

Unsupervised Domain Adaptation for Medical Images with VGG-19, and EfficientNet Feature Extractors

Progress Presentation Report

Sai Akshay Suresh
a1906525

August 20, 2025

Report submitted for **Data Science Research Project Part A** at
the School of Mathematical Sciences, University of Adelaide



THE UNIVERSITY
of ADELAIDE

Project Area: **Domain Adaptation**
Project Supervisor: **Xinyu Zhang**

In submitting this work I am indicating that I have read the University's Academic Integrity Policy. I declare that all material in this assessment is my own work except where there is clear acknowledgement and reference to the work of others.

I give permission for this work to be reproduced and submitted to other academic staff for educational purposes.

Abstract

In this study, an Unsupervised Domain Adaptation (UDA) for medical images with different scan modalities of different parts in the body has been performed. The source has six classes of 780 images, where the dataset is balanced and fully labelled, and the target dataset also has the same 6 balanced classes of 780 fully unlabelled images, but obtained from a different institution than the source and has a different distribution. In this research, the source has been pretrained on Convolutional Neural Network (CNN) architectures such as Visual Geometry Group-19 (VGG-19), and EfficientNet's baseline model (B0), and Unsupervised Domain Adaptation has been performed on target dataset with Generative Adversarial Network (GAN) variants of Domain Adversarial Neural Network (DANN), and Adversarial Discriminative Domain Adaptation (ADDA) by having the CNN models as feature extractors initialized by learning weights (as .pth files), to extract the knowledge from the source. The source pretraining has shown the performances of VGG-19 and EfficientNet-B0 with the variations in the usage of hyperparameters, namely epochs [5,10,20,40,80] and optimizers [SGD, Adam, RMSProp, AdaGrad, AdamW]. The Unsupervised Domain Adaptation performed using DANN was tested with the hyperparameter, lambda, of five variants [0.01, 0.05, 0.1, 1], and ADDA was tested with learning rate as the hyperparameter of five variants [1e-3, 5e-4, 2e-4, 1e-4, 5e-5]. The DANN with VGG-19's feature extractor has effectively identified 467 out of 780 images in the target domain based on the understanding of the source domain, followed by other variants.

1 Introduction

Transfer Learning (TL), in which knowledge obtained from one task is applied to another task, e.g., from binary-class classification to multiclass classification [31]. Domain Adaptation (DA), a subset of TL, uses the knowledge obtained from one fully labelled dataset (source) to identify the images on another partially labelled or unlabelled dataset (target), which is of the same domain as the source but has a different distribution [34]. For example, the source has images of a few car models taken in daylight, and the target has images of the same model classes of cars taken at night.

Medical images often come with a bottleneck of suffering from domain distribution shifts due to variations in scanning devices used in different

institutes and acquisition mediums. Such shifts experience deterioration in the detection of images across different domains [14]. Domain Adaptation is vital to align the heterogeneous distributions of domains and promote identification of unlabelled images which share the same classes as the labelled images [18]. Getting the correct classes of unlabelled medical images is costly, time-consuming and often hindered by privacy laws of institutions. DA enables unsupervised learning from the labelled source data and serves as a medium when labels are unavailable in a target dataset to identify [7]. In tasks like cross-modality segmentation (e.g MRI \leftrightarrow CT), a normal image translation could result in distortion of anatomical structures [5]. DA ensures the structures are preserved, with the usage of loss techniques like adversarial loss, cross-entropy loss, etc.

These loss techniques can be effectively applied with the help of Generative Adversarial Networks (GANs). GANs offer a valuable setup for transforming images from the target domain into the style/structure of source domain images [9]. GANs help preserve class labels, so the image classifiers trained on the source will recognise them. This is the adversarial image translation that aids in retention of meaningful features across different domains, even when acquisition means like contrast, resolution, etc. differ significantly [16]. GANs can work in two modes: by modifying/generating images, and by modifying/generating features. This study focuses on the usage of GANs in feature-based mode. This is because feature generation techniques for GANs are computationally cheaper than image generation, and extract more necessary information from the source to generate the target features to look like the source. By generating highly source-matching pseudo features in the target domain, the GANs allow labelled source classes annotations to be reused for detection [4]. Moreover, GAN-based adversarial training for DA encourages domain-invariant (same) structures to be identified even in the case of a distribution difference, ensuring that source-learned features apply well across domains, and boosting the capability of the classifier [9].

As a part of this study, two GAN variants-Adversarial Discriminant Domain Adaptation (ADDA), and Discriminant Adversarial Neural Network (DANN) were primarily used for Unsupervised Domain Adaptation (UDA) with the help of two CNN feature extractors-VGG-19, and EfficientNet-B0. ADDA uses the principle of discriminative modelling, where the adversarial training process happens to confuse a domain discriminator. This makes the features extracted from the source as indistinguishable to those of the target, making the model learn the domain-

invariant features [21]. Whereas, DANN relies on training to make target features look like the source, while parallelly making the classifier classify data in the source domain [18]. Despite having almost the same principles, the difference between ADDA and DANN is that ADDA performs the tasks sequentially, while DANN runs simultaneous tasks. Both ADDA and DANN can operate without the labelled target samples, making them as a practical option in the medical field under difficult conditions, where adaptation is crucial for real-world diagnosis. By aligning features rather than images, both ADDA and DANN have proven beneficial for classifying scans of the same, cross-modalities obtained in different medical centers or scan types for a consistent performance in various clinical settings [19].

From various studies, pretrained CNNs like VGG-19 or EfficientNet-B0 have been helpful in the extraction of features, from textures to complex structures, without compromising on relevant details, making them preferable backbones [20]. When the GANs are combined with the CNN-based pretrained feature extractors (initialized by learned weights saved as .pth checkpoints), the source features are learnt precisely to make them useful for adaptation with the target domains without modifying the existing feature space [2]. Our research uses VGG-19 as a feature extractor, which is reputed to have a long-lasting memory, especially in medical settings, and EfficientNet-B0 for its computationally efficient feature extraction. Using these two CNN models as feature extractors leads to strong initial points for adaptation, as the feature extractors serve as the brain and eyes for the GANs [23]. GANs with feature generators encourage the model to produce representations similar to almost the same representations from the target that are closer to the source. Coupling GANs with the CNNs helps the models trained on one modality or center to generalize well to different visual distributions [26]. This widely accepted framework emphasizes scalability and adaptability in medical image processing by the extraction of both high and low-level feature nuances effectively from variations in anatomical structures [8].

The combination of CNN feature extractors with feature-based GANs enables an option where the feature extractor and GAN components can be refined independently. Therefore, this experimental report focuses on the pretraining phases of the CNN architectures-EfficientNet-B0, VGG-19, and provides an emphasis on domain adaptation with GANs-ADDA, DANN, with the usage of pretrained CNN architectures in feature extractions aided by learned weight checkpoints. Due to VGG-19's memory saving capacity, it was observed that the DANN architecture with

VGG-19 feature extractor has achieved a highest accuracy of 60 percent approximately in domain adaptation, detecting the unlabelled target domain data. And, EfficientNet-B0 with DANN follows this performance with almost achieving a 46.5 percent highest accuracy for domain adaptation with the same VGG-19 for the same task.

2 Background

2.1 Challenges in Domain Adaptation

A very common problem that exists in Domain Adaptation is Negative Transfer, where the DA-specific models tend to misclassify the images, which degrades the performance on the target domain despite learning the source data's features. The paper [37] highlights that feature-alignment via naive adversarial training could miss class-level distinct information, leading to poor generalization and misalignment of domains for medical imaging tasks. Another work [33] focused on image segmentation with prototype alignment reports that in despite the same classes, the intra-class variability in features (common in medical imaging) can make the learning difficult and result in degraded segmentation accuracy. One more case [11], which deals with unsupervised domain adaptation for lesion scale (abnormal/damaged tissue scans). The paper reports that a simple, thoughtless alignment of features can cost significantly. When the source's representations are irrelevant or difficult to understand, the learning process becomes poor, thus resulting in undesirable prediction accuracy despite the presence of shared classes in both domains, as evidenced by the paper, where the difference in lesion proportions between the two domains leads to negative transfer.

A problem that contributes severely to negative transfer, where UDA-based methods suffer in medical imaging, is semantic misalignment and anatomical inconsistency during adaptation. This problem occurs even when source and target domains share labels; appearance-based transformations to the target data can distort class-specific structures or blur class boundaries. This paper [36] highlights in Medical Image Analysis, where GAN-based translations of features introduced visual discrepancies and style mismatches despite a real appearance. Such distortions misled the study of the segmentation model into fixing wrong patterns, thus demonstrating that segmentation performance degrades not only because of label mismatch, but the structural boundaries- like organ edges or lesion textures-can also lead to a misalignment of domains during adaptation.

