Unsupervised Domain Adaptation with VefNet Convolutional Neural Network Feature Extractor for Medical Images

Sai Akshay Suresh
a1906525

January 7, 2026

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

THE UNIVERSITY *of* ADELAIDE

Project Area: **Image Processing, Transfer Learning, Domain Adaptation**
Project Supervisor: **Dr. Xinyu Zhang**

Abstract

The importance of the medical image classifications of scans are growing rapidly to check, diagnose any condition, and categorise any disease for timely treatment and improvement of patient outcomes. This report deals with the detection of 3000 unlabelled medical scan images of various body parts with six classes (AbdomenCT, BreastMRI, ChestCT, ChestXRay, HandXRay, and HeadCT), collected from multiple Kaggle pages, through learning a labelled dataset with the same six classes of 3000 images, obtained from different institutes from the detection dataset, i.e in an unsupervised setup. This study leverages the use of two Domain Adaptation (DA) algorithms namely—Domain Adversarial Neural Network (DANN) and Conditional Adversarial Domain Adaptation with Entropy Conditioning (CDAN+E), that use various Convolutional Neural Network (CNN) architectures for learning both the labelled and unlabelled dataset for identifying the common features in images through the pre-learned source dataset weights (.pth files) via CNN architectures. This research proposes a CNN model, named VefNet, which utilises the deeper feature extraction stems of the VGG architecture for extensive learning of anatomical structures and has a lower number of training parameters, inspired by the EfficientNet series of CNN architectures. Although the higher training parameter-based architectures like VGG-19 (~140 M parameters), and EfficientNet-V2-S (~38.17 M parameters) achieved test accuracies above 99.50% in both the DA algorithms, the memory-friendly VefNet CNN architecture (~16.78 M parameters) as a feature extractor, proved with test accuracies of 90.37% (in DANN) and 99.40% (in CDAN+E).

1 Introduction

The Machine Learning (ML) techniques, especially Transfer Learning (TL), which involves leveraging the learned weights of a model from one task to another, Domain Adaptation (DA), a sub-branch of TL, goes a step further to learn and save learned weights of a labelled dataset, called the source domain dataset, and predict on a fully labelled or partially labelled dataset, called the target domain dataset that is from a related domain like, learning from numbers in white and identifying numbers in black text [14]. As the images were collected from various machines and institutions, DA is seen as the immediate technique to address the problems that arise with the domain shift that is present in the source to target, as DA has the mechanism to identify the common features of the source and target datasets [17]. This research delves into the usage of

various Convolutional Neural Networks (CNNs) architectures as feature extractors, that will serve as the eyes and brains for classifications on the target dataset, by tuning five variants of two hyperparameters (optimisers and epochs) for storing learned weights by pretraining the source dataset and using in two adversarial DA algorithms—Domain Adversarial Neural Network (DANN), and Conditional Adversarial Domain Adaptation with Entropy Conditioning (CDAN+E), with five variations in the lambda values from 0.01 to 1.0 that adjusts the losses in DANN and CDAN+E for training CNNs to learn domain-invariant features of both the source and target domain datasets[27].

In this research, a CNN feature extractor named VefNet is proposed and deployed, which is meticulously constructed with the two earlier layers of VGG CNN architecture by having stacks of three 2D-convolutions in the first block, and two 2D-convolutions in the second block for extensive feature extractions [28], and three stages of blocks inspired by EfficientNet CNNs, in which, stage A with three FusedMBConvolution blocks for faster training along a small overhead [30], stage B & C with four MBConv blocks along the Squeeze and Excite (SE) optimizations in each stage for achieving a higher accuracy with lower number of training parameters [10, 20]. Apart from this proposal, the research also inspects how transforming the images with techniques like histogram equalisation for lighting improvement and retrieval of the features of the poor quality images in the source dataset helps in improved detection.

This research is motivated by the challenge of vulnerability of closer anatomical structures of medical images, where models that learn images with such similar structures misclassify images of the wrong classes, resulting in loss [11], and to overcome memory-based issues like Out-of-Memory (OOM) that stem from the usage of high-parameter CNN models within limited hardware conditions like GPUs with lower memory and task handling capacities [12]. The study introduces the VefNet CNN architecture, focusing on memory efficiency and reduced misclassification rates. Among high-parameter models, VGG-19 reached 100% accuracy (in CDAN+E) and 99.97% (in DANN), while EfficientNet-V2-S scored 99.53% (in CDAN+E) and 90.37% (in DANN). Lower-parameter models like MobileNet-V2 attained 99.10% (in CDAN+E) and 82.47% (in DANN). VefNet, using 16.78M parameters, achieved a maximum accuracy of 99.40% (in CDAN+E) and 90.37% (in DANN), showing a strong performance among its contenders.

