deep Learning algorithm's classification of cancer images?

2. To what extent can we generate images that maintain a high level of realism and diversity?

3. What are the challenges associated with training GANs for medical image synthesis and how can it be prevented?

Chapter 2

Background

This chapter provides the foundational knowledge about the Machine Learning / Deep Learning algorithms and Generative Adversarial Networks (GANs) architectures used in synthetic data generation and the metrics used to evaluate their effectiveness. The aim of this chapter is to provide a comprehensive understanding of the terminologies and principles that underpin the current research, setting the stage for a detailed investigation in the subsequent section.

2.1 Machine Learning Models

2.1.1 Logistic Regression

Logistic Regression is a simple yet powerful algorithm that has been performing really well in medical domain. It is used for binary classification tasks. Logistic Regression works by applying sigmoid function which produces a probability value between 0 and 1 [5]. The model is trained by adjusting its weights to minimize the difference between the predicted probabilities and the actual outcomes, which typically is done by using cross entropy loss function [5]. Logistic Regression has shown its significance in the field of medical domain especially in cancer imaging

for its potential to be used for early detections through biomarkers etc [5].

2.1.2 Support Vector Machine

Support Vector Machine (SVMs) works by non-linearly transforming input vectors into a high dimensional feature space, where a linear decision surface would be constructed [6]. This decision surface is designed to maximise the margin between different classes, focusing on the support vector, which are critical data points nearest to the boundary. The key strength of SVMs rely on its ability to create a clear separation between the classes, even in complex, high dimensional spaces. They are more robust and versatile because of their ability to adapt to non-linear tasks by utilising kernel functions.

2.1.3 Random Forest

Random Forest is an ensemble method that is used for both classification and regression tasks [7]. It works by constructing multiple decision trees during the training and outputs the class thats either the mode of the class (for classification tasks) or the mean prediction (for regression tasks) of the individual trees. The advantage of Random Forest is their ability to handle overfitting, endurance to irrelevant features, and also their capability to provide insights into feature importance during predictions. Random Forest is a powerful method as they aggregate the results from several decision trees and improve their predictive accuracy.

2.2 Neural Networks

2.2.1 Convolutional Neural Networks

Convolutional Neural Networks [8] are a type of deep learning model which is built to utilise the spatial and structural information innate in 2D or 3D images. They maintain spatial relationships between pixels by using local receptive fields [1], weight sampling [2] and sub-sampling [3]. CNNs are highly effective in the fields of image classification [9] and segmentation [10] because of these aspects. The overall architecture of Convolutional Neural Networks is as follows:

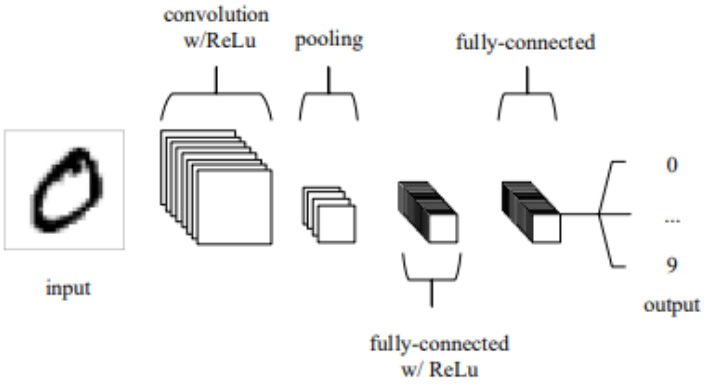

Figure 2.1: Convolutional Neural Network Architecture. Image from [1]

CNN [1] consists of three main parts: the convolution layers, the pooling layers and the fully connected layers. The convolutional layers apply filters to local region of the images to produce feature maps. Then the pooling layers are

[1]Allows each neuron in the convolutional layer to connect to a local region of the input image, which aids in detecting edges.

[2]This applies the same set of filters across different regions of the input image which helps the network to identify the same feature in different parts of the image.

[3]Otherwise called pooling, helps in reducing the dimensionality of the feature maps by preserving the crucial spatial information.

employed to reduce the spatial dimensions of the feature maps and the highest pixel of the image gets highlighted. Finally, the fully connected layers would take these extracted features from the convolutional and pooling layers and map them to their respective outputs. Their architecture is enhanced by the additional introduction of non-linearity such as Rectified Linear Unit (ReLU) [1] activation and various regularization methods such as dropouts which would help in preventing overfitting.

2.2.2 U-Net

U-Net [12] is designed for image segmentation tasks, particularly in the field of medical imaging. U-Net consists of two parts which are the contracting path (encoder) and the expansive path (decoder). The encoder resembles a convolutional neural network [8] and includes several blocks of operations. Each block features two consecutive 3x3 convolution layers which is followed by ReLU activation and 2x2 max pooling layer for downsampling. The decoder, on the other hand, handles the upsampling of feature maps. Each and every upsampling step would involve a 2x2 up-convolution, which would halve the feature channels, followed by concatenation with their corresponding feature maps from the encoder. This allows the model for precise localisation.

2.2.3 Inception V3

The Inception V3 architecture is a Deep Convolutional Neural Network that is designed to achieve high performance with efficient computation [13]. This

[1] ReLU [11] is mathematically expressed as

$$f(z) = \max(0, z)$$

(Where"z" is the input), is a non-linear function that outputs 0 for input lesser than 0 and for anything greater than 0 it outputs the input itself.

model builds upon the Inception architecture by using techniques like factorised convolutions, aggressive regularization and dimension reduction to increase the network's depth and width without excessively increasing the computation cost. It also utilises parallel structures within the Inception modules, employs batch normalisation and introduces the notion of label smoothing to improve generalisation and stability during training. Finally, the model is renowned for its ability to achieve state-of-the-art performance in large scale image classification tasks such as the "ImageNet Dataset" while still maintaining computational efficiency [13].

2.2.4 EfficientNetB5

EfficientNetB5 is a variant within the EfficentNet variations of Convolutional Neural Networks. It stands out unique because of its approach towards scaling, where it scales the network's depth (number of layers), width (number of channels) and resolution in a balanced manner [14]. Unlike the other variants, this variant uses 456x456 resolution and incorporates more layers making it able to capture complex patterns and features in the data. This is essential in the fields such as medical imaging where capturing fine details is very crucial.

2.3 Generative Adversarial Networks

2.3.1 Standard Generative Adversarial Networks

Generative Adversarial Networks have shown their prominence in the field of imaging as they are able to mimic real data to a great extent [15]. GANs have two networks: *Generator* and *Discriminator*. The Generator tries to deceive the Discriminator and the Discriminator tries to evaluate the generator's output by

comparing it with the real data to validate its authenticity. GANs are highly renowned because of their ability to render their network architecture depending upon the specific use case [4].

2.3.2 Conditional Generative Adversarial Networks

Conditional Generative Adversarial Networks [16] are an extension to the traditional GAN framework by including additional information in both the generator and discriminator. This additional information is referred to 'y' and it can be various forms of data such as class labels or other modality-specific data. By this incorporation of additional information Condition GANs are able to produce data based on specific conditions. This is achieved by feeding the additional information into the generator and discriminator as input layers which allows the model to generate more target specific data.

2.3.3 Categorical Generative Adversarial Network (CatGAN)

CatGAN improves upon the traditional GAN model by modifying the discriminator to provide probability distributions over multiple classes for each input, rather than just differentiating "real" or"fake" samples. CatGAN focuses on maximising the mutual information between the observed samples and their predicted class distributions to ensure the model learns to classify the data into distinct categories. The objective of CatGAN can be seen as a natural generalisation of the GAN framework or an extension to the regularised information maximisation framework [1] that aims to improve classification performance with limited labelled data.

[1] Aims to group unlabeled data into well separated categories [17]

2.3.4 Deep Convolutional Generative Adversarial Network (DCGAN)

DCGANs [18] are a class of Convolutional Neural Networks (CNNs) which are specifically designed to enhance the stability and performance of GANs. DCGANs consist of a series of architectural enhancements, that include substituting pooling layers with strided convolutions, incorporating batch normalization in both the generator and discriminator and further eliminating fully connected layers to create deeper architectures. The generator consists of ReLU in every layer except the output layer, which has TanH activation. The discriminator utilises LeakyReLU in all the layers. These adaptations in the architecture contribute in stabilising the training process and alleviating issues like mode collapse [1], improving the effectiveness of both the generator and discriminator.

2.3.5 StyleGAN2

StyleGAN2 [19] is an enhanced version of the original StyleGAN [20] architecture, which was introduced to address several limitations and improve the quality of the generated image. Vital improvements include the redesign of generator normalisation by replacing adaptive instance normalisation [2] with weight demodulation, which normalises convolutional layer weights for more stable training. Unlike the original StyleGAN, StyleGAN2 maintains a consistent resolution throughout training, avoiding artifacts and providing stable training. The model also supports larger capacities, allowing for higher-resolution outputs with finer details and realism.

[1] Happens when the generator doesn't produce a diverse set of outputs, instead, it generates a limited set of outputs that are similar to each other and collapse to a few specific patterns of the data distribution

[2] A method that modifies the content features to match the mean and variance of the style features, this allows the content of one image to blend with the style of the other, enabling style transfer [21]

2.4 Metrics

2.4.1 Wasserstein Distance

Wasserstein Distance [22] also known as Earth Mover's Distance (EMD) measures the distance between two probability distributions by calculating the minimal cost required to transform one distribution into the other. This is done by considering all possible ways of transporting mass (probability) from the points of one distribution to the points of the other and further find the transport plan that minimises the overall cost.

$$W(Pr, Pg) = \inf_{\gamma \in \Pi(Pr,Pg)} \mathbb{E}_{(x,y)\sim\gamma}[\|x - y\|] \quad [23]$$

The mathematical formulation is as follows: $\Pi(Pr, Pg)$ is the set of all joint distributions $\gamma(x, y)$ whose marginals are P_r and P_g, and this term, $\gamma(x, y)$ indicates the amount of mass that needs to be transported from 'x' to 'y' to transform the distribution of P_r into P_g. Wasserstein Distance would then be the cost of this optimal transport plan. In relevance to synthetic data generation, this can be used to measure how closely the generated data distribution matches with the real data distribution. Using Wassertein Distance as a loss function, the GAN can more effectively minimise the difference between the real and generated data distributions. This helps in addressing issues such as mode collapse, where the generator would produce repetitive outputs.