2.2 Related Works

The paper [32] applies UDA for digit classification, traffic signal classification, and general object recognition tasks with DANN. The paper introduced a novel dual-module network model to foster stronger domain-invariant feature learning. The adversarial loss function is used to maximize feature distribution discrepancy and minimize prediction result errors. The paper's proposed model significantly outperformed other widely recognized models with an improved accuracy of above 97 percent for digit and signal classification and an average accuracy of 69.2 percent for general object recognition with the grand VisDA-2017 dataset. This model also comes with a disadvantage in the increment in adversarial training time required compared to single-module systems.

A work based on the detection of unseen medical images [30] involved using the GANs-StarGAN with Cycle-Consistent Adversarial Domain Adaptation (CyCADA) for maintaining semantic consistency. The standard CNNs, such as LeNet, DenseNet, were used in the classification of digits and the detection of lung opacities scans (dense/dark area of chest) from multiple centers as source/target domains. This proposal achieves a 35 percent effective AUC for digits, and 25 percent for chest X-rays. It has suffered from the computationally expensive training cost of GANs, with the requirement of hardware such as GPUs, often absent in the healthcare field.

A paper [35] which highlighted the use of DANN for colon cancer diagnosis and COVID-19 diagnosis of whether it is pneumonia or COVID-19, and general images (Office-31 dataset) in an unsupervised domain adaptation setting. The paper illustrated the mitigation efforts of DANN which suffers from ignoring label noises. The DANN has achieved accuracies in the range of 65.5 to 73.1 percent in colon cancer detection, and showed a 75.9 percent accuracy in COVID-19 diagnosis. For Office-31 general images DANN recorded an accuracy of 76.9 percent.

Another paper [22] uses an ADDA-based adversarial domain adaptation approach for a UDA on chest X-ray classification. The task uses CheXpert as the source and ChestX-ray14 as the target domains. This paper has focused on UDA for multi-label classification. It was noticed that the ADDA baseline has achieved a 0.51 average AUC, while the adversarial domain adaptation variant of ADDA with a conditional adversarial discriminator has achieved an average AUC of 0.79 approximately. The paper has shown that the drawback of ADDA is that for a multi-label discriminator, which lacks class awareness, can conflict with

the actual classification task and could potentially undermine feature discriminability. One more paper [7], which uses EfficientNet-B2 as for transfer learning under the domain adaptation framework for classifying distribution variations within Brain Tumors (MRI), and Lung diseases based on Chest X-ray scans, and requires low computational cost. It has recorded an exceptional accuracy of above 97 percent in both brain tumor and lung disease X-ray scan detections. It has also presented a bottleneck caused by varied devices and procedures in the procurement of scans, which hinders the model’s generalization.

3 Methods

3.1 Workflow

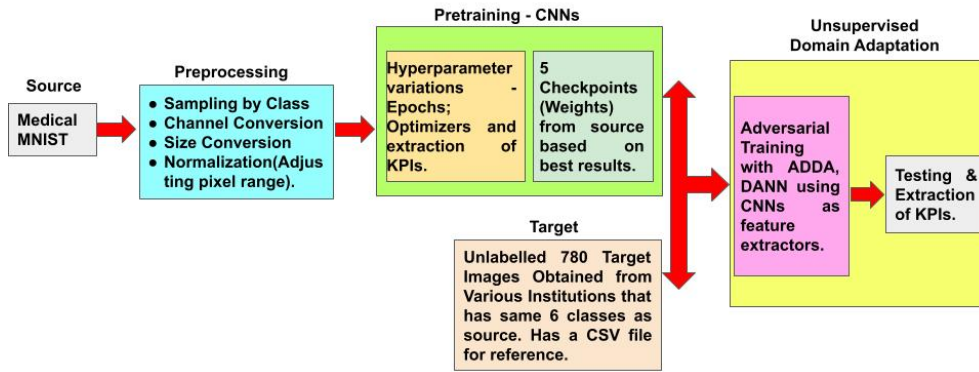


Figure 1: Project Workflow

As shown in Figure 1 above, the Medical MNIST dataset used as the source (fully-labelled), obtained from a Kaggle page with 58,954 images, has been sampled to 780 images in total with stratification by class to have 130 images per class. Preprocessing strategies like channel conversion, image size conversion, and normalization have been done to perform pretraining on CNN architectures-VGG-19, and EfficientNet-B0, with the variations of epochs and optimizers for freezing 5 valuable checkpoints/learned weights with the highest scores in terms of performance metrics, along with a consideration of not choosing the overfitted checkpoints. These checkpoints are named by optimizer and epoch, and were used as pretrained learned weights on the GANs that uses the insights

from the source-features and connectivity patterns via adversarial feedback to yield domain-invariant representations that improve prediction quality in the target domain with the knowledge and class labels of the source domain. After choosing 5 checkpoints (saved as .pth files) and initializing the CNNs with them, the UDA has been performed on ADDA, and DANN with the insertion of the target domain dataset (fully unlabelled), which also has the same 6 classes and 130 images per class as the source, accounting for a total of 780 images, but has a structurally different distribution to the source and was obtained from different Kaggle pages. Preprocessing has also been performed on these target images. To analyse the results of UDA with ADDA and DANN, a few performance indicators on the target domain were evaluated with a CSV file that has the image names and class names for each image (same as source) as two columns.

3.2 Convolutional Neural Networks for Source Pre-training and Feature Extraction

VGG-19, a deep convolutional neural network, introduced by the Visual Geometry Group of Oxford University in 2014, with the intention of improving image recognition performance, on the large ImageNet dataset that uses many small and consistent convolutional layers instead of varying large filters. In our study, VGG-19 has approximately 139.59 million training parameters, indicating a huge complexity to be introduced in pretraining and feature extraction. The architecture used for this study had 19 total layers, where 16 of them were convolution layers to learn the visual features, while three of them were fully connected layers at the end for image classification. Every convolution layer used small 3x3 filters, making the architecture simpler and powerful. Pooling layers compressed the image information using max-pooling of 2x2 steps, to reduce the size, but preserving important information. At each step, by default "Relu" activation function was kept to learn the complex pattern in the data. Finally, three dense fully connected layers make the final prediction, up to 1000 image classes, with the usage of SoftMax to assign output probabilities for predictions [3]. This design is built based on the original design [25].

EfficientNet-B0, introduced in 2019 by researchers at Google AI, achieves better accuracy with fewer resources by using a smarter scaling method, rather than making networks deeper or wider. In our study, EfficientNet-B0 had 7.1 million training parameters, making it extremely cheaper than VGG-19. EfficientNet-B0 used balanced scaling/compound

scaling to scale the size, depth, and input resolution instead of increasing them, resulting in a powerful model. It stacks MBConv blocks, taken from MobileNetV2 that use inverted residuals and lightweight building units. Squeeze-and-Excitation (SE) is also used by this model, which aids the network in focusing on the most important attributes inside each image. The architecture used in this study, built based on the original design [27], starts with a simple convolution, passes through several blocks of MBConv and SE, and then ends with a final convolution, global pooling, and a classifier head for image detection [12].

Hyp_name	Variants
Optimizers	SGD, Adam, RMSprop, Adagrad, AdamW
Epochs	5, 10, 20, 40, 80

Table 1: Hyperparameters used in CNNs for Pretraining

As shown in Table 1, for each of the optimizers, each of the epoch variants has been tested. The optimizers are the algorithms that help machine learning models by adjusting their internal settings to minimize losses. The SGD optimizer, a steady generalizer with a simple method that uses learning rate steps. Adam, chosen for its fast converging rate and for accuracy improvement. The RMSprop, picked up for its smooth stabilization quality. And Adagrad, AdamW were chosen due to their learning rate adaptations, and weight decay-friendly nature, which can reduce overfitting [6]. And, the epochs of five variants were chosen for their effective, intense learning abilities with tuning.

3.3 Generative Adversarial Networks for Unsupervised Domain Adaptation

To understand the working of the Generative Adversarial Network (GAN) Variants in this study, the Figure 2 as shown below, has the components-Feature Extractor (CNNs), Adapter, Discriminator, and a Classifier. After the extraction of image features from the source dataset, the feature extractors, initialized by pretrained source checkpoints, pass the information to the discriminator, which also absorbs the target domain data (unlabelled) via an adapter. The adapter and discriminator play a cat-and-mouse game, where the adapter, or generator, generates features of target domain data based on its understanding of the source domain images. The discriminator, which has learnt well the source domain features, tries to verify if adapter-generated target domain features align or adapt well with the source domain features. If the features do

not align, the discriminator sends feedback to the adapter, called adversarial feedback, for correction, and the adapter again modifies/generates features of the target domain with adversarial training in a loop. Whenever the adapter generates foolproof features of the target domain images to look like the source, the discriminator passes this information to the classifier to predict the images [15].