2 Background

2.1 Related Works

A recent work [31], trained on images of normal and cardiomegaly chest conditions from one centre (Pad-Chest), identifies the same conditions from another centre (ULM University Medical Centre) without labels. This study examined the use of various CNN-based backbones, i.e., feature extractors such as DenseNet and ResNet variants, in Deep Reconstruction-based DA algorithms to learn the source dataset, and reconstruct and predict the target dataset based on domain-invariant features. This Unsupervised Domain Adaptation (UDA) setup had an improvement over the competitors of 6.96% G-Mean metric for the measurement of performance between the majority and minority classes. Another study [8], which performed UDA with the labelled source (Chest X-Ray scans of multiple diseases) and unlabelled target (ChestXRay-14), with the usage of CNN-backbones such as ResNet-50, and Wasserstein distance with class-wise discrepancy loss for reduction of domain distances and improvement of tighter class-alignment. This explicit class-aware alignment setup provided an increase of 14.80% of AUC when compared to no DA.

A relevant work [35], discussing adaptation to a new Chest-XRay (CXR) domain when the source data becomes inaccessible with a scarcity of target labels. This Semi-supervised Multi-source-free Domain Adaptation (SMDA) contained multiple teacher CNNs that guided one student CNN to learn domain-invariant features by giving adversarial feedback in case of mistakes for learning the six classes of CXR conditions. This source-free setup was useful to address open-set real-world problems and achieved a 92.84% AUCROC over the existing models. A study that showed the prowess of hybrid models [24], especially a fusion of VGG-16's 3x3 convolution stacks and EfficientNet-B0's lower training parameter-based efficient features named MDEV. The source dataset (ImageNet) had 20,000+ classes, while the target dataset had two classes of CXR (Pneumonia and Normal). This ensemble method of VGG and EfficientNet was stacked with a meta-learner to do identification based on instances, and for stronger feature learning. This model has shown an improved accuracy of 92.15%, which was higher than VGG's 87.82% and EfficientNet's 90.38%. Furthermore, another interesting work [7], which features a backbone that is built on the DeepLab-ResNet-50 for unsupervised cross-modality DA for medical image segmentation (MRI $\leftrightarrow$ CT). This study involved using the Brain Tumour and Heart segmentation datasets collected from different institutions as labelled source and

unlabelled target. This proposed work performs image translation and feature alignment between the images for generating target-style images from source and vice versa, measured with adversarial losses and reconstruction losses, and produces an improved similarity score between the predicted and ground truth labels of 78.50% compared to the previous approaches.

2.2 Challenges in the Existing Methods

A paper that discussed the effects of having anatomically similar classes based on unsupervised cross-modality DA [25], with contrast-enhanced T1 MRI in source and High-resolution T2 MRI in target for the segmentation of vestibular schwannoma (tumor from inner ear to brain) and cochlea (another part in inner ear for hearing) confirmed that, due to the position and resemblance of structures of the two classes used, the backbone CNNs used were confused and lead to performance degradation. Another related study that discussed the misclassification by the DA module [26] was obtained from experiments of UDA for small-bowel segmentation (contrast CT in source and non-contrast CT in target). As the small bowel loops have a very similar texture to nearby intestines, after the DA, boundaries between the loops remain structurally alike, and this stimulates the detection of false positives, resulting in degradation of performance indicators.

To illustrate the ill-effects of high-parameter CNNs, a study used multiple Unsupervised DA benchmarks (e.g., VisDA-2017) [33], which were explicitly implemented on various CNN architectures like VGG-19 for classifying across domains of various datasets and observed that the computational runs were expensive and required advanced hardware equipment. A potential solution is to reduce the mini-batch size, but this would likely degrade the detection accuracy [21]. Another study [32] highlighted that when a class in the source and target are not similar, it results in negative transfer, i.e., incorrect mappings by the DA algorithms within an adversarial learning framework, leading to misclassification and a reduction in performance. It is expected that the images in the source and targets show some kind of similarity for right predictions. Additionally, a research showed that the unlabelled target data that contained the images of classes in the CheXpert dataset [3], taken in a smartphone that had poor lighting, blur quality, etc had suffered poor identification by AI models with the original ChestXpert source-based pretraining by various CNNs. This study indicates that the images with low illumination and photographic artefacts can potentially cause misalignment

from clean digital images, contrast/edges fade, and intricate features go missing, which is necessary for proper learning or classification.