2.4.2 Inception Score

Inception Score [24] works by applying the Inception model [1] to each generated image to derive conditional label distribution. The score measures the image quality by evaluating how confidently the Inception model can classify each generated image into a specific category. Furthermore Inception Score evaluates the diversity of the generated images by ensuring that the set of images covers a wide range of categories. It is particularly relevant for synthetic data generation as it provides a measure of both the quality and diversity of the generated images, which are crucial for the effectiveness of the synthetic data which is generated.

2.4.3 Frechet Inception Distance (FID)

Unlike the Inception score, which only considers the output label distribution, FID [26] measures the distance between two multivariate Gaussian distributions fitted to the feature representations of real and generated images extracted from Intermediate layers of the Inception model. The overall process involves passing in the real and generated images into the Inception V3 model and extracting the activations from the intermediate layers and then these activations are used to estimate two gaussian distributions, one for the real images and another for the generated images.

[1] Its a Convolutional Neural Network built to utilise the factorised convolutions and parallel structures within the network, which enables it to perform well under constraints of memory and computational budgets [25]

Chapter 3

Related Work

This chapter reviews the existing works and research specifically focusing on the use of Generative Adversarial Networks (GANs) and their variants for synthetic data generation in the field of cancer imaging. This chapter aims to provide a detailed examination of the advancements and applications of GANs methodologies in this domain. Through thorough investigation, this chapter seeks to identify the advancements that has been made and the knowledge gaps that remains.

3.1 Generative Adversarial Networks

3.1.1 Skin Lesion Image Generation

Rashid et al. [27] implemented a GAN-based approach to generate synthetic dermoscopic images for skin lesion classification. The method consisted of a GAN architecture utilizing a generator and a discriminator. The generator, a deconvolutional neural network, learns to produce realistic images from the noise , while the discriminator, a convolutional neural network tries to classify the images as real or fake. This study used the "ISIC 2018 Challenge Dataset" [28]. The dataset was augmented by conventional techniques such as gaussian blurring, noise ad-

dition, cropping along with synthetic data generated by GAN. The generated images from GAN was passed into a CNN structure and the performance of the model was significantly improved but it fell into the limitation of instability in training and sometimes presence of artifacts were found.

3.2 Pix2Pix

3.2.1 Skin Lesion Image Generation

As discussed earlier, one of the difficulties in the field of medical domain is getting hands on large and diverse datasets for training models. Ghorbani et al. [29] acknowledges this challenge in their research particularly focusing in dermatology. They propose a DermGAN which is modified Pix2Pix architecture designed to synthesise images of skin conditions. Pix2Pix [30] is a model designed to answer image-to-image translation problems using a Conditional Generative Adversarial Network. The model aims to learn the mapping from an input image to an output image and also simultaneously learn a loss function to train this mapping. This approach allows for the application of a consistent technique to several image translation tasks which would need different loss function. DermGAN works by translating the semantic maps which encode details of the skin condition, including size, location and underlying skin colour. Major improvement to the original pix2pix model is that this includes replacing transposed convolution layers which might lead to "Checkerboard Artifacts" [1]. To get rid of "CheckerBoard Artifacts" the deconvolution layer was replayed by nearest-neighbour resizing layer [31]. The DermGAN architecture consists of a modified U-Net generator that incorporates the nearest-neighbour resizing layers and fully-convolutional discriminator. The dataset used in this study was collected from teledermatology service consist-

[1] grid-like patterns that distort the smoothness of the generated images

ing of 49,920 images from 9897 cases across 17 clinical sites [32]. By employing DermGAN the model translates the semantic maps into realistic images of the skin conditions. The semantic maps used in this study are RGB images, where 'R' channel encodes the skin color, while 'G' and 'B' channel would encode the presence of the skin condition under the Region of Interest (ROI). The training process entail minimising four loss components which are l_1 reconstruction loss for the entire image, l_1 reconstruction loss for pathological region, feature matching loss for the second-to-last activation layer of the discriminator and finally the mini-max GAN loss. DermGAN was evaluated using Inception Score, Frechet Inception Distance and Human Evaluation. The authors also mentioned that Inception Score isn't an ideal metric for this case because it presumes that all the classes given it are a part of predefined classes available in the Inception model [33]. The results achieved showcased that the synthetic images were able to contain the characteristics of the respective skin conditions and could efficiently represent diverse skin colours and variance in skin and location of the conditions. The authors also pointed a drawback that the generated images sometimes lacked fine details and high resolution necessary for certain diagnostic applications. This could affect the effectiveness of the generated images in accurately representing the subtle features which are crucial in diagnosis perspective.

3.2.2 Gastrointestinal Images Generation

Adjei et al. [34] addressed the issue of scarce medical imaging data in gastrointestinal images and employed Pix2Pix framework to generate synthetic esophagael cancer images from segmentation maps. Their pix2pix architecture comprised of a U-Net generator with skip connections and a PatchGAN discriminator [30].

The model was evaluated using Multiscale Structural Similarity Index [1] and perceptual image path similarity score [2], with U-Net architecture and cross-entropy loss producing more realistic images. One of the limitation of the model was the generator was producing limited and repetitive outputs.

3.3 Categorical Generative Adversarial Networks (CatGAN) and Wasserstein Distance

3.3.1 Dermoscopy Image Generation

The CatGAN framework extends the traditional GAN model by modifying the discriminator's objective. Yi et al. [37] employed a CatGAN with wasserstein distance to generate lesion images that are specific for dermatology applications. Instead of simply distinguishing between real and fake samples, the discriminator outputs confidence values for the input belonging to each of the underlying classes. To stabilize the training, CatGAN utilised Wasserstein distance to measure the distance between distribution of the real and generated samples. The performance of CatGAN was evaluated on the "ISIC 2016 Skin Lesion Challenge Dataset" [38] and the "PH2 Dataset" [39]. The "ISIC dataset" consisted of 900 training images (173 melanomas [3] 727 benign cases [4]) and a test set with 75 melanomas and 304 benign cases. The initial step was pre-processing the training images to standard input size and augmentation techniques (rotation, flipping, etc) were implemented to bring diversity and increase the size of the dataset, thus improving the robustness of the model. The second dataset "PH2" consisted of

[1] Assesses image quality across different resolutions and combines these measurements to offer a comprehensive evaluation [35]

[2] employs deep features extracted from pre-trained neural networks to assess the perceptual similarity reflecting human visual perception more accurately than traditional metrics [36]

[3] Type of Skin Cancer which is considered to be the most cancerous

[4] Non-cancerous that do not spread to different parts of the body

200 dermoscopy images of melanocytic lesions which includes 40 melanomas, and was used as a validation dataset. CatGAN showed significant improvement with proper fine-tuning, achieving a precision of 0.424 in just 140 labelled images. The model was also able to generate realistic dermoscopy images that could be used for data augmentation, enhancing the training dataset and improving the classifier performance. The authors pointed out a limitation related to instability during training and difficulties in generating high-resolution images. Overall, this research emphasises the potential of CatGAN to improve classification tasks in dermatology in scenarios with limited data.

3.4 Deep Convolutional Generative Adversarial Network (DCGAN)

3.4.1 Cervical Cancer Image Generation

The study [40] employs DCGANs to generate synthetic microscopic images of cervical cancer cells, aiming to augment training datasets for classification models. The architecture consists of two components: the generator and the discriminator. The generator uses fractional-stride convolutions in place of pooling layers to learn from random noise vectors and generate images through spatial up-sampling. Contrarily, the discriminator evaluates the authenticity of the images, ensuring they resemble the real data. The model was evaluated with Mean Squared Error (MSE) [1], Structural Similarity Index (SSIM) [2], and Peak Signal To Noise Ratio (PSNR) [3]. Additionally, visual inspection by domain experts was held to validate the realism. The study used a dataset consisting of cervical

[1]Computes the pixel and colour intensity
[2]Computes the structural similarity between the distributions of real and generated images
[3]Calculates the noise between real and generated images

cancer images categorised as normal and abnormal. Each class contained 500 images, respectively. The dataset was augmented using OpenCV and additional 100 images from each class was reserved for validating a CNN classifier. This augmented dataset was then used for training the DCGAN model and it proved to be effective. The generated data was assessed by CNN on various ratios of real to synthetic data. With various settings, the ratio of 80:20 gave the best results. DCGAN showed promise in generating realistic synthetic data and improving classification performance, it had its own limitations, such as training instability, mode collapse, which are common issues in GANs. The generated images were realistic but occasionally failed to accurately differentiate finer details.

3.4.2 Liver Lesion Image Generation

Continuing on the use of DCGANs, Frid-Adar et al. [41] applied a similar approach to generate synthetic data for CT images of liver lesions to improve classification performance in medical imaging. The architecture utilised a generator that transforms random noise to realistic images through fractionally-strided convolutions and a discriminator that evaluates these images against the real samples. The synthetic data, alongside the real ones was passed in CNN for classification and the results showed that there is a significant increase in the performance. However, this approach also faced the same issue of mode collapse and instability during training.

3.5 Adaptive StyleGANS2

3.5.1 Brain Tumour Image Generation

Tariq et al. [42] utilised the StyleGAN2 architecture to generate synthetic brain MRI images. The architecture consisted of a generator and a discriminator with an extra addition of ADA [1]. The authors have used a Kaggle dataset comprising of 154 brain MRI samples and a total of 3064 T1-weighted images categorised into Glioma, Meningioma and Pituitary. They leveraged pre-trained models from the FFHQ [44], BreCaHad [45] and AFHQ [46] to enhance the training process through transfer learning [2]. StyleGAN2 employs path length regularisation [3] and weight demodulation [4] techniques to handle droplet artifacts and stabilise training process. The results showed that using FFHQ model for transfer learning produced the highest quality synthetic images with variability. Furthermore, through visual inspection and evaluation, it was evident that the synthetic images closely matched with the real MRI brain images. Despite these advancements, the research faced challenges related to training instability, with occasional artifacts occurring in the generated images.