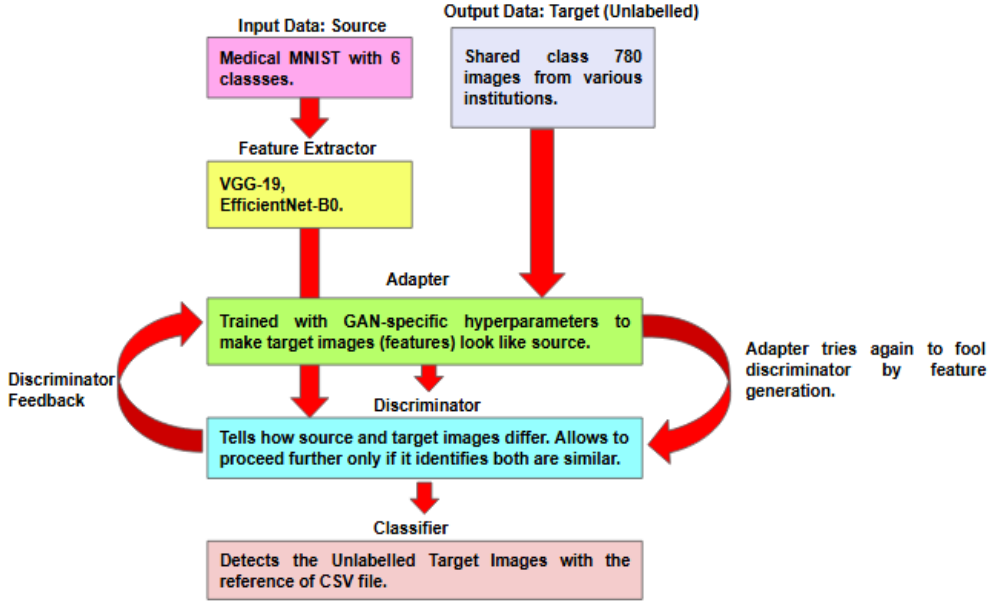


Figure 2: A general GAN Architecture for Domain Adaptation used in the Study.

As shown in the Figure 3 below, the unsupervised domain adaptation with Domain Adversarial Neural Network (DANN) has been performed with the usage of three main components-shared encoder (instead of adapter), discriminator, and task classifier. Once the feature extractor is initialized with source features, the shared encoder, which takes the unlabelled target data, is connected to the discriminator, for adversarial training when feedback is given to generate/modify the target features. The discriminator uses a Gradient Reversal Layer (also known as GRL) to align the target features with source features, and is useful for the adversarial feedback. And this DANN architecture is built based on the original work [10]. It is to be noted that the encoder learns source features that aid in the classification task to generate domain-invariant features from the target, and this is because the GRL flips the discriminator's directions (gradients) during adversarial feedback.

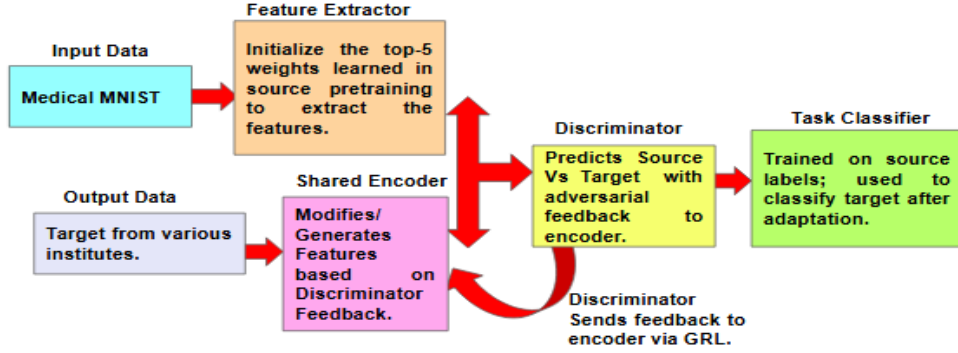


Figure 3: Domain Adversarial Neural Network (DANN) Architecture with CNN-based Feature Extractor

This process is given in Equation (1) below in terms of loss denoted by \mathcal{L} . The source-based classification happens first in the feature extractor initialized by checkpoints (seen by the first term) for learning the source features well, and the discriminator measures the differences between source and target (second term), sending signals back to the shared encoder with the help of GRL. For clarity, the discriminator assigns label 0 to the source and 1 to the target. The overall loss \mathcal{L} is the Cross-Entropy loss (CE)—measuring the difference between true labels and the model’s predicted probabilities.

$$\mathcal{L} = \underbrace{\mathcal{L}_{\text{cls}}^{\text{src}}}_{\text{source classification CE}} + \underbrace{\mathcal{L}_{\text{dom}}^{\text{src+tgt}}}_{\text{domain CE on [source=0, target=1]}} \quad (1)$$

This framework enables unsupervised domain adaptation despite the absence of target labels during training. To study the result of DANN in different scenarios, the lambda hyperparameter, which is the GRL’s scaling factor to adjust the influence of the domain discriminator on the encoder, is taken with variations from a low value of 0.01 to a high value of 1 [24]. There were 5 variants of lambda, [0.01, 0.05, 0.1, 0.5, 1.0] were studied for the DANN-based domain adaptation on each of the five source pretrained checkpoints with the VGG-19 and EfficientNet-B0.

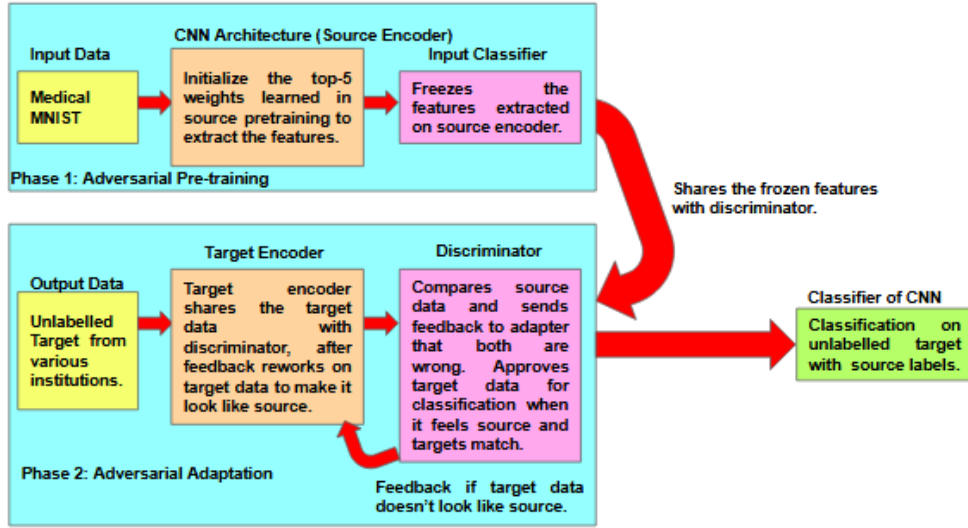


Figure 4: Adversarial Discriminant Domain Adaptation (ADDA) Architecture with CNN-based Feature Extractor

Adversarial Discriminant Domain Adaptation (ADDA), as seen in the Figure 4 above, uses a sequential flow, unlike DANN, which uses a parallel pipeline, and ADDA is built based on this paper [29]. It has two significant phases, where, in the first phase, adversarial pretraining for freezing the features happens with the source encoder (feature extractor) of the source labelled data, and, in the second phase, discriminator will enable adversarial adaptation by sending adversarial feedback to the target encoder (adapter), which modifies the target data features based on the feedback. After once discriminator ensures that both source and target features generated by the target encoder match, classification on the unlabelled target data happens with the CNN-based classifier, which knows the source image class labels. It is to be noted that the classifier is not a part of the adversarial adaptation loop.

The ADDA's working for two phases is given below in Equation 2 for the adversarial pretraining on the source, and in Equation 3 for domain adaptation in the target [28]. In the Equation 2, the training on the source encoder, denoted by ϕ_S , and classification F on labelled source data (X_S, y_S) in the pretraining phase for learning and freezing the source features is shown. Equation 3 shows the alignment process (feature generation/modification) of target encoder ϕ_T with source representations based on adversarial feedback received from the discriminator D . The target encoder, trying to fool the discriminator, is shown with $D(\phi_T(X_T))$ making the target features look like the source. Meanwhile, the discriminator learns to distinguish between the source and target en-

codings using adversarial loss \mathcal{L}_{adv} with the knowledge of source features.

$$\min_{\phi_S, F} \mathcal{L}_{\text{task}}(F(\phi_S(X_S)), y_S) \quad (2)$$

$$\begin{cases} \min_{\phi_T} \mathcal{L}_{\text{adv}}(D(\phi_T(X_T)), \text{Source}) \\ \min_D \mathcal{L}_{\text{adv}}(D(\phi_S(X_S)), \text{Source}) + \mathcal{L}_{\text{adv}}(D(\phi_T(X_T)), \text{Target}) \end{cases} \quad (3)$$

In order to check the performance of ADDA in many instances, the learning rates [1e-3, 5e-4, 2e-4, 1e-4, 5e-5] have been checked for each of the checkpoints obtained from source pretraining on CNN earlier. Higher rates (e.g., 1e-3) were chosen to see faster but less stable adversarial alignment, whereas the lower rates (e.g., 5e-5) were checked for a smoother and better alignment between source and target domains. This tuning has been done to identify a more precise balance for effective feature-based domain adaptation [15].