2.3 Work Since Last Submission

In the previous trimester's submission, DA with two algorithms, namely, DANN, and a sequential DA algorithm, Adversarial Discriminant Domain Adaptation (ADDA), were used on the baselines—VGG-19, and EfficientNet-B0 as CNN feature extractors. The labelled MedicalMNIST dataset obtained from Kaggle, which had the six classes of medical scan images (AbdomenCT, BreastMRI, ChestCT, ChestXR, HandXR, HeadCT) with 130 images per class and the same six classes of unlabelled target data with 130 images per class from different Kaggle pages to the source, was used for this study. For the validation of the target data by the algorithms with CNN feature extractors, a CSV file with image and class names of the images was created. It was observed that the VGG-19 produced a peak test accuracy of 59.87% in DANN, and a 30.64% in ADDA, and EfficientNet-B0 produced 46.54% in DANN, and 36.15% in ADDA. Due to ADDA's poor performance, it was dropped. The current setting of DA in this trimester, deals with improvements to poor-quality images in the source dataset, development of the VefNet CNN architecture, study of the DA algorithms—DANN, and the advanced CDAN+E with an increase in the number of images to 500 per class in both the source and target datasets, and comparative investigation of baselines (VGG-19, EfficientNet-B0, EfficientNet-V2-S), competitors (ResNet-50, MobileNet-V2), and VefNet CNN models.

3 Methods

3.1 Workflow of Experiments

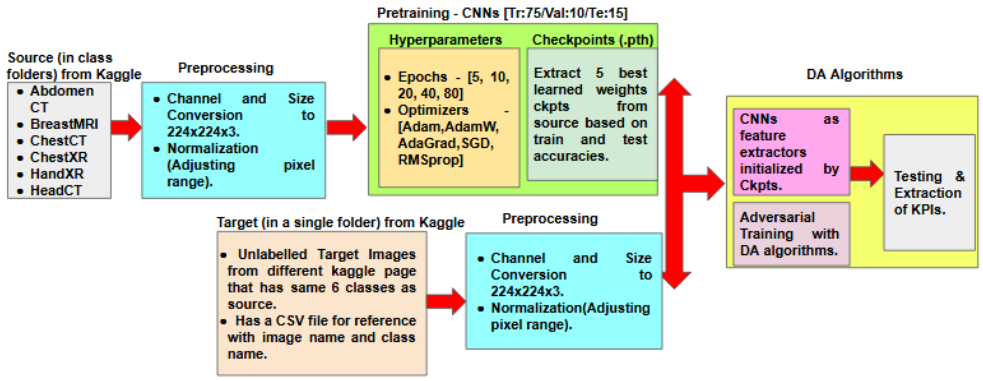


Figure 1: Project Workflow

The flow of experiments begins with preprocessing the Kaggle obtained labelled source dataset of six classes—AbdomenCT, BreastMRI, ChestCT, ChestXR, HandXR, Head (in six folders) and the unlabelled target dataset with the same six classes (in a single folder) obtained from Kaggle pages different from the source dataset, with the necessary steps such as size, channel conversions and normalisations, as shown in Figure 1. The source dataset goes through pretraining on the CNN architectures (built from scratch according to original architectures) with a 75% train, 10% validation, and 15% test split with the variations of optimisers (Adam, AdamW, Adagrad, RMSprop, and SGD), and epochs (5, 10, 20, 40, 80). For each of the optimisers, every value of the epoch was tested on the source dataset and saved as learned weights (.pth checkpoints).

From the results, based on train and test accuracies, the top-5 checkpoints are selected for every CNN architecture and are initialised again on the CNN feature extractors in the adversarial DA algorithms (DANN, CDAN+E) for performing DA on the unlabelled target dataset with a csv reference file that has image and class names for validation of DA algorithms and extraction of key performance indicators. The checkpoints are initialised as it gives a start to the feature extractors with the source image features required for detection on the target dataset. Each of the five chosen checkpoints is tested on the five values of hyperparameters of the DA algorithms.

3.2 The CNN architectures used in Source Pretraining and Feature Extraction in DA Algorithms

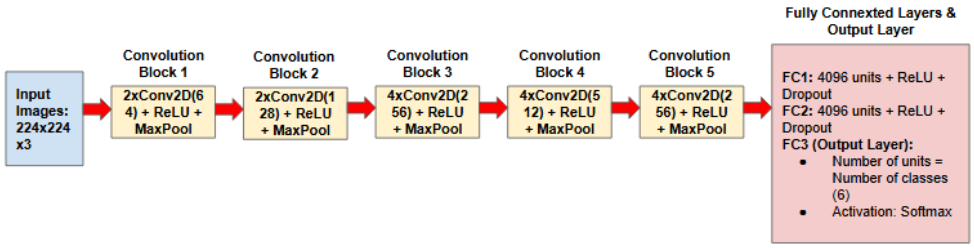


Figure 2: Architecture of VGG-19 CNN

The baseline architecture, VGG-19 (16 Convolutional Layers + 3 Fully Connected Layers), developed by Visual Geometry Group (VGG) in 2014 at the University of Oxford, was used to study how the network depth can affect/improve accuracy for large-scale image recognition [28]. The usage of multiple 3x3 2D convolutions, as shown in Figure 2, makes VGG-19 a suitable candidate for deeper feature extraction tasks. The ReLU activation at the end provides non-linearity, and this ensures that the network learns complex patterns of the images in the source dataset. VGG-19 comes with ~140 M trainable parameters.