3.6 Summary

In this chapter, various methodologies for synthetic data generation in cancer imaging was explored. GAN-based approaches such as Standard GAN, Pix2Pix,

[1] employs a variety of augmentations to the discriminator's input images, preventing overfitting without allowing these augmentations to leak into the generated images [43]

[2] A technique utilising knowledge acquired from one or more tasks to enhance the learning performance in a different but related task [47]

[3] Ensure that a consistant change in the latent space leads to a predictable change in the output image, promoting a smooth and stable mapping from latent vectors to the generated images [48]

[4] Adjusts the weights of the convolutional layers to maintain consistent signal strength, improving overall image stability.

CatGAN, DCGAN and StyleGAN2 were extensively explored. Pix2Pix leverages U-Net architecture was effective in producing realistic images but faced limitations in fine detail accuracy and repetitive outputs. CatGAN, utilised Wasserstein distance, showed high precision with minimal labelled data, but encountered training instability difficult in generating high resolution images. DCGANs was employed to enhance classification performance in microscopic and CT images, yet common issues like mode collapse, training instability persevered. StyleGAN2 along with ADA and transfer learning produced high quality synthetic brain MRI images highlighting improvements over previous models but still struggled with stability and occasional artifacts. These methodologies showed the significant advancements in synthetic data generation while also revealing persistent challenges such as instability in training, mode collapse and issues in generating high resolution images. Below is a table that summarises the main work that we explored.

Author Name	Description	Algorithm Used	Metrics Used	Results Achieved	Limitations
Ghorbani et al.	Generated synthetic skin condition images using modified Pix2Pix	Pix2Pix, U-Net	MS-SSIM, LPIPS	Improved image realism and variability	Difficulty capturing Fine detail accuracy and generating high-resolution images required for diagnostic
Yi et al.	Utilised CatGAN with Wasserstein distance for dermatology (Skin) lesions	CatGAN, Wasserstein Distance	Precision, Inception Score	High precision score with minimal labeled data	Instability during training and difficulties in generating high-resolution images
Ali et al.	Used DCGANs to generate synthetic microscopic cervical cancer images	DCGAN	MSE, SSIM, PSNR	Enhanced classification performance	Training instability and mode collapse
Frid-Adar et al.	Applied DCGANs to generate synthetic liver lesion CT images	DCGAN	Accuracy, AUC	Significant performance improvement in classification, high quality generated images	Mode collapse and instability during training
Tariq et al.	Utilized StyleGAN2 with ADA to generate synthetic brain MRI images	StyleGAN2, ADA	Inception Score, FID	Highest quality synthetic images with diversity	Training stability issues and occasional artifacts noticed in the generated images
Adjei et al.	Used Pix2Pix to generate synthetic esophageal cancer images	Pix2Pix, U-Net	MS-SSIM, LPIPS	More realistic images with cross-entropy loss	Limited and repetitive outputs was generated from the generator
Rashid et al.	GAN-based approach to generate synthetic dermoscopic images	GAN	Precision, Recall, F1-score	Improved classification performance	Instability in training and noticed presence of artifacts in generated images
Tariq et al.	StyleGAN2 with transfer learning for synthetic brain MRI images	StyleGAN2, Transfer Learning	Inception Score, Human Evaluation	High-quality synthetic images, visual inspection	Training stability issues and occasional artifacts in generated images

Table 3.1: Summarisation of the related works chapter

Chapter 4

Data

The dataset which is used in this study is taken from Kaggle [2], which is curated for a challenge that aims at developing image-based algorithms to identify histologically confirmed skin cancer cases. The dataset features single-lesion crops from 3D Total Body Photos (3D-TBP), resembling close up smartphone images. One of the motives for choosing this dataset is its potential to address the significant challenges of early skin cancer detection, especially in underserved populations. Many populations around the world lack access to specialised dermatologic care, which can delay diagnosis and treatment. Moreover, the dataset consists of over 400k images out of which only 393 are malignant. This stark imbalance highlights another critical reason for selecting this dataset as it presents the real world challenge of class imbalance which is very common in medical domain. The dataset also provides a lot of diversity and quality of images in its collection. The dataset consists of wide range of skin lesions captured from thousands of patients across three continents. The dataset consists of weak-labelled tiles, which were considered benign based on doctor's visual inspection. This distinction allows for the development of models that can differentiate between malignant and benign lesions, even when the labelling confidence differs.

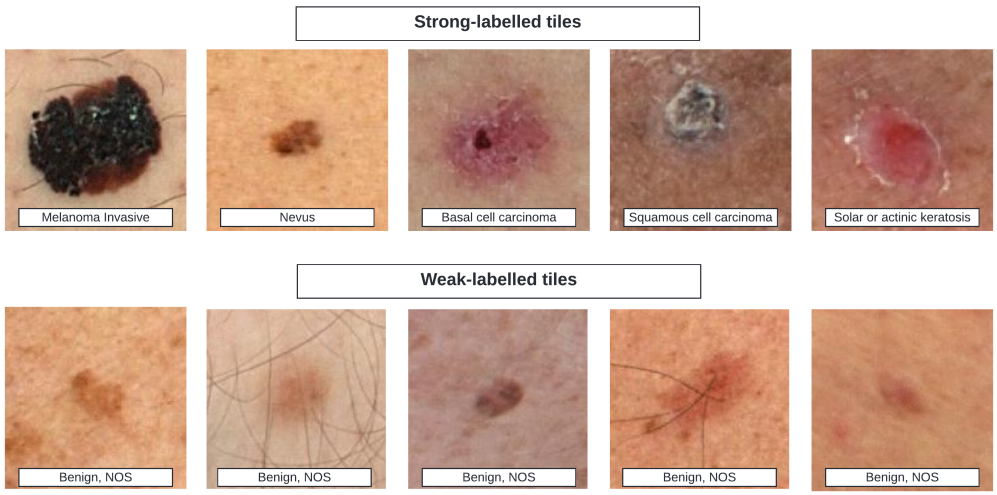

Figure 4.1: Example of Strong and Weak Labelled tiles [2]

More importantly, while this dataset includes metadata, the focus of this research is on the image data itself. The images are provided in high resolution JPEG format, and each image is associated with a binary label indicating whether an image is benign or malignant. This structure is particularly suitable for Generative Adversarial Networks (GANs), leading to creating more diverse dataset suitable for making Machine Learning and Deep Learning models generalise more to the dataset and tackle the problem of class imbalance. This way, the dataset serves as a great utilisation to equip GAN architecture which is aimed at improving the image quality and diversity leading to better generalisability for the models.

Chapter 5

Methodology

This chapter outlines the methodologies architectures of GANs explored to investigate the generation of synthetic images to check whether they would help algorithms to tackle class imbalance and improve classification performance. The goal is to explore the potential of GANs in addressing class imbalance. 300 images were used for this generation purpose. To achieve this, the following GANs architectures were explored:

1. CycleGAN with Unsupervised Domain Adaptation.

2. Wasserstein GAN with Gradient Penalty (WGAN-GP).

Among these two architectures, CycleGAN with Domain Adaptation emerged successful by providing images with realism and diversity that significantly helped in the improvement of classification performance of various algorithms. Consequently this chapter will provide an in-depth exploration of CycleGAN architecture, discussing its implementation, and the factors which contributed to its success. While in contrast the WGAN-GP didn't achieve the expected outcomes in this specific application. Though this method is theoretically sound, it faced a lot of issues in generating synthetic images. Despite its limitation, its worthwhile

to explore WGAN-GP and reasons for its shortcomings such as training instability and mode collapse. The reason to choose these architectures was not just because of their ability to generate realistic images but also with diversity.

5.1 CycleGAN

CycleGAN is one of the many powerful architectures of the GANs variants and was developed for image-to-image translation tasks [49]. Moreover, CycleGAN is specifically built for unpaired datasets [3]. Unpaired datasets are the ones where there is no correspondence between the input images in one domain and the output images in the other domain. For example, if you building a model to translate horses to zebras, an unpaired dataset would consist of a collection of horse images and a separate collection of zebra images without them having any explicit pairing between horse images and zebra images [3]. Although the model is a powerful architecture, it needs to be fine tuned properly else it will suffer from training instability. Therefore, it is recommended to fine tune the model to specific tasks. CycleGANs consist of two generators (G_A, G_B) and two discriminators (D_A, D_B). If we are given a set of images from one domain X and a different set of images from domain Y, we can then train a model which can learn to map $G : X \rightarrow Y$, in a way that the output is $\hat{y} = G(x)$, $x \in X$ and it is identical to images $y \in Y$ which is achieved by an adversary trained to classify $\hat{y}$ apart from y [3]. This just means that the model can learn to convert an image from one group into a realistic image that could belong to the second group. The adversary, also known as the discriminator, will help to ensure that the transformed images are indistinguishable from the actual images, leading to realistic conversions. However, the challenge arises when we don't have paired examples that can guide in the translation task. Rather, we assume there is

an underlying relationship between both the domains in a way that two different representations of the same scene and we aim to learn that relationship. The goal is to train a mapping $G : X \rightarrow Y$ so that the output $\hat{y} = G(x)$ is indistinguishable from images in domain Y. In theory, this approach would create a distribution over $\hat{y}$ that would match the actual distribution Y, it doesn't guarantee that each individual image in domain X is paired meaningfully with an image from Y. Moreover, training only with the adversary objective can be difficult, often would lead to mode collapse where the model generates the same output to all the input images without any diversity [3].