4 Results

4.1 Data and Preprocessing

The source and target body part scan datasets, with the classes - Abdomen CT, Breast MRI, Chest CT, Chest X-ray (CXR), Hand X-ray, and Head CT, as seen in Figure 5, have differences in their colour backgrounds and the machines from which the scans are obtained. With the common structural similarities, such as limbs attached to fingers for the hand, are studied from the source domain by the GANs-DANN, and ADDA to classify the same class images in the target domain. The source domain dataset classes, especially Abdomen CT and Chest CT, were in low resolutions. And fewer classes also have minute anatomical misalignments, like thicker and broader images of Breast MRI, and different poses of Hand X-ray in the target than in the source, etc.

As seen in Table 2 below, the source dataset obtained from the same Kaggle page has had a uniform distribution of data throughout all six classes with the size 64x64 and a grayscale channel as denoted by 1.

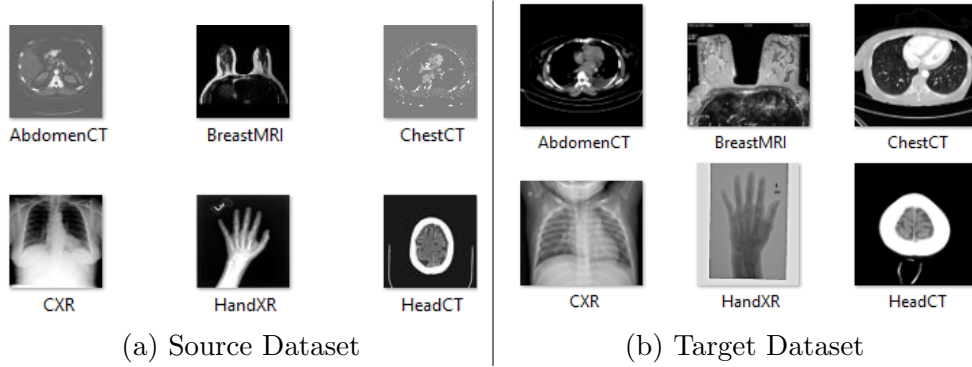


Figure 5: Side-by-side Preview of Differences between Source Dataset (a) and Target Dataset (b).

Src_Cls	Nos	S/C	Tgt_Clas	Nos	S/C
AbdomenCT	130	64x64x1	AbdomenCT	130	512×512x3
BreastMRI	130	64x64x1	BreastMRI	130	Var, Chan:1,3
CXR	130	64x64x1	CXR	130	Var, Chan:1
ChestCT	130	64x64x1	ChestCT	130	512×512x1
HandXR	130	64x64x1	HandXR	130	Var, Chan:1,3
HeadCT	130	64x64x1	HeadCT	130	512×512x1

Table 2: Data Summary of Source (Columns 1,2,3) and Target Dataset (Columns 4,5,6) before Preprocessing

The target dataset, obtained from various Kaggle pages, was seen to have different sizes and channels throughout all six classes. In rows 2, 3, and 5 of the target dataset columns on the right side of Table 2, as there was no single uniform image size for all the images, only the channels were mentioned, and the various image sizes were denoted with "Var". It was observed that the target dataset had both grayscale and RGB channels, denoted by numbers "1" and "3". As mentioned earlier, the number of images considered in this study for both source and target datasets was 130 per class, making 780 images in both source and target datasets present separately, adding a total of 1560 images considered.

All the images in the source and target have been resized and channel converted to 224x224 with RGB channels denoted by "3" to suit the requirements of VGG-19, and EfficientNet-B0 [17]. After conversion, both the source and target datasets were normalized (adjustment of pixel range to standard size from 0 to 1), for better learning [1]. As it is generally expected to have a post-normalization mean of 0 and a standard deviation of 1 for the whole dataset, the statistics of normalization before

Dataset	Norm_Type	Mean	Std.Dev
Source	Pre_WholeDS	0.354	0.280
Source	Post_1Batch	0.314	0.341
Source	Post_WholeDS	3.19e-08	1
Target	Pre_WholeDS	0.255	0.290
Target	Post_1Batch	-0.427	0.833
Target	Post_WholeDS	-5.67e-08	1

Table 3: Pre and Post-Normalization Summary Statistics of Source (Rows: 2,3,4), and Target (Rows: 5,6,7)

and after the transformation are displayed in Table 3 in terms of standard deviation and means for one batch and the entire dataset, denoted by "1Batch" and "WholeDS". A batch size of 64 was taken for normalization in both the source and target.

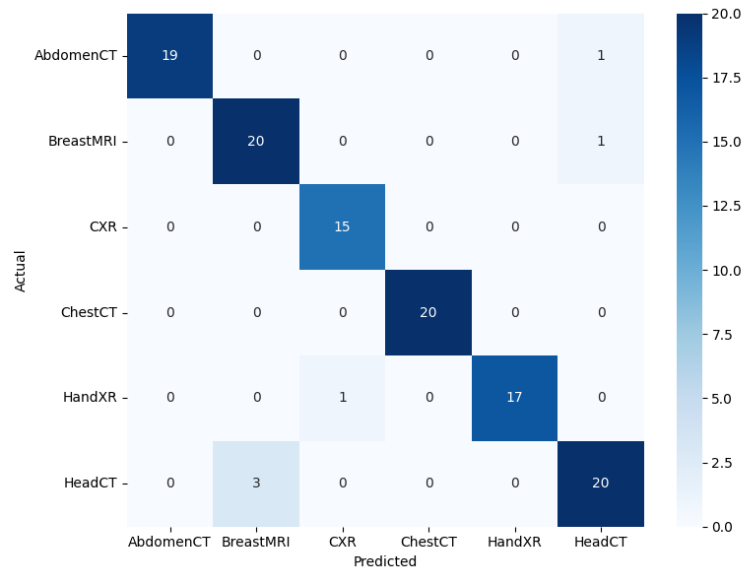
4.2 Setup and Evaluation Metrics

The experiments were conducted in Google Colab on L4 GPU (Colab Pro) with the following processing capacity [13], as shown in Table 4 below-

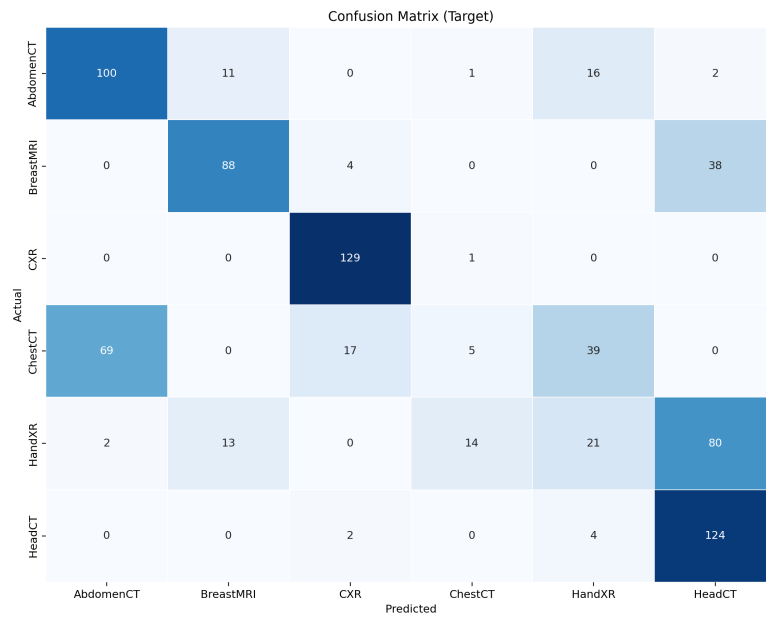
Specification	Value
GPU Architecture	NVIDIA Ada Lovelace (AD104)
CUDA Cores	7,424
Memory	24 GB GDDR6 VRAM
Memory Bandwidth	≈ 300 GB/s
TDP	72 W
FP32 Performance	≈ 30.3 TFLOPS

Table 4: Core specifications of the NVIDIA L4 GPU (used through Google Colab Pro)

Both the DANN and ADDA GANs were trained with 10 epochs on the Adam optimizer. In the source pretraining phase, the Medical MNIST dataset was divided into three sets: 75 percent for Training, 10 percent for Validation, and 15 percent for Testing. The confusion matrices obtained were used in the extraction of most of the performance metrics in both the source pretraining (test set) and domain adaptation phases (target set), as shown below in the Figure 6.