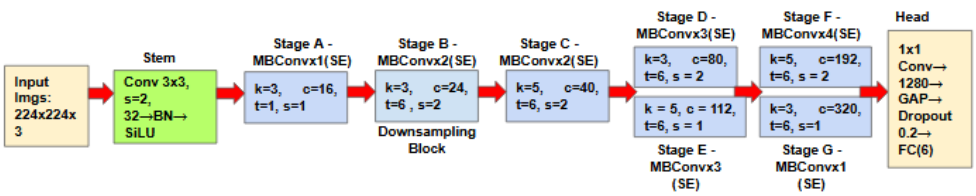


Figure 3: Architecture of EfficientNet-B0 CNN

The EfficientNet-B0, as shown in Figure 3, was developed by Google in 2019 as the first architecture in the EfficientNet family with compound model scaling that scales the model depth and width for better accuracy on the ImageNet dataset [29]. EfficientNet-B0's usage of MBConv blocks with Squeeze-and-Excitation(SE) for capturing features from informative channels while making the number of parameters lower is preferred for low-computational experiments. In the Figure 3, kernels were represented by "k", strides by "s", output channels by "c", and expansion size by "t". The Global Average Pooling (GAP) at the Fully Connected (FC) downsamples the feature maps into a single value. EfficientNet-B0 has ~7.16 M trainable parameters. This architecture was chosen to study its ability to perform with the components—MBConv and SE.

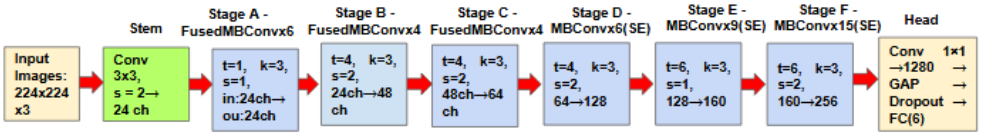


Figure 4: Architecture of EfficientNet-V2-S CNN

The EfficientNet-V2-S, another baseline used in the construction of VefNet, was developed in 2021 at Google, with the advantage of being a smaller model like the previous EfficientNets and a faster training with the usage of automated Training-Aware Neural Architecture Search (NAS), and scaling for image classification and transfer learning tasks [30]. The Fused-MBConv, along with standard convolutions without the use of SE in the earlier stages, as shown in Figure 4, aims to reduce training time and limit memory usage. While the later stages use MBConv with SE blocks, depthwise convolutions and channel attention for giving in-depth feature-level focus while learning the source dataset. EfficientNet-V2-S comes with ~38.17 M parameters.

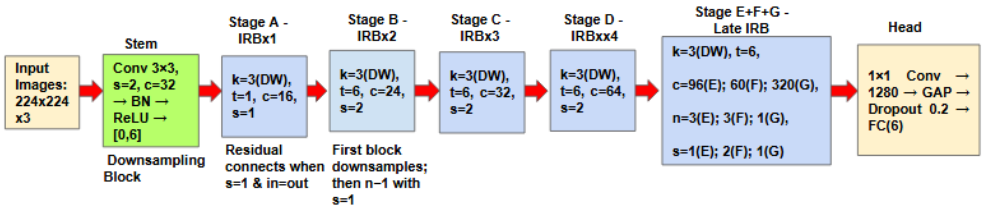


Figure 5: Architecture of MobileNet-V2 CNN

The MobileNet-V2, as shown in Figure 5, was developed in 2018 by Google to address the accuracy-efficiency trade-off in image detection compared to earlier models. The use of Inverted Residual Blocks (IRB) connects the bottleneck layers with a lower number of channels/feature maps, utilising a skip connection to make neural networks more efficient by reducing computational cost and complexity, which is particularly beneficial for mobile devices with restricted power and memory [20]. As shown in the first downsampling stem, Batch Normalisation (BN) does channel-wise normalisation and scaling for faster training, and the [0,6] corresponds to the outputs produced by using the Relu6 activation function for ensuring compatibility with portable devices. MobileNet-V2 had the lowest trainable parameters of ~2.23 M among all the executed models. The "n" in the last stage block represents repeats number of repeats of blocks.