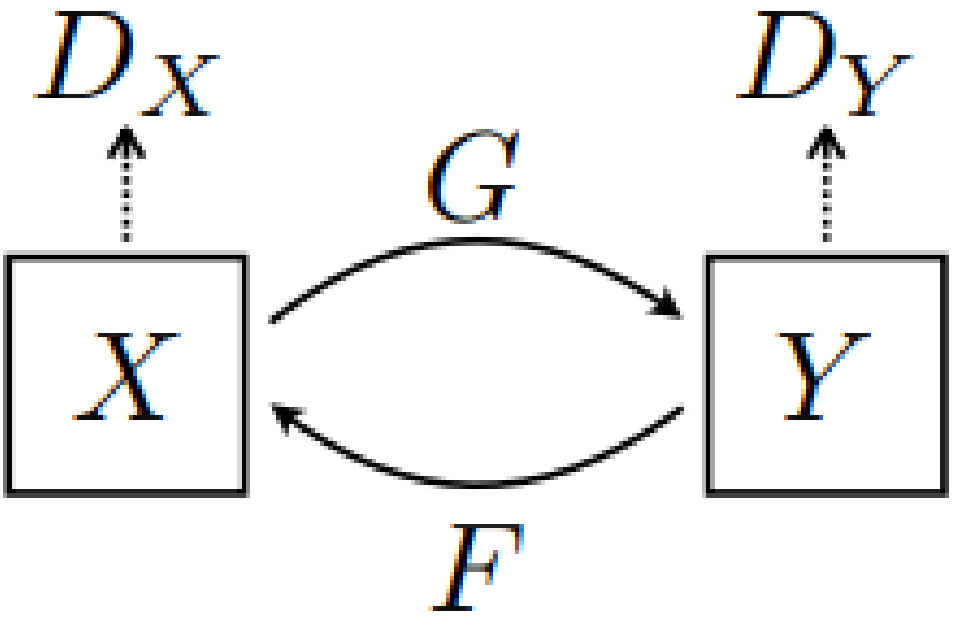

Figure 5.1: The goal pf CycleGAN [3]

For a meaningful translation, the model needs to be "cycle consistent", meaning if we have two translators $G : X \rightarrow Y$ and $F : Y \rightarrow X$, then this should

mean that G and F are inverses of each other [3] as shown in figure 5.1. We train both the mappings G and F at the same time and include what is called a "cycle consistency loss [1]" which ensures that $F(G(x)) \approx x$ and $G(F(y)) \approx y$. This, when combined with the adversarial losses for both domains X and Y will get the complete objective for translating images between unpaired domains. The goal of CycleGAN is to learn optimal mappings for the generators G_A, G_B and two discriminators D_A, D_B. The discriminators are designed to distinguish between the real images and the generated images, outputting a value between 0 and 1 where Ω represents the set of all possible image samples [49]. Finally, the overall objective of CycleGAN is composed of three loss functions are they are:

1. Adversarial Loss
2. Cycle Consistency Loss
3. Identity Loss

5.1.1 Adversarial Loss

This puts both the generator and the discriminator against each other in a two player game. The adversarial loss $L_{\text{adv}}(G_A, D_B)$ measures how well generator G_A is doing in fooling the discriminator D_B [3]. The adversarial loss can be defined as:

$$L_{\text{adv}}(G_A, D_B) = \mathbb{E}_{b\sim B}[\log D_B(b)] + \mathbb{E}_{a\sim A}[\log(1 - D_B(G_A(a)))]$$

The first term $\mathbb{E}_{b\sim B}[\log D_B(b)]$ represents the average of the log of $D_B(b)$, where b is sampled from the real images of the domain B. In this case, the discriminator D_B tries to output a value close to 1 for real images b. The second term

[1] Ensures that when an image is translated from one domain to another and translated back to original domain, the resulting image will highly resemble the original input image, thus maintaining the integrity of the transformation [50].

$\mathbb{E}_{a\sim A}[\log(1 - D_B(G_A(a)))]$ represents the average of the log of $1 - D_B(G_A(a))$, where a is sampled from domain A and $G_A(a)$ is the generated image in domain B. Finally, D_B will output a value close to 0 for the fake images that are generated by G_A from domain A. In summary, these two terms will define a loss function that G_A tries to minimise and D_B tries to maximise. The generator G_A improves by trying to produce images $G_A(a)$ that will make the discriminator D_B output a value close to 1, i.e, making the discriminator think that the generated images are real. For the pair, G_B and D_A the adversarial loss is defined similarly except the roles are reversed.

5.1.2 Cycle Consistency Loss

Cycle consistency loss is used to ensure that the conversion from one domain to another and reverting back to its original domain results in a final image that closely resembles the original image [3]. That is, if you start with an image a from domain A, convert it to an image in domain B by using a generator G_A and revert back to its original domain A using another generator G_B, the final image should be similar to the original image a. Cycle consistency loss $L_{\text{cyc}}(G_A, G_B)$ can be written as:

$$L_{\text{cyc}}(G_A, G_B) = \mathbb{E}_{a\sim A}\left[\|G_B(G_A(a)) - a\|\right] + \mathbb{E}_{b\sim B}\left[\|G_A(G_B(b)) - b\|\right].$$

It is the sum of the differences between the original images and the reconstructed images after applying both the generators sequentially. The function motivates the generators to cooperate in a way that the output of one generator becomes the input to the other. The transformation happens in a way that it preserves the content of the original image.

5.1.3 Identity Loss

Identity Loss [3] is a technique used to ensure that when the input image already belongs to the target domain, the generator does not change its key characteristics. This helps to retain the original characteristics of the image. Identity loss $L_{\text{id}}(G_A)$ for G_A can be defined as:

$$L_{\text{id}}(G_A) = \mathbb{E}_{b \sim B} \left[\| G_A(b) - b \| \right].$$

Similarly, the identity loss for G_B can also be written.

5.1.4 Full Objective

The full objective is the linear combination of the adversarial loss, cycle consistency loss and identity loss. Each of these losses contribute certain aspects of the overall transformation process. Like the adversarial loss makes encourages the generators to produce images that are not differentiable from the real images, the cycle consistency loss makes sure the original characteristics are retained when the image is transformed from one domain to another and back to original domain. Finally, the identity loss preserves the characteristics of the images that are already in the target domain. Furthermore, these loss functions are weighted by factors such as $\lambda_1, \lambda_2, \lambda_3, \lambda_4$, and λ_5 which determine the relative importance of each loss in the overall objective. $\lambda_1, \ldots, \lambda_5$ would be the weights assigned to each loss functions and their values typically range between $\lambda = (1, 1, 10, 5, 5)$ [49]. The full objective function can be defined as:

$$\begin{aligned} L(G_A, G_B, D_A, D_B) = \lambda_1 L_{\text{adv}}(G_A, D_B) + \lambda_2 L_{\text{adv}}(G_B, D_A) \\ + \lambda_3 L_{\text{cyc}}(G_A, G_B) + \lambda_4 L_{\text{id}}(G_A) + \lambda_5 L_{\text{id}}(G_B) \end{aligned}$$

This configuration makes sure that the discriminator are involved in the adversarial losses while the generator tries to maintain the cycle consistency and identity properties to produce realistic images and transformation across all domains.

5.2 Domain Adaption

Domain Adaptation is a technique within the field of transfer learning that focuses on improving the performance of a model trained on domain to adapt and generalise to a different but related domain [51]. The main challenge in domain adaptation is the shift that happens when the distribution of data from the target domain is different from the source domain. Traditional machine learning model's might fail to capture these intricacies and there would be a drop in their performance. Domain Adaptation tries to mitigate this issue by aligning the data distribution from the target and the source domains. There are several types of domain adaptations techniques such as:

1. Semi-supervised Domain Adaptation [52]
2. Unsupervised Domain Adaptation [53]
3. Adversarial Domain Adaptation Neural Network (ADAN) [54]

and many more. We will focus on unsupervised domain adaptation as it was used in the proposed architecture.

5.2.1 Unsupervised Domain Adaptation

Unsupervised Domain Adaptation (UDA)[53] is a technique used when we have more of source domain but less of target domain. This approach is particularly

useful for scenarios where collecting more data is infeasible. In context to my study, where there is abundance of benign images (source domain) and less of malignant images (target domain). By applying UDA, the model can be trained on the benign images and could be made to adapt to generate malignant images without the need of additional malignant images.

5.3 Architecture - 1 CycleGAN with Unsupervised Domain Adaptation

In this section, I present the implementation of the CycleGAN architecture enhanced with Unsupervised Domain Adaptation (UDA) to generate malignant images from the benign images. This architecture mitigates the issue of class imbalance and facilitates the generation high quality and diverse malignant images. The model is trained on the ample benign images available (3000 images were taken for this study) and malignant images (300 images were used). The adversarial loss plays a crucial role in training the generators, as it encourages the generators to produce images that are not differentiable from the real images in the target images. Then the cycle consistency loss was implemented to help in ensuring that the "cycle consistency" is intact. That is, the transformation from one domain to another and back to original is stable. The intricate details of the image such as the color, the structure of the images etc are taken care by the implementation of the identity loss. This loss helps in establishing the generated images aren't too far off from the characteristics of the original image. The structural setup of the architecture would be visited in the next chapter.

5.4 Wasserstein GAN with Gradient Penalty

Wasserstein GAN(WGAN) addresses the critical issues of traditional GANs such as training instability and mode collapse. Wasserstein GAN minimises the wasserstein distance as it is a more smoother and meaningful loss function [55] providing better gradients for training the generator. To make sure, WGAN model satisfies the mathematical conditions required to compute the Wasserstein distance the discriminator (known as critic in WGAN) must be constrained to 1-Lipschitz function. Originally it was proposed to use clipping of weights to make sure the critic is constrained to 1-Lipschitz function but that led to various problems and one of them is vanishing / exploding gradients [55]. To overcome this, WGAN-GP introduces gradient penalty which ensures that the inputs have norm 1 and ensures 1-Lipschitz constraint. Gradient penalty is added as a regularisation term to the WGAN objective, which would penalise the critic if it deviates from the desired norm.