(a) Confusion Matrix of VGG-19 Source Pretraining with RMSPProp Optimizer and Epoch 40 on Test Set



(b) Confusion Matrix of Domain Adaptation with VGG-19 on DANN with 0.05 Lambda on Target Data

Figure 6: The Confusion Matrices of (a) Source Pretraining on Test Set and (b) Domain Adaptation on Target Data

The following performance metrics were obtained for the source pre-training and domain adaptation with GANs in Python with the sklearn library, along with the use of Confusion Matrices, as shown in Figure 6.

- **Training and Testing Accuracies (only Source Pretraining):** The number of images correctly predicted on total number of images on the source's training set (75 percent) and testing set (15 percentage).
- **Classification Rate (only Domain Adaptation):** The number of images in the target set correctly predicted on total number of images by the classifiers of the GANs.
- **Precision (both Source Pretraining and Domain Adaptation):** A measure of the number of true positive predictions made on the total number of positives in the source's testing set (15 percent) and target dataset.

$$\text{Precision} = \frac{TP}{TP + FP}$$

- **Sensitivity/Recall (both Source Pretraining and Domain Adaptation):** A measure of the correct number of true positive predictions made on the total number of correct predictions in the source's testing set (15 percent) and target dataset.

$$\text{Sensitivity} = \frac{TP}{TP + FN}$$

- **Specificity (both Source Pretraining and Domain Adaptation):** A measure of the correct number of true negative predictions made on the total number of true negatives and false positives in the source's testing set (15 percent) and target dataset.

$$\text{Specificity} = \frac{TN}{TN + FP}$$

- **AUC (both Source Pretraining and Domain Adaptation):** A measure of Sensitivity against the False Positive Rate (FPR) at various classification thresholds according to epoch runs on the source's testing set (15 percent) and target dataset.

4.3 Results of Source Pretraining and UDA with VGG-19

4.3.1 Results of VGG-19 Source Pretraining

Opt-Eph	Tr_ac	Te_ac	Prec	Sens	Spec	AUC
Adagrad-40	100%	100%	1	1	1	1
Adagrad-20	100%	99.15%	0.991	0.989	0.998	1
RMSprop-40	98.63%	99.15%	0.992	0.991	0.998	1
SGD-20	95.56%	94.02%	0.949	0.939	0.988	0.995
AdamW-5	91.11%	92.31%	0.927	0.927	0.985	0.999

Table 5: 5 Checkpoints Obtained from Source Pretraining on VGG-19 based on Performance (Stratified by Testing Accuracy)

As observed in the Table 5 above, the 5 best performing checkpoints were carefully chosen for VGG-19 source pretraining, which has the reduced risk of over-fitting/under-fitting. The SGD optimizer was seen to be increasing the testing accuracy from 52 percent to 88 percent with the increment of epochs from 5 to 80. The Adam optimizer has suffered an underperformance to SGD on two occasions (epochs 10 and 40) with a very low testing accuracy of 19 percent and 14 percent. The specificities were always higher than the sensitivities for Adam. The RMSProp, in all the cases, has had a testing accuracy of above 70 percent, reaching a maximum of around 99 percent with the increment of epochs from 5 to 40, and the sensitivity, and specificity were in the range of 0.65 to 0.98 in all the cases of RMSProp. As demonstrated in the Table 5 above, Adagrad with 40 epochs has emerged as a winner by having perfect scores in all aspects, followed by a marginal reduction in Adagrad with 20 epochs, which was on par with RMSProp with 40 epochs in terms of testing accuracy, sensitivity, and specificity. Almost all the variants of hyperparameters have had AUC-ROC scores above 0.9, except for a few instances with 0.5 (Adam with 80 epochs, AdamW with 10 epochs, and AdamW with 80 epochs). All the learned weight checkpoints (saved as .pth files) occupied a memory of 532.5 MB on the Google Drive storage. And this is a result of having a higher number of training parameters in VGG-19, indicating a costly and longer training duration in both the pretraining and domain adaptation with GANs.

4.3.2 Results of UDA with DANN

The Table 6, as shown below, has demonstrated the best performances of the DANN architecture for each of the checkpoints extracted from

Ckpt	Lambda	Cl.Rate	Prec	Sens	Spec	AUC
AdamW_5	0.05	59.87%	0.538	0.599	0.920	0.820
Adagrad_40	0.05	57.56%	0.492	0.576	0.915	0.799
SGD_20	0.1	55.77%	0.524	0.558	0.912	0.810
Adagrad_20	1	52.82%	0.480	0.528	0.906	0.870
RMSprop_40	0.01	49.62%	0.443	0.496	0.899	0.790

Table 6: Checkpoint Wise Best Performances - DANN with VGG-19 Feature Extractor (Stratified by Classification Rate)

source training on VGG-19. The checkpoints in the Table are represented as an optimizer along with the epoch value (like SGD_20). AdamW-5 epochs showed a peak classification rate of almost 59.87 percent with precision and sensitivity scores of 0.538 and 0.599. The AdamW-5 checkpoint has had an average classification score of around 40 percent across various lambda values. The Adagrad-40 epochs follow the previous result by a peak classification rate of 57.56 percent in identifying the target domain images correctly. Despite a marginal drop, Adagrad-40 has had a better average accuracy among various lambdas of 42.74 percent. The SGD-20 follows Adagrad-20 with a peak image classification rate of 55.77 percent, with slightly improved precision and sensitivity value. The average classification rate among various lambdas was 34 percent for the SGD-20 checkpoint. The Adagrad-20's peak has had marginally lower performance in all aspects, including classification rate, with 52.82 percent, and a slightly higher AUC of 0.87. The Adagrad-20 has had an average classification rate of 41.3 percent among all the lambdas. The RMSProp-40 has shown the maximum classification rate of 49.62 percent, along with marginally lower values in all the metrics, with an average classification rate of 37.31 percent across all the lambda values.

It is to be noted that despite having good specificity scores, the DANN with VGG-19 has been struggling to achieve good scores in terms of precision, sensitivity. Across all the checkpoints, the lambda values 0.01, 0.05, and 0.1 consistently have an average classification rate of around 43 to 47 percent, followed by poor average classification rates in the higher lambda values. The Figure 7, as shown below, discusses the Number of Images Classified Correctly (Top-5) on the 780 images in the unlabelled target across all the checkpoints and lambda values. It is to be noted that, as seen in Table 6, the bar chart matches the classification rate values in terms of the number of images, and reinforces that AdamW-5 is the highest performing checkpoint in this setup, with the correct prediction of 467 images, followed by Adagrad-40 of 449 images.

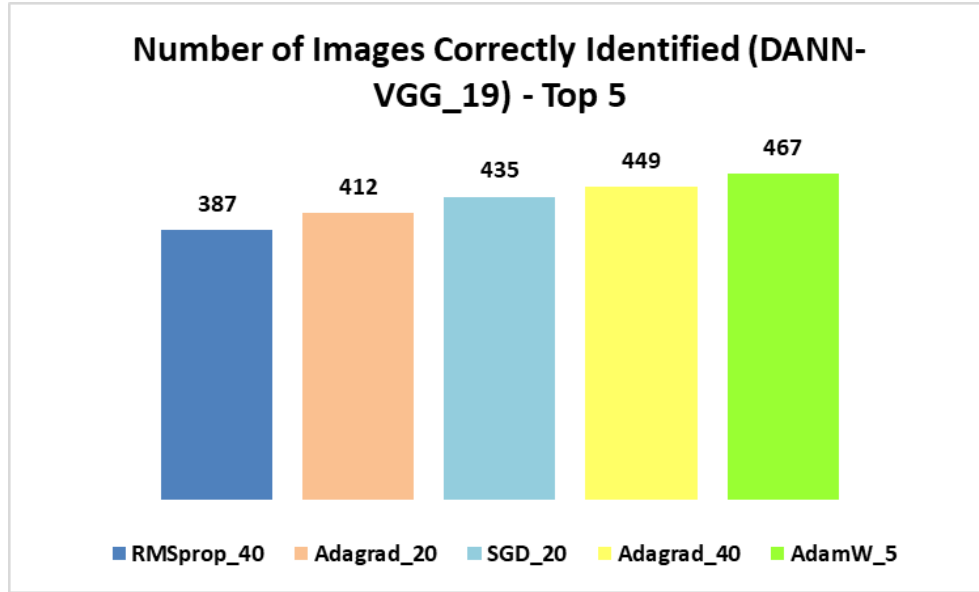


Figure 7: Top-5 Performances (Number of Images) Correctly Identified by DANN with VGG-19 Feature Extractor

4.3.3 Results of UDA with ADDA

The checkpoint-wise best performances of ADDA with the VGG-19 feature extractor have been shown below in Table 7. The AdamW-5 has had a peak classification rate of 30.64 percent, along with worrisome sensitivity and specificity. The AdamW-5 across different learning rates had an average classification rate of 19.17 percent. The Adagrad-40 has been marginally down from AdamW-5 in terms of all the metrics, highlighted with a maximum classification rate of 27.31. The Adagrad-40 has had an average classification rate across different learning rates has had a very slight improvement of 19.64 percent from AdamW-5's 19.17 percent.