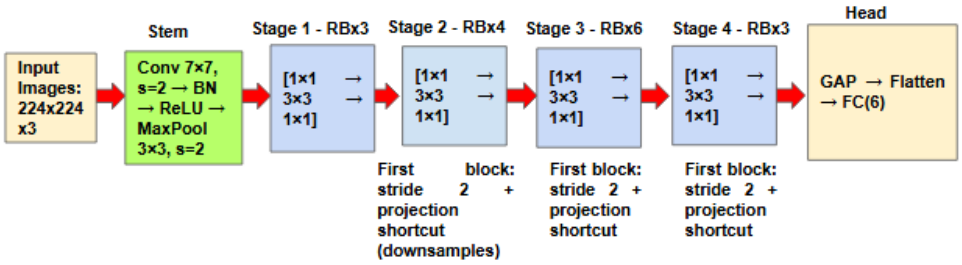


Figure 6: Architecture of ResNet-50 CNN

A contemporary architecture, ResNet-50, as shown in Figure 6, was published in 2015 by Microsoft to manage optimisation problems stemming from very deep CNNs [9]. The usage of Residual Blocks (RBs) to address the vanishing gradients problem, where gradients shrink that minimise the model error, resulting in reduced accuracy and overfitting. The convolution kernels inside every residual stage are represented in 1x1 and 3x3. ResNet-50 has ~ 23.53 M parameters.

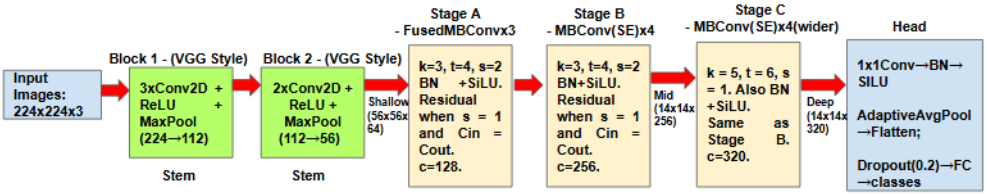


Figure 7: Architecture of VefNet CNN

The proposed VefNet architecture, as shown in Figure 7, was developed for the purpose of deeper learning of the source dataset's features with VGG's stems and a lower model complexity with a limited number of trainable parameters with EfficientNet's attributes. The first VGGish deeper 3x3 2D convolutional stem with max pooling (224 to 112) aims for learning the edges and textures of the classes of the dataset, and the second stem with maxpooling (112 to 56) aims to strike a balance between a lower number of parameters and Floating Point Operations (FLOPs) before the convolutions. The FusedMBConv blocks from EfficientNet-V2-S for faster training and lower memory in Stage-A, and MBConv blocks with SE for channel-wise information pull and ensuring the accuracy-efficiency trade-off in Stage-B, along with feature taps (shallow/mid/tap) before Stage-A, and before and after Stage-C, ensure that the CNN algorithm is fit for DA. These feature taps are feature maps extracted from shallow, mid, and deep layers of the VefNet architecture and exposed so that extra heads or losses are computed via

these taps, and gradient backpropagation adjusts all layers so that the final classification improves, and source-target features are more aligned at the tap levels. The shallow tap signifies a high resolution 56x56 for keeping nuanced local details and 64 channels for registering the low-level features like edges, the mid tap signifies a compressed resolution of 14x14 for a rough understanding of images and 256 channels capture mid-level details like tumor shape, etc, and the deep tap with a resolution 14x14 and the 320 channels for high-level and class-specific understanding like what differentiates the classes [23]. This model had 16.78 M trainable parameters.