5.5 Architecture - 2 Wasserstein GAN with Gradient Penalty

In this section, the implementation of WGAN-GP is explained. The purpose of choosing this architecture is as stated before, because of its ability to generate high quality and diverse synthetic malignant images. The architecture of the generator consists of down-sampling and up-sampling layers, combined with residual blocks to capture complex features and generate high resolution output images. The residual blocks [1] helps in preserving the high quality features during the

[1] takes an input tensor 'x' passes through convolutional layer followed by batch normalisation and ReLU activation and the result is added back to the original input tensor 'x', this helps in implementing shortcut connections and mitigating vanishing gradient problem

transformation process. The discriminator is a Convolutional Neural Networks designed to classify real / fake images. Spectral normalisation [1] is employed to each convolutional layer to stabilise the training by controlling the Lipschitz constant of the network. The discriminator will output a single value stating indicating the authenticity of the image where higher values corresponds to real images. The training of the model involved alternating between updating the discriminator and the generator. The discriminator is updated multiple times per generator step to ensure that it provides accurate gradients. The generator loss is a combination of adversarial loss and perceptual loss [2].Again while in theory, this model is very robust, it didn't yield the expected results in this case which we would explore why in the upcoming chapter.

[1] A weight normalisation technique that normalises spectral norm of each layer's weight matrix, ensuring that the largest singular value of the matrix is constrained to 1 [56]

[2] This loss leverages VGG19 which was pre-trained on ImageNet to evaluate the quality of the generated images. Instead of solely, relying on pixel-wise differences this loss measures the discrepancy between the generated and real images in the feature space that is learned by VGG19 [57]

Chapter 6

Experiments and Results

As the class imbalance exists, the scope of the experiments would be what would happen before synthesizing the minority class (malignant images) and what would happen after adding the synthetic malignant images along with the existing ones to see whether it improves the performance of classification. To achieve this, we would:

1. Apply algorithms such as Logistic Regression, Support Vector Machine, Random Forest, Convolutional Neural Network, Inception V3 and EfficientNetB5. Analyse its performance on the skewed data.

2. Synthesise malignant class, also holding a degree of realism and diversity which would be defined by FID score.

3. Add the synthetic images back with the existing images to see whether it improves the performance of the models.

All the algorithms were evaluated using Precision [1], Recall [2] and F1 Score [3]

[1]Precision tells, of all the predicted positives how many are actually positive [58]

[2]Recall tells, of all the actual positives how many did the model correctly predict as positive [58]

[3]Harmonic mean of precision and recall, it balances both the scores and provides a single measure of the model's performance [58]

as the main metrics.

6.1 Data Preparation

Before conducting the experiments, it was essential to organise the dataset into a format that would be suitable for training the models. The images provided in the dataset were associated with unique IDs, with their corresponding binary labels (0 for benign and 1 for malignant) stored in a csv file. To smoothen the process of loading and utilising the images, I created a custom function that maps these IDs to their respective images. Furthermore, another function was made to create two separate directories corresponding to the class labels and finally all the images are normalised. This would help in smoother workflow in implementing the models.

6.2 Computational Resources and Libraries used

For running the experiments, I utilised both my local system and Kaggle's GPU. The specifications of my local system is:

1. RAM - 8GB
2. Graphics Card - Intel Iris XE

Several libraries were used in implementing the experimentation's such as:

1. PyTorch: To implement GAN architectures.
2. Tensorflow: To implement CNN and other pre-trained model's such InceptionV3, EfficientNetB5

3. - Scikit-Learn: To implement data splitting, evaluation metrics and machine learning models such as Logistic Regression, Support Vector Machine, Random Forest.

Other common libraries were used such as Pandas, NumPy, Matplotlib, Seaborn to import and manipulate the data.

6.3 Experimentation Set-Up

For this experimentation 300 malignant images and 3000 benign images were used and the reasons are two fold:

1. Due to computation limitations and also this set-up gives perfect imbalance for experimentation's.

2. To maintain the integrity of the experiments, 93 malignant samples and 93 benign samples were excluded from the training data to serve as a test set. These images were kept entirely unseen by the model, providing a robust evaluation of the model's performance. The remaining 300 malignant images and 3000 benign images (out of 400k benign images) were used for training the GANs, ensuring that the test set remained completely independent of the training process.

6.4 Experimentation's and Results - Without Synthesising

The performance of various Machine Learning and Deep Learning is assessed on the task of classifying benign and malignant skin lesions. The primary focus is on evaluating the precision and recall metrics for both classes, highlighting

the effectiveness of the model's in dealing with the significant class imbalance. Each of the model's implemented were selected for its unique advantages. The splitting was 70:30 were 70% is used for training and 30% is used for validation. The splitting was also stratified meaning the splitting happens in a way such that the distribution of classes reflects the overall distribution of the dataset. By this way, the splitting doesn't happen where one class is dominating but rather the distribution is as same as the class distribution. This is an essential splitting step when dealing with imbalanced datasets. The details of the implementations are outlined below:

6.4.1 Logistic Regression

Logistic Regression, a traditional Machine Learning algorithm was employed as a baseline model. The logistic regression model was implemented using 'scikit-learn' library in its default configuration. The results were a poor performance in the malignant class which is expected due to the class imbalance. For benign class, the precision and recall are 0.94 and 0.82 respectively while for the malignant class it was a precision and recall of 0.34 and 0.22 respectively.

6.4.2 Support Vector Machine

The Support Vector Machine was utilised next. SVMs are powerful classifiers that work by finding the best hyperplane that separates the data points. The SVM model was also implemented in its default configuration. The precision and recall for the benign class was 0.72 and 0.84 respectively while for the malignant class the precision was 0.81 and a very poor recall of 0.04. This model also suffers significantly as reflected by the very poor recall score for the malignant class.

6.4.3 Random Forest

Random Forest is an ensemble technique that works by utilising multiple decision trees and outputting the mode of the class for classification tasks, it is well known for its performance in imbalanced datasets. The Random Forest algorithm seemed to have worked way better than both the Logistic Regression and the Support Vector Machine model as it got a good precision and recall score of 0.87 and 0.80 respectively for benign class and 0.74 precision and a good increase of 0.57 in recall for the malignant class.

6.4.4 Convolutional Neural Network

To explore Deep Learning approaches, CNN was used. The architecture of the CNN consisted of three convolutional layers, three max pooling layers with dropouts. It was constructed with one hidden layer only. Each of the convolutional layers and the hidden layer had 'ReLU' as their activation while the output layer consisted of 'sigmoid' as its activation. 'Adam' optimiser and 'binary crossentropy loss' was used to train the model. The model was trained for 10 epochs. As expected, the CNN model gave a significantly higher performance of 0.94 (precision) and 0.92 (recall) for the benign class and 0.85 (precision) and 0.80 (recall) fo the benign class.

6.4.5 Inception V3

Inception V3 is a more advanced Convolutional network architecture that utilises mutiple convolutional filter sizes in each block. The top layers of the inception model was freezed and trained upon one hidden layer with consisting of 'ReLU'. The optimizer, loss function and epochs are same as CNN. The results are very impressive as it achieved 0.92 (precision) and 0.86 (recall) for benign class and

0.94 (precision) and 0.86 (recall) for malignant class.

6.4.6 EfficientNetB5

Lastly, EfficientNetB5, a highly optimised Deep learning model known for its efficiency in terms of both computation and accuracy was implemented. The top layers of the model was freezed and consisted of one hidden layer and every other configuration set for Inception V3 model. The model gave 0.85 (precision), 0.81 (recall) for benign classes and 0.62 (precision), 0.20 (recall) for the malignant class. This model wasn't giving a good performance on the malignant class.

Below is a summary table consisting of the scores achieved for all the models 6.1

Algorithms	**Precision, Recall for Benign**	**Precision, Recall for Malignant**
Logistic Regression	Precision: 0.94 Recall: 0.82	Precision: 0.34 Recall: 0.22
Support Vector Machine	Precision: 0.72 Recall: 0.84	Precision: 0.81 Recall: 0.04
Random Forest	Precision: 0.87 Recall: 0.80	Precision: 0.74 Recall: 0.57
Convolutional Neural Network	Precision: 0.94 Recall: 0.92	Precision: 0.85 Recall: 0.80
Inception V3	Precision: 0.92 Recall: 0.86	Precision: 0.94 Recall: 0.86
EfficientNetB5	Precision: 0.85 Recall: 0.81	Precision: 0.62 Recall: 0.20

Table 6.1: Comparison of Precision and Recall for Different Algorithm's (Without Synthesising)

6.5 Implementation of CycleGAN with Domain Adaptation

The implementation of the CycleGAN model comprised of two generators and two discriminators.

- Generator Architecture: Each Generator consists of several convolutional layers with BatchNormalisation and ReLU activation functions. The architecture is designed to capture complex features from the input images and generate realistic images in the target domain. The generator first encodes the input images through a series of convolutional layers, then decodes it using transposed convolutional layers to reconstruct the image in the target domain. Finally, the output is passed through a Tanh activation function to ensure that the pixel values are within the range [-1, 1].
- Discriminator Architecture: The discriminator is a CNN that classifies real or fake. It consists of several convolutional layers with LeakyReLU activation functions and Batch Normalisation. The output of the discriminator would be a single value between 0 and 1, stating the probability that the input image is real or fake.

The generator architecture is constructed as follows:

- Input: $3 \times 140 \times 140$ RGB image
- Conv2D $(3 \rightarrow 64) \rightarrow$ BatchNorm $\rightarrow$ ReLU
- Conv2D $(64 \rightarrow 128) \rightarrow$ BatchNorm $\rightarrow$ ReLU
- Conv2D $(128 \rightarrow 256) \rightarrow$ BatchNorm $\rightarrow$ ReLU
- ConvTranspose2D $(256 \rightarrow 128) \rightarrow$ BatchNorm $\rightarrow$ ReLU

- ConvTranspose2D (128 → 64) → BatchNorm → ReLU
- Conv2D (64 → 3) → Tanh

The Discriminator architecture is as follows:

- Input: $3 \times 140 \times 140$ RGB image
- Conv2D (3 → 64) → LeakyReLU
- Conv2D (64 → 128) → BatchNorm → LeakyReLU
- Conv2D (128 → 256) → BatchNorm → LeakyReLU
- Conv2D (256 → 512) → BatchNorm → LeakyReLU
- Conv2D (512 → 1)

The training process is followed by the implementation of adversarial loss, cycle consistency loss and identity loss. The model was ran for 1000 epochs.