Ckpt	LR	Cl.Rate	Prec	Sens	Spec	AUC
AdamW_5	1e-4	30.64%	0.200	0.306	0.861	0.630
Adagrad_40	5e-4	27.31%	0.267	0.273	0.855	0.594
Adagrad_20	5e-4	24.36%	0.252	0.244	0.849	0.645
RMSprop_40	2e-4	24.10%	0.355	0.241	0.848	0.573
SGD_20	1e-4	22.95%	0.238	0.229	0.846	0.515

Table 7: Checkpoint Wise Best Performances - ADDA with VGG-19 Feature Extractor (Stratified by Classification Rate)

The Adagrad-20's highest classification rate was 24.36 percent, but

it surprised us with an AUC of 0.645, surpassing all the other top contenders. The average classification rate across learning rates was 19.53 percent, which is a bit more than AdamW-5 and slightly lower than Adagrad-40. The RMSProp-40, despite having a peak of 24.10 percent classification rate, has shown an improvement in terms of precision with 0.355, which is the highest among all the peak-performing checkpoints, and comes with an average classification rate through learning rates of 15.41, which is lower than all the previous checkpoints. The SGD-20 comes up with a peak classification rate of 22.95 percent and shows dull performance in almost all aspects of the performance, and an average classification rate among the learning rates of 17.02 percent is noted.

It was noted that the learning rate $1e-4$ has had a better classification rate among all the others, with an average of 21.25 percent across the checkpoints, followed by $5e-4$ with an average classification rate of 18.20 percent across all the checkpoints. The Figure 8 shown below, confirms the top-most performance of AdamW-5, by correct prediction of 239 images out of 780 images in the target domain, followed by Adagrad-40 with a correct prediction of 213 images.

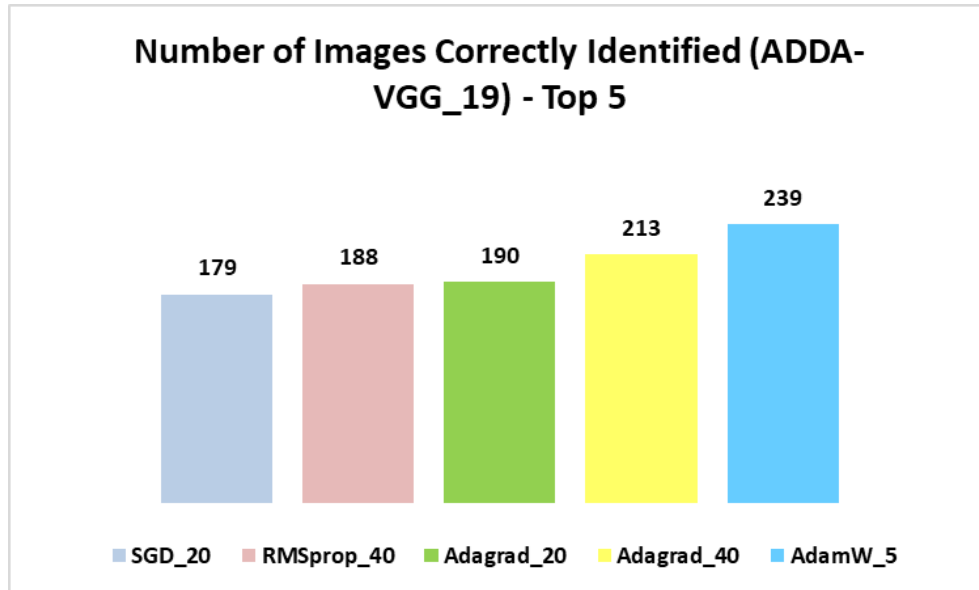


Figure 8: Top-5 Performances (Number of Images) Correctly Identified by ADDA with VGG-19 Feature Extractor

4.4 Results of Source Pretraining and UDA with EfficientNet-B0

4.4.1 Results of EfficientNet Source Pretraining

Opt-Eph	Tr_ac	Te_ac	Prec	Sens	Spec	AUC
RMSprop-40	100.00%	100.00%	1	1	1	1
AdamW-40	100.00%	100.00%	1	1	1	1
SGD-80	98.80%	99.15%	0.992	0.993	0.998	1
Adam-20	100.00%	99.15%	0.991	0.989	0.998	1
Adagrad-20	100.00%	99.15%	0.991	0.989	0.998	1

Table 8: 5 Checkpoints Obtained from Source Pretraining on EfficientNet based on Performance (Stratified by Testing Accuracy)

The EfficientNet-B0 has been better in terms of performance than VGG-19, as observed from Table 8. The model has achieved a 100 percent test accuracy in almost 8 instances with all the optimizers except SGD. Only a few points, which were observed to be precise in terms of fitting, were chosen. The precision, sensitivity, and specificity were mostly above 0.9, and the AUC never went down below 0.86. Each of the weights took a space of 27.6 MB in storage, highlighting the improved performance with fewer training parameters and a feasible computation cost of EfficientNet-B0 than VGG-19.

4.4.2 Results of UDA with DANN

Ckpt	Lambda	Cl.Rate	Prec	Sens	Spec	AUC
Adam_20	0.1	46.54%	0.374	0.465	0.893	0.719
SGD_80	0.5	41.28%	0.340	0.413	0.883	0.700
AdamW_40	0.5	40.90%	0.388	0.409	0.882	0.782
Adagrad_20	0.1	38.33%	0.348	0.383	0.877	0.677
RMSprop_40	0.01	33.72%	0.236	0.337	0.867	0.649

Table 9: Checkpoint Wise Best Performances - DANN with EfficientNet-B0 Feature Extractor (Stratified by Classification Rate)

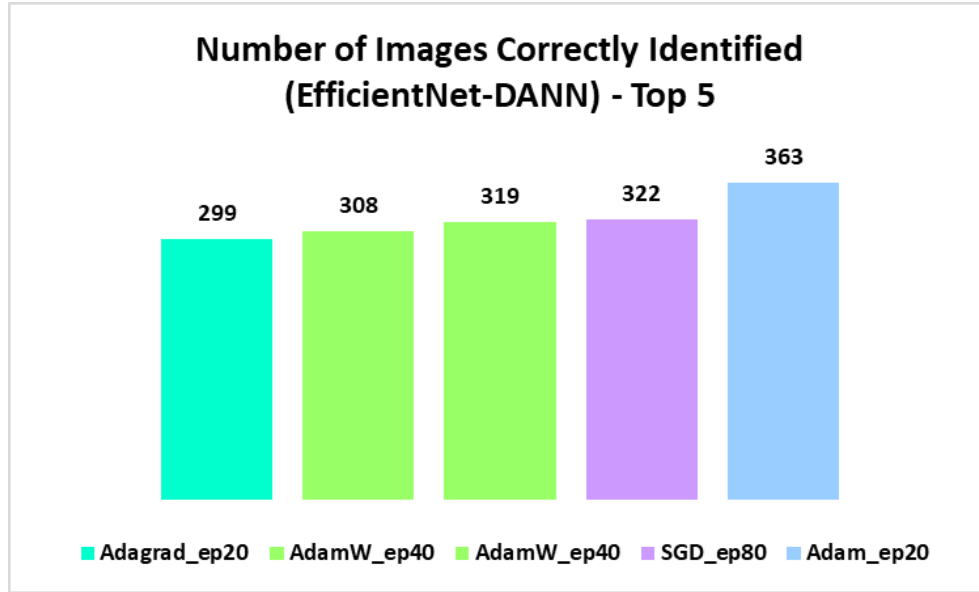


Figure 9: Top-5 Performances (Number of Images) Correctly Identified by DANN with EfficientNet-B0 Feature Extractor

The checkpoint-wise best results of DANN with EfficientNet-B0, as shown in Table 9, had a peak performance in the Adam-20 checkpoint with a 46.54 percent classification rate with a specificity of 0.893, followed by substandard sensitivity and precision of 0.465 and 0.374, respectively. Adam-20 has had an average classification rate of 37.48 percent across the lambdas. The SGD-20 has had a slightly lower performance than Adam, 20, with a peak classification rate of 41.28 percent, and a similar trend has repeated in other metrics. It has had an average classification rate of 36 percent across the lambdas. AdamW-40, despite having a peak classification rate of 40.90 percent, has outshone others in terms of precision and AUC with 0.388 and 0.782, respectively. AdamW-40 had an average classification rate of 34.84 percent across lambdas. Adagrad-20 and RMSProp-40 have had peak classification rates of 38.33 and 33.72 percent, along with a reduction of other metrics, and average classification rates of 32.53 and 28.05 percent throughout lambdas.