3.3 The Algorithms for Unsupervised Domain Adaptation

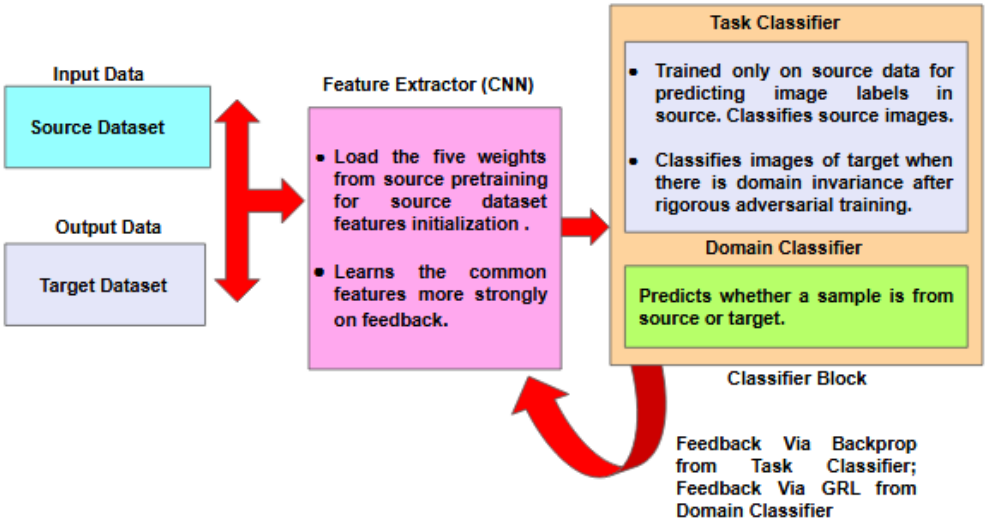


Figure 8: The flow of the DANN Algorithm

The Domain Adversarial Neural Network (DANN) algorithm, seen in Figure 8, uses the CNNs as feature extractors for learning the source and target datasets [6]. The feature extractor gets initialised with top-5 checkpoints, which were obtained through pretraining in the CNNs with optimisers and epochs, selected by top train and test accuracies for a start with source data features. With these checkpoints, the CNNs learn the datasets initially, and send the feature representations to the classifier block, which contains the Task Classifier (TC), trained on source data and classifies images with source data labels, and the Domain Classifier (DC), which predicts whether a sample is from the source or target set. Due to shared classes between the source and target, the TC tries

to classify the target images, produces loss in case of wrong predictions and sends feedback to the CNN through backpropagation. If the source-target features do not match, i.e source is classified as source and target is classified as target separately, the DC block uses the Gradient Reversal Layer (GRL), i.e., a router to send a feedback signal to the feature extractor, so that the CNN can thoroughly learn the features of both datasets and identify commonalities. This mechanism of sending feedback by the TC and DC to the feature extractor is called adversarial training, so that with every feedback, the CNN improves its learning and sends the features again to the classifier block. The TC correctly classifies the target data when the feature extractor has completely identified the domain-invariant features between the source and target datasets. The objective of the DANN is given as follows in Equation (1)-

$$\min_{F,C} \max_{D} \mathcal{L}_{\text{DANN}}(F, C, D) = \underbrace{\mathbb{E}_s[\ell_y(C(F(x_s)),\, y_s)]}_{\text{label loss on source}} - \lambda \underbrace{\mathbb{E}_{s,t}[\ell_d(D(F(x)),\, d(x))]}_{\text{domain loss on src+tar}} \tag{1}$$

In Equation (1), the objective of DANN is to minimise the loss in the feature extractor F, and the task classifier C (when it gives wrong labels to the target images), and maximise the loss in the domain classifier D (max has to be 0.5 probability score where the DC is not able to say which image is from source/target), so that it gets fooled that the source is target and vice versa, and the C classifies the images with domain invariance [6]. The first expectation term $\mathbb{E}_s[\cdot]$ is the average class loss $\ell_y(\cdot)$ from the task classifier over labelled source images (x_s, y_s). The second expectation $\mathbb{E}_{s,t}[\cdot]$ is the average domain loss $\ell_d(\cdot)$ from domain classifier over combined source–target images x with domain label $d(x) \in \{0, 1\}$. The hyperparameter $\lambda > 0$ controls how strongly we enforce domain alignment.

The Conditional Adversarial Domain Adversarial Network with Entropy Conditioning (CDAN+E) is an extension of DANN, with the addition of multilinear conditioning and entropy weighting for enhanced accuracy over DANN [15]. After the feature extractor, as seen in Figure 9, there is a bottleneck block to reduce the feature vector, for a lighter discrimination, and the bottleneck sends the resultant feature vector z to the classifier block. The TC, apart from classifying and sending signals, also produces class probabilities g now with the application of the softmax function. The multilinear conditioning works on aligning the feature

representations with their classes, instead of mixing both the source and target datasets. This conditioning is represented through $z \otimes g$. The feedback mechanism works in the same way as DANN; additionally, DC also gets updated and predicts again if there is maximum domain loss for further verification. Entropy weighting uses the model's confidence (entropy) to scale each sample's domain loss, so that the confidently predicted samples are more and the uncertain ones are less. Such conditional improvements over DANN ensure that there is an even more rigorous DA that happens for aligning the domain-invariant features of the datasets.