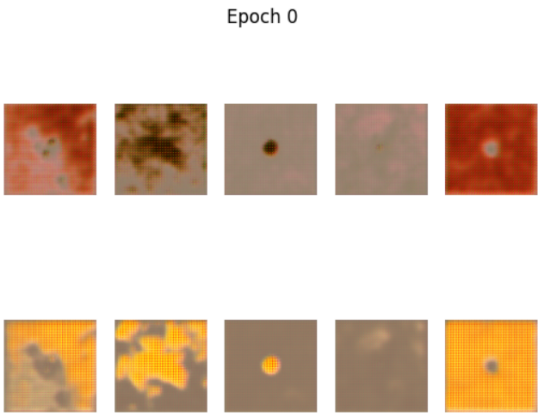

Figure 6.1: CycleGAN Epoch 0

The starting epoch 6.1 of the model seems to be a bit good, as it is able to produce some features of the image while the coloring and other attributes are missing. But for the first epoch, it is fairly acceptable.

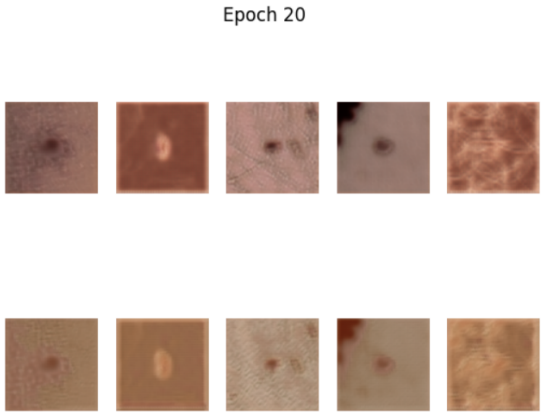

Figure 6.2: CycleGAN Epoch 20

After 20 epochs, the model seems to have really understood a lot more aspects of the images. The detailing of the skin and other elements has improved a lot. This states that the model is indeed learning in a good phase, as shown in Figure 6.2.

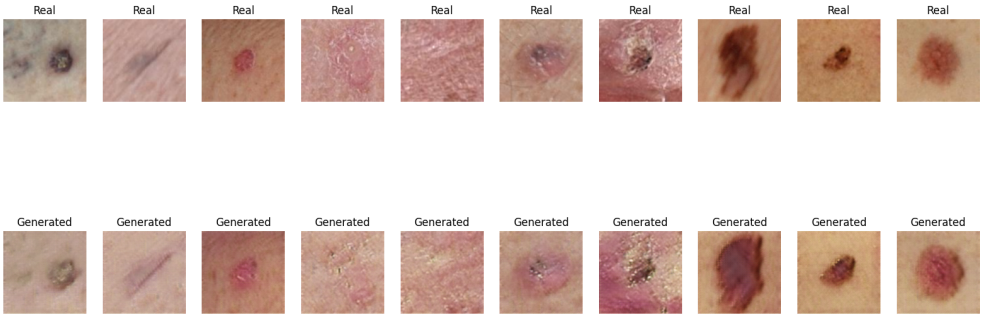

Figure 6.3: CycleGAN Epoch 1000

Fast forward to the end of the training 6.3(1000 epochs), the model has learned the features of the images and has produced really good synthetic data. To check the realism and diversity of the images generated FID score is used. The evaluation process involved generating a batch of images using the trained CycleGANs model and calculating the FID score. The FID score was computed iteratively, and image generation was stopped once the FID score dropped below a predefined threshold which is 30%. The model was able to successfully generate 2600

images with a FID score of 22%, indicating a high level of realism and diversity.

6.6 Experimentation's and Results - After Synthesising

The experimentation was conducted with the same algorithms across five phases, with a total of 2600 images. The images are divided into batches of 520 images each, In each phase, 520 synthetic images were added incrementally and the algorithms were rerun. The purpose of this approach was to assess whether the inclusion of synthetic data consistently enhances the model's performance, stabilises it or potentially leads to reduction in accuracy. All these phases are evaluated on the test set and below is the final results table.

Algorithms	**Precision, Recall for Benign**	**Precision, Recall for Malignant**
Logistic Regression	Precision: 0.96 Recall: 0.90	Precision: 0.92 Recall: 0.88
Support Vector Machine	Precision: 0.70 Recall: 0.62	Precision: 0.88 Recall: 0.36
Random Forest	Precision: 0.88 Recall: 0.80	Precision: 0.92 Recall: 0.94
Convolutional Neural Network	Precision: 0.97 Recall: 0.94	Precision: 0.96 Recall: 0.86
Inception V3	Precision: 0.88 Recall: 0.74	Precision: 0.98 Recall: 0.95
EfficientNetB5	Precision: 0.74 Recall: 0.65	Precision: 0.78 Recall: 0.62

Table 6.2: Phase 1: Comparison of Precision and Recall for Different Algorithms

Algorithms	Precision, Recall for Benign	Precision, Recall for Malignant
Logistic Regression	Precision: 0.95 Recall: 0.89	Precision: 0.93 Recall: 0.90
Support Vector Machine	Precision: 0.71 Recall: 0.61	Precision: 0.85 Recall: 0.38
Random Forest	Precision: 0.89 Recall: 0.82	Precision: 0.93 Recall: 0.95
Convolutional Neural Network	Precision: 0.96 Recall: 0.93	Precision: 0.97 Recall: 0.88
Inception V3	Precision: 0.90 Recall: 0.77	Precision: 0.97 Recall: 0.96
EfficientNetB5	Precision: 0.76 Recall: 0.66	Precision: 0.80 Recall: 0.64

Table 6.3: Phase 2: Comparison of Precision and Recall for Different Algorithms

Algorithms	Precision, Recall for Benign	Precision, Recall for Malignant
Logistic Regression	Precision: 0.94 Recall: 0.88	Precision: 0.92 Recall: 0.91
Support Vector Machine	Precision: 0.68 Recall: 0.60	Precision: 0.83 Recall: 0.36
Random Forest	Precision: 0.91 Recall: 0.83	Precision: 0.94 Recall: 0.96
Convolutional Neural Network	Precision: 0.97 Recall: 0.95	Precision: 0.96 Recall: 0.89
Inception V3	Precision: 0.91 Recall: 0.80	Precision: 0.98 Recall: 0.97
EfficientNetB5	Precision: 0.77 Recall: 0.67	Precision: 0.81 Recall: 0.65

Table 6.4: Phase 3: Comparison of Precision and Recall for Different Algorithms

Algorithms	Precision, Recall for Benign	Precision, Recall for Malignant
Logistic Regression	Precision: 0.95 Recall: 0.90	Precision: 0.93 Recall: 0.92
Support Vector Machine	Precision: 0.65 Recall: 0.58	Precision: 0.80 Recall: 0.34
Random Forest	Precision: 0.90 Recall: 0.82	Precision: 0.93 Recall: 0.96
Convolutional Neural Network	Precision: 0.97 Recall: 0.94	Precision: 0.97 Recall: 0.92
Inception V3	Precision: 0.92 Recall: 0.81	Precision: 0.98 Recall: 0.97
EfficientNetB5	Precision: 0.78 Recall: 0.69	Precision: 0.82 Recall: 0.66

Table 6.5: Phase 4: Comparison of Precision and Recall for Different Algorithms

Algorithms	Precision, Recall for Benign	Precision, Recall for Malignant
Logistic Regression	Precision: 0.96 Recall: 0.91	Precision: 0.94 Recall: 0.93
Support Vector Machine	Precision: 0.64 Recall: 0.57	Precision: 0.79 Recall: 0.33
Random Forest	Precision: 0.91 Recall: 0.85	Precision: 0.94 Recall: 0.97
Convolutional Neural Network	Precision: 0.98 Recall: 0.96	Precision: 0.97 Recall: 0.94
Inception V3	Precision: 0.93 Recall: 0.83	Precision: 0.98 Recall: 0.98
EfficientNetB5	Precision: 0.79 Recall: 0.71	Precision: 0.83 Recall: 0.68

Table 6.6: Phase 5: Comparison of Precision and Recall for Different Algorithms

The results from the incremental experimentation's across the five phases showed different effects on different algorithms. For most models, especially Deep Learning model's such as Convolutional Neural Network and Inception V3 it is observed that the precision and recall scores increases in the later phases. However, the inclusion of synthetic data has a minor reduction of performance in classifying the benign classes as it might have introduced some noises or complexities that made the model struggle at the early phases. But as more data was added, the performance increased and eventually stabilised, which shows the benefits of adding synthetic data.

Conversely, the Support Vector Machine (SVM) model struggled which is very noticeable, particularly in the malignant cases. The reasons might be that SVMs are sensitive to the distribution of the data and how well it separates the classes. The addition of synthetic data might have made it harder for the model to draw a line to separate the benign cases from the malignant ones. Provided, SVMs need proper fine-tuning to best fit with the data, since the model was ran in baseline settings it found it hard to generalise. With proper fine-tuning the model should be able to perform better. Overall, this incremental experimentation's show that the inclusion of synthetic data helps in improving and stabilising the model performance, where some model's are benefiting more than the other model's.