The lambda values from 0.01 to 0.05 have had average classification rates across all the checkpoints in the range of 33 to 37 percent, and face a fall to 29 percent for a lambda of 1. The number of images correctly identified, as seen in Figure 9, illustrates the highest performance of Adam-20 in successfully detecting 363 images, followed by SGD-80 with a detection of 322 images, and two predictions of AdamW-40 with 319 and 308 images in the target of 780 images, respectively.

4.4.3 Results of UDA with ADDA

Ckpt	LR	Cl.Rate	Prec	Sens	Spec	AUC
AdamW_40	2e-4	36.15%	0.388	0.362	0.872	0.617
Adagrad_20	1e-3	30.64%	0.298	0.306	0.861	0.578
Adam_20	2e-4	22.31%	0.201	0.223	0.845	0.530
RMSprop_40	1e-3	22.05%	0.192	0.221	0.844	0.526
SGD_80	1e-3	21.67%	0.270	0.217	0.843	0.518

Table 10: Checkpoint Wise Best Performances - ADDA with EfficientNet-B0 Feature Extractor (Stratified by Classification Rate)

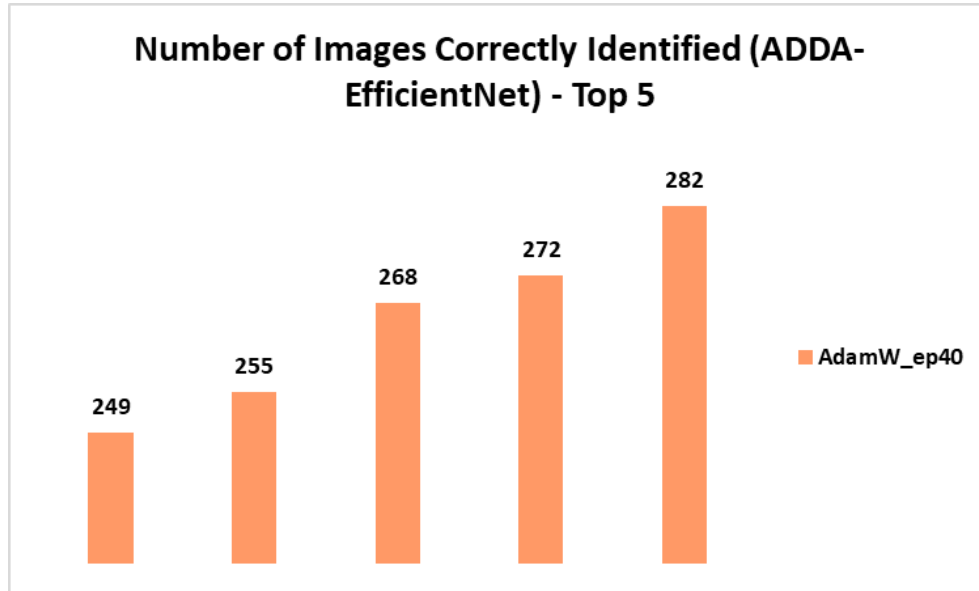


Figure 10: Top-5 Performances (Number of Images) Correctly Identified by ADDA with EfficientNet-B0 Feature Extractor

The best performing checkpoints in UDA on ADDA with EfficientNet-B0 are shown above in Table 10. The AdamW-40 emerges with a 36.15 percent peak classification rate with underperformances in sensitivity and specificity. The average classification rate of AdamW-40 among all the learning rates was 34 percent. The Adagrad-20 has a peak of 30.64 percent classification rate, with a significant reduction in all the metrics than AdamW-40. It was noted to have an average classification rate of 29.94 percent across learning rates. A significant deterioration of peak classification rates, as found in Adam-20 with 22.31 percent, RMSProp-40 with 22.05 percent, and SGD-80 with 21.67 percent, has also suffered below-

par precisions, specificities, sensitivities, and AUC. Adam-20, RMSProp-40, and SGD-80 had 20.92, 21.05, and 20.38 percent of average classification rates across learning rate variants. As shown above in Figure 10, AdamW-40 alone was correctly identifying the images in a range of 249 to 282 out of 780 on the target domain in all the top-5 instances.

4.5 Overall Trends

GAN Type	Lam/LR	Cl.Rate	Prec	Sens	Spec	AUC
VGG-19_DANN	Lam: 0.05	59.87%	0.538	0.599	0.920	0.820
EfN_DANN	Lam: 0.1	46.54%	0.374	0.465	0.893	0.719
EfN_ADDA	LR: 2e-4	36.15%	0.388	0.362	0.872	0.617
VGG-19_ADDA	LR: 1e-4	30.64%	0.200	0.306	0.861	0.630

Table 11: Best Performances of UDA from Every Experiment Conducted (Stratified by Classification Rate)

The overall trend, as seen in Table 11, was in favour of DANN with VGG-19 feature extractor, having the highest among top-most performances with a classification rate of 59.87 percent in the prediction of 467 images on the target domain with 780 images. This is furthermore supported by precision, sensitivity of 0.538 and 0.599, along with specificity of 0.920 and an AUC of 0.820. The DANN with VGG-19 has had a classification rate of 39.09 percent across all the checkpoints and lambda values tested. Once again, the DANN model with the EfficientNet-B0 feature extractor has shown the highest classification rate of 46.54 percent by predicting 363 images correctly on a target dataset of 780 images, illustrated by precision, sensitivity, specificity, and AUC of 0.374, 0.465, 0.893, and 0.719, respectively. This pipeline has had an average classification rate of 33.79 percent across all the tested checkpoints and lambda variations. The ADDA with the EfficientNet-B0 feature extractor has shown a peak classification rate of 36.15 percent with a prediction of 282 images in the target, supported by precision, sensitivity, specificity, and AUC of 0.388, 0.362, 0.872, and 0.617. ADDA with EfficientNet-B0 has shown an average classification rate of 25.26 percent across all the tested checkpoints and learning rates. ADDA with the VGG-19 feature extractor has had a substandard peak classification rate of 30.64 percent by predicting 239 target dataset images, illustrated with precision, sensitivity, specificity, and AUC of 0.2, 0.306, 0.861, and 0.630. This ADDA with VGG-19 combination has an average classification rate of 18.16 percent across all the tested checkpoints and learning rates. It is to be observed from the results that, despite having high specificities, almost all the

models have shown underperformance in precision, and sensitivities influenced by True Positives (TP), leading to a medium to high negative transfer supported by subtle anatomical discrepancies like thick images of Breast MRI class in the target than the source. This shows that the models (with no majority class, as all the classes are balanced) in our setting have established strong results in terms of identifying True Negatives, seen by stronger specificities and above-average AUC scores for the task of unsupervised domain adaptation.

5 Conclusion

The Unsupervised Domain Adaptation for Medical Images research has highlighted the efficiency of the Generative Adversarial Network variant Domain Adversarial Neural Network with VGG-19 and EfficientNet-B0's feature extractors of having a peak classification rate of 59.87 and 46.54 percent. Adversarial Discriminant Domain Adaptation variant of GAN has shown a peak classification rate of 36.15 percent with the usage of EfficientNet-B0, followed by VGG-19, with 30.64 percent. It was observed that VGG-19's memory-based advantage has worked well on the DANN pipeline with parallel task processing ability, while it struggled on the sequential task processing ADDA pipeline. EfficientNet's low training parameter structure has also produced better outcomes with the DANN architecture than the ADDA architecture. The negative transfer throughout the experiments was seen due to the lower identification of true positives seen with lower sensitivities and precisions, with a range of 0.5 to 0.6. And low resolutions of a few classes, coupled with minor anatomical misalignments, also have contributed to this negative transfer effect. This work can be scaled further by additional GANs with stronger domain adaptation capabilities, and the feature extractors of CNNs with stronger computation abilities can be replaced for analyzing the performances in different scenarios.

Acknowledgements

The experiments and research studies in this report were carried out with the support and guidance of the project supervisor, Ms. Xinyu Zhang, Research Fellow at The University of Adelaide.