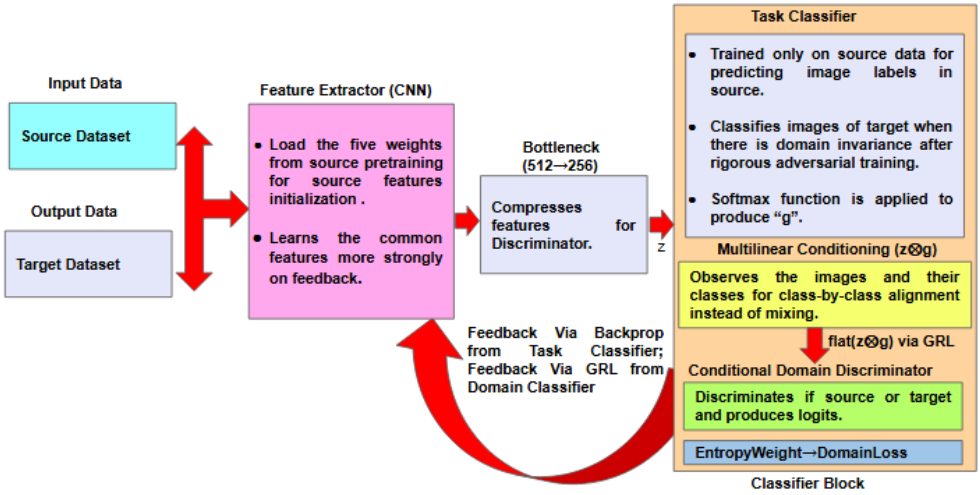


Figure 9: The flow of the CDAN+E Algorithm

$$\min_{F,C} \max_{D} \mathcal{L}_{\text{CDAN+E}}(F,C,D) = \underbrace{\mathbb{E}_s[\ell_y(C(F(x_s)),\, y_s)]}_{\text{label loss on source}} + \lambda \underbrace{\mathbb{E}_{s,t}[w(H(g(x)))\ \ell_d(D(T(f(x),g(x)))\,,\ d(x))]}_{\text{conditional + entropy-weighted domain loss}} \tag{2}$$

In Equation (2), CDAN+E keeps the same source classification term as DANN but modifies the domain loss. Here $g(x)$ is the class-probability vector after softmax's application, and $H(g(x)) = -\sum_c g_c(x) \log g_c(x)$ is its entropy, which shows how uncertain the prediction is (high entropy = very unsure). The weight $w(H(g(x))) = 1 + \exp(-H(g(x)))$ is therefore larger for low-entropy samples, which indicates high confidence and the weight is smaller for highly uncertain ones. The weight modulates how much each of the images contributes to domain loss. The $T(f,g)$ represents the multilinear conditioning unit for doing the outer product

of the feature and class-probability vectors.

3.4 Hyperparameters

Hyperparameter	Used for	Values/Variants
Optimisers	Pretraining	SGD, Adam, RMSprop, Adagrad, AdamW
Epochs	Pretraining	5, 10, 20, 40, 80
Lambda	DA	0.01, 0.05, 0.1, 0.5, 1

Table 1: Summary of Hyperparameters Used

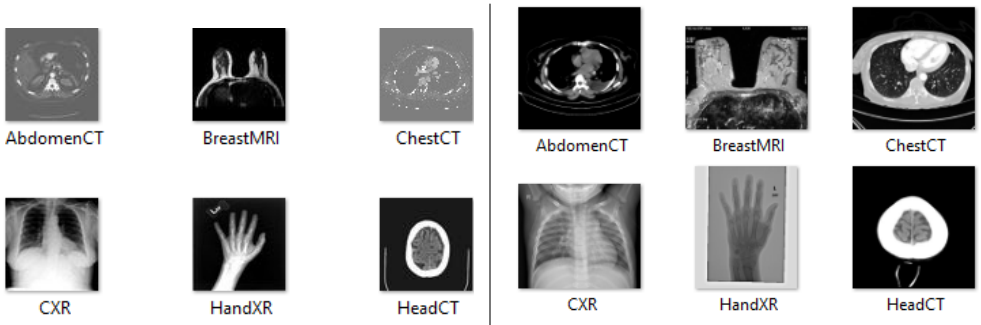
The hyperparameters, as shown in Table 1, were used for both source dataset pretraining (optimisers and epochs) with CNNs, and DA (lambda) with CNN feature extractors in DANN, and CDAN+E. The optimisers—The Stochastic Gradient Descent (SGD), published in 1951, has a feature of taking mini-batches of the data to update model's parameters and this reduces errors; The Adam, published in 2014, combines the mean and variance of gradients for adapting the learning rates, and provides a faster performance; The RMSProp, published in 2012, uses an exponentially decaying mean of the squared gradients in the past for normalizing the step size, thus preventing the learning rate getting reduced to a very lower value; Adagrad, published in 2011, assigns an unique adaptive learning rate for each parameter by fixing updates inversely proportional to the square root of the sum of all the past squared gradients of a specific parameter, and is helpful for learning rarely seen features in an image; Adam with Weight Decay (AdamW), published in 2017, seperates the weight decay from gradient-based updates, and it is directly applied to the model parameters, and this ensures a better generalization and reduced overfitting [2, 1]. An epoch, which is one complete pass through the whole dataset, makes sure that the whole dataset is seen multiple times for every epoch value fixed [13]. For every variant of the optimiser, each value of the epoch was tested for source data pretraining with the CNNs.