6.7 Revisiting WGAN-GP Architecture

6.7.1 Implementation and Performance Review of WGAN-GP

The generator network architecture consists of several convolutional layers which is interspersed with residual blocks, which help in maintaining the information

flow through the network and enable the generation of more complex image structures. The generator is using down-sampling layers ("conv2D") to reduce the image size and capture abstract features, followed by up-sampling layers ("Conv2dTranspose") to reconstruct the image back to its original size. The critic, is composed of convolutional layers consisting of LeakyReLU as its activations and spectral normalisation to stabilise the training by controlling Lipschitz of the discriminator. Alongside the generator and the critic, a perceptual model VGG19 was utilised to calculate the perceptual loss. The training happens in a adversarial way where the generator tries to fool the critic, and the critic tries to predict where the image is real or fake. The loss function used for both is the wasserstein loss, which provides a smooth gradients and tackle mode collapse. To enforce Lipschitz constriant required for wasserstein loss, a gradient penalty term is added to the discrimonator's loss function. The training process also involves alternatively updating the generator and the discriminator, with the discriminator update is been twice per generator update.

```
7/7 ━━━━━━━━━━━━━━━━━━━━ 285s 13s/step - d_loss: 1372.0082 - g_loss: 25.8255
Epoch 2/150
7/7 ━━━━━━━━━━━━━━━━━━━━ 17s 2s/step - d_loss: 646.9099 - g_loss: 23.7903
Epoch 3/150
7/7 ━━━━━━━━━━━━━━━━━━━━ 17s 2s/step - d_loss: 471.9597 - g_loss: 21.7476
Epoch 4/150
7/7 ━━━━━━━━━━━━━━━━━━━━ 18s 2s/step - d_loss: 457.2156 - g_loss: 20.4756
Epoch 5/150
7/7 ━━━━━━━━━━━━━━━━━━━━ 18s 2s/step - d_loss: 267.4865 - g_loss: 20.1247
Epoch 6/150
7/7 ━━━━━━━━━━━━━━━━━━━━ 17s 2s/step - d_loss: 237.5146 - g_loss: 18.5806
Epoch 7/150
7/7 ━━━━━━━━━━━━━━━━━━━━ 17s 2s/step - d_loss: 244.8921 - g_loss: 18.2331
Epoch 8/150
7/7 ━━━━━━━━━━━━━━━━━━━━ 17s 2s/step - d_loss: 171.6330 - g_loss: 17.3381
Epoch 9/150
7/7 ━━━━━━━━━━━━━━━━━━━━ 18s 2s/step - d_loss: 146.2991 - g_loss: 17.3885
Epoch 10/150
7/7 ━━━━━━━━━━━━━━━━━━━━ 18s 2s/step - d_loss: 147.9141 - g_loss: 15.9659
Epoch 11/150
7/7 ━━━━━━━━━━━━━━━━━━━━ 17s 2s/step - d_loss: 78.0967 - g_loss: 15.7978
Epoch 12/150
7/7 ━━━━━━━━━━━━━━━━━━━━ 17s 2s/step - d_loss: 61.7863 - g_loss: 15.5294
Epoch 13/150
7/7 ━━━━━━━━━━━━━━━━━━━━ 17s 2s/step - d_loss: 59.9602 - g_loss: 15.9457
...
Epoch 149/150
7/7 ━━━━━━━━━━━━━━━━━━━━ 17s 2s/step - d_loss: -13.0213 - g_loss: 16.5845
Epoch 150/150
7/7 ━━━━━━━━━━━━━━━━━━━━ 17s 2s/step - d_loss: -10.0455 - g_loss: 16.2564
```

Figure 6.4: Training Instability of WGAN-GP

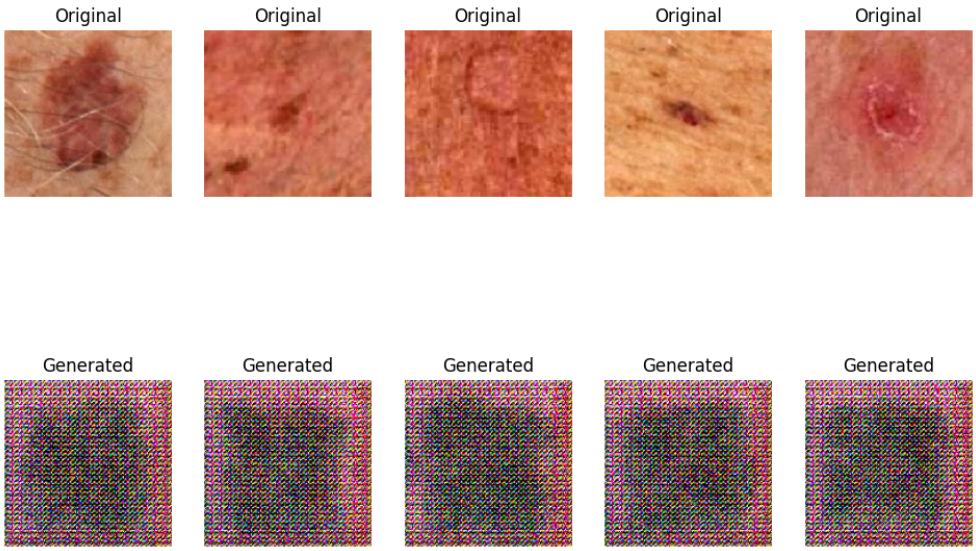

Figure 6.5: WGAN-GP Generated Images

The model was trained for 150 epochs. However as observed in the training process 6.4 and the generated images 6.5 the model was no where close to

producing a good image. The reasons might be of several folds:

1. The discriminator loss starts at a high value and gradually decreases as the training proceeds. This is expected as the discriminator becomes better at distinguishing real images from the fake ones.

2. The generator loss decreases slower rates and stabilises within the range of 15 - 16. It indicates that the generator is struggling to produce real images.

So over the course of time, the discriminator has learnt to distinguish between real and fake images while the generator is struggling to produce realistic images. The model has fallen in mode collapse where it is repeatedly producing the same outputs, in this case, noisy images. Despite the implementation of wasserstein loss and gradient penalty which is supposed to tackle vanishing gradients the model has very unstable gradients. The quality of the generated images also states that the perceptual loss isn't properly guiding the generator in producing realistic images. Perhaps with better fine-tuning such as learning rate, gradient penalty coefficient etc the model might improve. The addition of perceptual loss requires the generator to handle multiple objectives which can be challenging for the generator. Improving the generator architecture would also help.

6.8 Summary

In this chapter, a comprehensive set of experiments was conducted to assess the impact of synthetic data generation on the classification of the skin lesions. The experiments were conducted on traditional Machine Learning model's and Deep Learning model's without synthesising additional malignant images. The model's showed limitations in their performance due to the obvious class imbalance. To address this, the proposed CycleGAN with Domain Adaptation implementation

was seen which is used to generate synthetic malignant images from the abundant benign images. The quality of the generated images and the diversity of it was calculated using the FID score. The CycleGAN model was able to produce 2600 images. Then the inclusion of the synthetic malignant images as 5 phases showed that the addition of these images helps in improving the performance and also stabilise the performance for most of the models except from SVM which struggled to generalise. Finally, the WGAN-GP model was explored and despite it been a powerful architecture the model didn't yield beneficial results as it fell into a lot of shortcomings.

Chapter 7

Conclusion and Future Research

In this study, we explored the use of Generative Adversarial Networks (GANs) for generating synthetic medical images to address the challenges of data scarcity, particularly in the domain of skin lesion. The focus was on the architecture that can best produce synthetic images and also improve training data for Machine Learning and Deep Learning model's by generating the underrepresented class (malignant class). Through a series of experimentation's, two GAN architectures were explored which includes "CycleGAN with Unsupervised Domain Adaptation" and "Wasserstein GAN with Gradient Penalty (WGAN - GP)" to enhance the dataset. The implementation of CycleGAN with Unsupervised Domain Adaptation has showed significant potential in generating realistic malignant images from the benign samples, leading to providing a richer dataset for classification. Even though WGAN - GP didn't produce good results, its shortcomings were explored. The use of evaluation metrics like FID helped in validating the quality of the generated images confirming that the synthetic data was sufficiently realistic to contribute to the training of the selected model's.

7.1 Future Research

The potential future research areas include:

1. Exploration of different domain adaptation techniques as to adapting to different scenarios.

2. Exploration of different GANs architectures such as StyleGAN3, BigGAN or ProGAN as they are sophisticated architectures which is well versed in understanding the underlying structure of fine details and textures of the image.

3. In terms of data, expanding the scope of synthetic data generation to include broader ranges of medical imaging modalities such as CT scans, MRI scans. This would require adapting the techniques discussed in this thesis onto these modalities which would bring upon its own challenges.

References

[1] K. O'Shea and R. Nash, "An introduction to convolutional neural networks," 2015. vii, 6

[2] M. G. K. K. W. R. A. C. Nicholas Kurtansky, Veronica Rotemberg, "Isic 2024 - skin cancer detection with 3d-tbp," 2024. [Online]. Available: https://kaggle.com/competitions/isic-2024-challenge vii, 22, 23

[3] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," 2020. [Online]. Available: https://arxiv.org/abs/1703.10593 vii, 25, 26, 27, 28, 29

[4] A. Paproki, O. Salvado, and C. Fookes, "Synthetic data for deep learning in computer vision & medical imaging: A means to reduce data bias," *ACM Comput. Surv.*, may 2024, just Accepted. [Online]. Available: https://doi.org/10.1145/3663759 1, 9

[5] A. Pattanaik, P. Gour, S. Mishra, V. Sharma, and H. Alabdeli, "Modeling a logistic regression based sustained approach for cancer detection," in *2023 International Conference for Technological Engineering and its Applications in Sustainable Development (ICTEASD)*, 2023, pp. 262–267. 4, 5

[6] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*,

vol. 20, no. 3, pp. 273–297, Sep. 1995. [Online]. Available: http://dx.doi.org/10.1007/BF00994018 5

[7] S. Rahman, D. Siregar, R. B. Syah, H. Setiawan, A. E. Maulana, and Hamsiah, "The effective breast cancer classification with the random forest algorithm," in *2023 International Conference of Computer Science and Information Technology (ICOSNIKOM)*, 2023, pp. 1–5. 5

[8] D. Shen, G. Wu, and H.-I. Suk, "Deep learning in medical image analysis," *Annual Review of Biomedical Engineering*, vol. 19, no. 1, p. 221–248, Jun. 2017. [Online]. Available: http://dx.doi.org/10.1146/annurev-bioeng-071516-044442 6, 7

[9] Q. Li, W. Cai, X. Wang, Y. Zhou, D. D. Feng, and M. Chen, "Medical image classification with convolutional neural network," in *2014 13th International Conference on Control Automation Robotics Vision (ICARCV)*, 2014, pp. 844–848. 6