Notes: The codes and Google Drive link for this study can be found here at a GitHub repository — **UDA for Medical Images**.

References

- [1] Steffen Albert, Barbara D Wichtmann, Wenzhao Zhao, Angelika Maurer, Juergen Hesser, Ulrike I Attenberger, Lothar R Schad, and Frank G Zoellner. Comparison of image normalization methods for multi-site deep learning. *Applied Sciences*, 13(15):8923, 2023.
- [2] Mohd Ali, Mehboob Ali, Mubashir Hussain, and Deepika Koundal. Generative adversarial networks (gans) for medical image processing: recent advancements. *Archives of Computational Methods in Engineering*, 32(2):1185–1198, 2025.
- [3] Monika Bansal, Munish Kumar, Monika Sachdeva, and Ajay Mittal. Transfer learning for image classification using vgg19: Caltech-101 image data set. *Journal of ambient intelligence and humanized computing*, 14(4):3609–3620, 2023.
- [4] Jiawei Chen, Ziqi Zhang, Xinpeng Xie, Yuexiang Li, Tao Xu, Kai Ma, and Yefeng Zheng. Beyond mutual information: Generative adversarial network for domain adaptation using information bottleneck constraint. *IEEE Transactions on Medical Imaging*, 41(3):595–607, 2021.
- [5] Wenshuang Chen, Qi Ye, Lihua Guo, and Qi Wu. Unsupervised cross-modality domain adaptation via source-domain labels guided contrastive learning for medical image segmentation. *Medical & Biological Engineering & Computing*, 63(7):2145–2159, 2025.
- [6] David Gramling, Ph.D. Optimizers (sgd, adam, rmsprop). Saw-DataScience.com, March 2025. Online; published March 2, 2025; Accessed: 17-08-2025.
- [7] Arifa Akter Eva, Jamin Rahman Jim, Ashifur Rahman, Hanif Bhuiyan, and Md Mohsin Kabir. Domain adaptation in medical imaging: Evaluating the effectiveness of transfer learning. In *Data-Driven Clinical Decision-Making Using Deep Learning in Imaging*, pages 1–23. Springer, 2024.
- [8] Azin Shokraei Fard, David C Reutens, and Viktor Vegh. Cnns and gans in mri-based cross-modality medical image estimation. *arXiv preprint arXiv:2106.02198*, 2021.
- [9] D Ganesh, Upendra K Verma, Smita Patil, and Intekhab Alam. Domain adaptation using generative adversarial networks for medical image synthesis. In *International Conference on Data Science, Machine Learning and Applications*, pages 28–33. Springer, 2023.

-
- [10] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario March, and Victor Lempitsky. Domain-adversarial training of neural networks. *Journal of machine learning research*, 17(59):1–35, 2016.
- [11] Jun Gao, Qicheng Lao, Qingbo Kang, Paul Liu, Le Zhang, and Kang Li. Unsupervised cross-disease domain adaptation by lesion scale matching. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 660–670. Springer, 2022.
- [12] GeeksforGeeks. EfficientNet Architecture. <https://www.geeksforgeeks.org/computer-vision/efficientnet-architecture/>, July 23 2025. [Online; Accessed: 17-08-2025].
- [13] Google Cloud. Introducing g2 vms with NVIDIA L4 gpus. <https://cloud.google.com/blog/products/compute/introducing-g2-vms-with-nvidia-l4-gpus>, 2025. Accessed: 17-08-2025.
- [14] Brian Guo, Darui Lu, Gregory Szumel, Rongze Gui, Tingyu Wang, Nicholas Konz, and Maciej A Mazurowski. The impact of scanner domain shift on deep learning performance in medical imaging: an experimental study. *arXiv preprint arXiv:2409.04368*, 2024.
- [15] Mahta HassanPour Zonoozi and Vahid Seydi. A survey on adversarial domain adaptation. *Neural Processing Letters*, 55(3):2429–2469, 2023.
- [16] Paolo Iacono and Naimul Khan. Structure preserving cycle-gan for unsupervised medical image domain adaptation. In *2024 46th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 1–5. IEEE, 2024.
- [17] Shiwangi Kulhari, Amit Verma, and Abhimanyu Singh Kulhari. Utilizing vgg-19 for enhanced face recognition: A comprehensive analysis. *Research Journal of Recent Sciences* -----ISSN, 2277:2502, 2025.
- [18] Suruchi Kumari and Pravendra Singh. Deep learning for unsupervised domain adaptation in medical imaging: Recent advancements and future perspectives. *Computers in Biology and Medicine*, 170:107912, 2024.

- [19] Yujie Liu and Qicheng Zhang. Multi-source unsupervised domain adaptation for medical image recognition. In *International Conference on Intelligent Computing*, pages 428–440. Springer, 2024.
- [20] Amirreza Mahbod, Nematollah Saeidi, Sepideh Hatamikia, and Ramona Woitek. Evaluating pre-trained convolutional neural networks and foundation models as feature extractors for content-based medical image retrieval. *Engineering Applications of Artificial Intelligence*, 150:110571, 2025.
- [21] Kazuki Omi and Toru Tamaki. On the instability of unsupervised domain adaptation with adda. In *International Workshop on Advanced Imaging Technology (IWAIT) 2022*, volume 12177, pages 380–383. SPIE, 2022.
- [22] Duc Duy Pham, SM Koesnadi, Gurbandurdy Dovletov, and Josef Pauli. Unsupervised adversarial domain adaptation for multi-label classification of chest x-ray. In *2021 IEEE 18th International Symposium on Biomedical Imaging (ISBI)*, pages 1236–1240. IEEE, 2021.
- [23] Rehan Raza, Fatima Zulfiqar, Muhammad Owais Khan, Muhammad Arif, Atif Alvi, Muhammad Aksam Iftikhar, and Tanvir Alam. Lung-effnet: Lung cancer classification using efficientnet from ct-scan images. *Engineering Applications of Artificial Intelligence*, 126:106902, 2023.
- [24] Anthony Sicilia, Xingchen Zhao, and Seong Jae Hwang. Domain adversarial neural networks for domain generalization: When it works and how to improve. *Machine Learning*, 112(7):2685–2721, 2023.
- [25] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.
- [26] Youssef Skandarani, Pierre-Marc Jodoin, and Alain Lalonde. Gans for medical image synthesis: An empirical study. *Journal of Imaging*, 9(3):69, 2023.
- [27] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019.
- [28] Adapt Development Team. adapt.feature.based.ADDA: Adversarial discriminative domain adaptation documentation. https://adapt-python.github.io/adapt/generated/adapt.feature_based.ADDA.html, 2023.

- [29] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7167–7176, 2017.
- [30] Aly A Valliani, Faris F Gulamali, Young Joon Kwon, Michael L Martini, Chiatse Wang, Douglas Kondziolka, Viola J Chen, Weichung Wang, Anthony B Costa, and Eric K Oermann. Deploying deep learning models on unseen medical imaging using adversarial domain adaptation. *PloS one*, 17(10):e0273262, 2022.
- [31] Jichen Yang, Lei Wang, and Heng Lian. Debiased transfer learning estimation and inference for multinomial regression. *Statistics and Computing*, 35(3):73, 2025.
- [32] Yiju Yang, Tianxiao Zhang, Guanyu Li, Taejoon Kim, and Guanghui Wang. An unsupervised domain adaptation model based on dual-module adversarial training. *Neurocomputing*, 475:102–111, 2022.
- [33] Mei Yu, Zhiyuan Xu, Jie Gao, Jian Yu, and Mankun Zhao. An unsupervised domain adaptive network based on category prototype alignment for medical image segmentation. In *International Conference on Intelligent Computing*, pages 168–179. Springer, 2023.
- [34] Ye Yu, Weixiao Chen, Fengxin Chen, Wei Jia, and Qiang Lu. Night-time vehicle model recognition based on domain adaptation. *Multi-media tools and applications*, 83(4):9577–9596, 2024.
- [35] Yifan Zhang, Ying Wei, Qingyao Wu, Peilin Zhao, Shuaicheng Niu, Junzhou Huang, and Mingkui Tan. Collaborative unsupervised domain adaptation for medical image diagnosis. *IEEE Transactions on Image Processing*, 29:7834–7844, 2020.
- [36] Boyun Zheng, Ranran Zhang, Songhui Diao, Jingke Zhu, Yixuan Yuan, Jing Cai, Liang Shao, Shuo Li, and Wenjian Qin. Dual domain distribution disruption with semantics preservation: Unsupervised domain adaptation for medical image segmentation. *Medical Image Analysis*, 97:103275, 2024.
- [37] Zhen Zheng, Rui Li, and Cheng Liu. Learning robust features alignment for cross-domain medical image analysis. *Complex & Intelligent Systems*, 10(2):2717–2731, 2024.