[10] D. Jha, M. A. Riegler, D. Johansen, P. Halvorsen, and H. D. Johansen, "Doubleu-net: A deep convolutional neural network for medical image segmentation," in *2020 IEEE 33rd International Symposium on Computer-Based Medical Systems (CBMS)*, 2020, pp. 558–564. 6

[11] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, p. 436–444, May 2015. [Online]. Available: http://dx.doi.org/10.1038/nature14539 7

[12] N. Siddique, S. Paheding, C. P. Elkin, and V. Devabhaktuni, "U-net and its variants for medical image segmentation: A review of theory and applications," *IEEE Access*, vol. 9, pp. 82 031–82 057, 2021. 7

[13] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," 2015. [Online]. Available: https://arxiv.org/abs/1512.00567 7, 8

[14] T. Awan, D. Khattak, and A. Mannan, "A compact cnn model for automated detection of covid-19 using thorax x-ray images," *Journal of Intelligent Fuzzy Systems*, vol. 44, pp. 1–21, 02 2023. 8

[15] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," 2014. 8

[16] M. Mirza and S. Osindero, "Conditional generative adversarial nets," 2014. 9

[17] A. Krause, P. Perona, and R. Gomes, "Discriminative clustering by regularized information maximization," in *Advances in Neural Information Processing Systems*, J. Lafferty, C. Williams, J. Shawe-Taylor, R. Zemel, and A. Culotta, Eds., vol. 23. Curran Associates, Inc., 2010. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2010/file/42998cf32d552343bc8e460416382dca-Paper.pdf 9

[18] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," 2016. 10

[19] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, "Analyzing and improving the image quality of stylegan," 2020. 10

[20] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," 2019. 10

[21] X. Huang and S. Belongie, "Arbitrary style transfer in real-time with adaptive instance normalization," 2017. 10

[22] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein gan," 2017. 11

[23] M. Arjovsky and L. Bottou, "Towards principled methods for training generative adversarial networks," 2017. 11

[24] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," 2016. 12

[25] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," 2015. 12

[26] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," 2018. 12

[27] H. Rashid, M. A. Tanveer, and H. Aqeel Khan, "Skin lesion classification using gan based data augmentation," in *2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2019, pp. 916–919. 13

[28] N. Codella, V. Rotemberg, P. Tschandl, M. E. Celebi, S. Dusza, D. Gutman, B. Helba, A. Kalloo, K. Liopyris, M. Marchetti, H. Kittler, and A. Halpern, "Skin lesion analysis toward melanoma detection 2018: A challenge hosted by the international skin imaging collaboration (isic)," 2019. 13

[29] A. Ghorbani, V. Natarajan, D. Coz, and Y. Liu, "DermGAN: Synthetic Generation of Clinical Skin Images with Pathology," in *Proceedings of the Machine Learning for Health NeurIPS Workshop*, ser. Proceedings of Machine Learning Research, A. V. Dalca, M. B. McDermott, E. Alsentzer,

S. G. Finlayson, M. Oberst, F. Falck, and B. Beaulieu-Jones, Eds., vol. 116. PMLR, 13 Dec 2020, pp. 155–170. [Online]. Available: https://proceedings.mlr.press/v116/ghorbani20a.html 14

[30] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," 2018. 14, 15

[31] A. Odena, V. Dumoulin, and C. Olah, "Deconvolution and checkerboard artifacts," *Distill*, 2016. [Online]. Available: http://distill.pub/2016/deconv-checkerboard/ 14

[32] Y. Liu, A. Jain, C. Eng, D. H. Way, K. Lee, P. Bui, K. Kanada, G. de Oliveira Marinho, J. Gallegos, S. Gabriele, V. Gupta, N. Singh, V. Natarajan, R. Hofmann-Wellenhof, G. S. Corrado, L. H. Peng, D. R. Webster, D. Ai, S. J. Huang, Y. Liu, R. C. Dunn, and D. Coz, "A deep learning system for differential diagnosis of skin diseases," *Nature Medicine*, vol. 26, no. 6, p. 900–908, May 2020. [Online]. Available: http://dx.doi.org/10.1038/s41591-020-0842-3 15

[33] S. Barratt and R. Sharma, "A note on the inception score," 2018. 15

[34] P. E. Adjei, Z. M. Lonseko, and N. Rao, "Gan-based synthetic gastrointestinal image generation," in *2020 17th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP)*, 2020, pp. 338–342. 15

[35] Z. Wang, E. Simoncelli, and A. Bovik, "Multi-scale structural similarity for image quality assessment," *Proceedings of the IEEE Asilomar Conference Signals, Systems and Computers*, 02 2004. 16

[36] M. W. Gondal, B. Schölkopf, and M. Hirsch, "The unreasonable effectiveness of texture transfer for single image super-resolution," in *Computer Vision –*

ECCV 2018 Workshops, L. Leal-Taixé and S. Roth, Eds. Cham: Springer International Publishing, 2019, pp. 80–97. 16

[37] X. Yi, E. Walia, and P. Babyn, "Unsupervised and semi-supervised learning with categorical generative adversarial networks assisted by wasserstein distance for dermoscopy image classification," 2018. 16

[38] D. Gutman, N. C. F. Codella, E. Celebi, B. Helba, M. Marchetti, N. Mishra, and A. Halpern, "Skin lesion analysis toward melanoma detection: A challenge at the international symposium on biomedical imaging (isbi) 2016, hosted by the international skin imaging collaboration (isic)," 2016. [Online]. Available: https://arxiv.org/abs/1605.01397 16

[39] C. Barata, M. Ruela, M. Francisco, T. Mendonça, and J. S. Marques, "Two systems for the detection of melanomas in dermoscopy images using texture and color features," *IEEE Systems Journal*, vol. 8, no. 3, pp. 965–979, 2014. 16

[40] M. Ali, M. Ali, and M. Javed, "Dcgan for synthetic data augmentation of cervical cancer for improved cervical cancer classification," in *2024 IEEE International Students' Conference on Electrical, Electronics and Computer Science (SCEECS)*, 2024, pp. 1–7. 17

[41] M. Frid-Adar, E. Klang, M. Amitai, J. Goldberger, and H. Greenspan, "Synthetic data augmentation using gan for improved liver lesion classification," in *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)*, 2018, pp. 289–293. 18

[42] U. Tariq, R. Qureshi, A. Zafar, D. Aftab, J. Wu, T. Alam, Z. Shah, and H. Ali, "Brain tumor synthetic data generation with adaptive stylegans," in

Artificial Intelligence and Cognitive Science, L. Longo and R. O'Reilly, Eds. Cham: Springer Nature Switzerland, 2023, pp. 147–159. 19

[43] T. Karras, M. Aittala, J. Hellsten, S. Laine, J. Lehtinen, and T. Aila, "Training generative adversarial networks with limited data," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 12 104–12 114. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2020/file/8d30aa96e72440759f74bd2306c1fa3d-Paper.pdf 19

[44] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 4396–4405. 19

[45] A. Aksac, D. Demetrick, T. Ozyer, and R. Alhajj, "Brecahad: a dataset for breast cancer histopathological annotation and diagnosis," *BMC Research Notes*, vol. 12, 12 2019. 19

[46] N. Kumari, R. Zhang, E. Shechtman, and J.-Y. Zhu, "Ensembling off-the-shelf models for gan training," 2022. 19

[47] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2010. 19

[48] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, "Analyzing and improving the image quality of stylegan," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 19

[49] T. An and C. Joo, "Cycleganas: Differentiable neural architecture search for cyclegan," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2024, pp. 1655–1664. 25, 27, 29

[50] T. Zhou, P. Krähenbühl, M. Aubry, Q. Huang, and A. A. Efros, "Learning dense correspondence via 3d-guided cycle consistency," 2016. [Online]. Available: https://arxiv.org/abs/1604.05383 27

[51] W. Liu and F. Su, "Unsupervised adversarial domain adaptation network for semantic segmentation," *IEEE Geoscience and Remote Sensing Letters*, vol. 17, no. 11, pp. 1978–1982, 2020. 30

[52] H.-Y. Chen and J.-T. Chien, "Deep semi-supervised learning for domain adaptation," in *2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP)*, 2015, pp. 1–6. 30

[53] Y. Chen, W. Li, C. Sakaridis, D. Dai, and L. Van Gool, "Domain adaptive faster r-cnn for object detection in the wild," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 30

[54] L. Jie, P. Liang, Z. Zhao, J. Chen, Q. Chang, and Z. Zeng, "Adan: An adversarial domain adaptation neural network for early gastric cancer prediction," in *2022 44th Annual International Conference of the IEEE Engineering in Medicine Biology Society (EMBC)*, 2022, pp. 2169–2172. 30

[55] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein gans," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates,

Inc., 2017. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2017/file/892c3b1c6dccd52936e27cbd0ff683d6-Paper.pdf 32

[56] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral normalization for generative adversarial networks," 2018. [Online]. Available: https://arxiv.org/abs/1802.05957 33

[57] Y. Liu, H. Chen, Y. Chen, W. Yin, and C. Shen, "Generic perceptual loss for modeling structured output dependencies," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021, pp. 5424–5432. 33

[58] H. Dalianis, *Evaluation Metrics and Evaluation*. Cham: Springer International Publishing, 2018, pp. 45–53. [Online]. Available: https://doi.org/10.1007/978-3-319-78503-5_6 34

Appendix A

Appendix-A- GitHub Repository

All the code for the thesis is hosted in GitHub [1]. The repository is structured as three folders:

1. Without Synthesising: This folder consists of the 6 Machine Learning and Deep Learning models as .py files.
2. After Synthesising: Within this folder there are 6 other folders for every algorithm used for each phases.
3. GAN Architectures: This folder consists of the code for CycleGAN with Unsupervised Domain Adaptation and WGAN-GP.

[1] https://github.com/NJ2504/Synthetic-Data-Generation-for-Medical-Images-Using-GAN