The lambda, which is the coefficient of DC loss and a trade-off parameter between TC loss and DC loss, is used for DA in both DANN and CDAN+E. As the objective of both DANN and CDAN+E is to minimise the TC loss and maximise the DC loss for achieving domain-invariance, lambda controls how much the model should sacrifice TC accuracy [36]. A high lambda focuses on achieving domain-invariance in DC, but can still produce wrong predictions, and a low lambda aims for reducing loss

in TC, but can miss feature invariance. Each of the top-5 pretraining checkpoints was validated with all the selected lambda values.

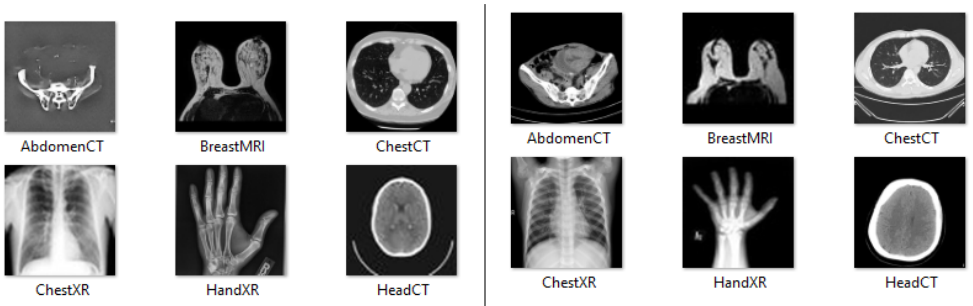
4 Results

4.1 Dataset Selection



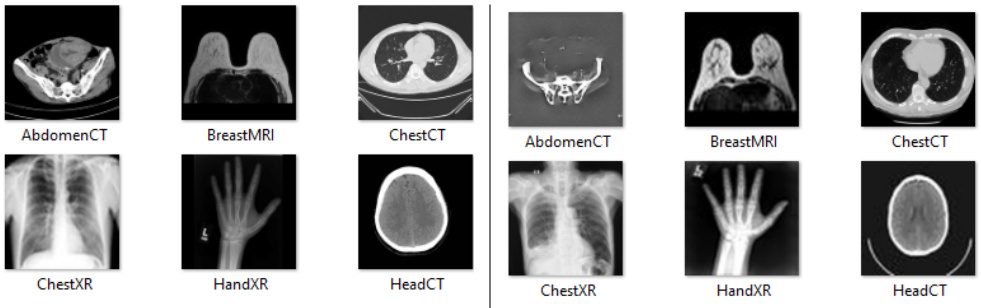
(a) Source Dataset (b) Target Dataset

Figure 10: Side-by-side Preview of the Dataset-1, Source Dataset (a) and Target Dataset (b).



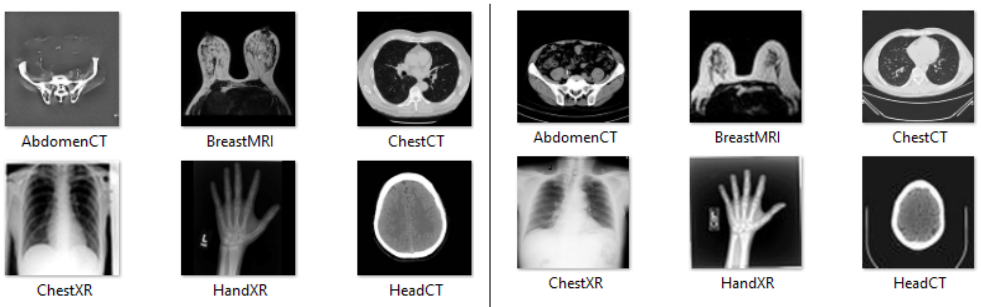
(a) Source Dataset (b) Target Dataset

Figure 11: Side-by-side Preview of the Dataset-2, Source Dataset (a) and Target Dataset (b).



(a) Source Dataset (b) Target Dataset

Figure 12: Side-by-side Preview of the Dataset-3, Source Dataset (a) and Target Dataset (b).



(a) Source Dataset (b) Target Dataset

Figure 13: Side-by-side Preview of the Dataset-4, Source Dataset (a) and Target Dataset (b).

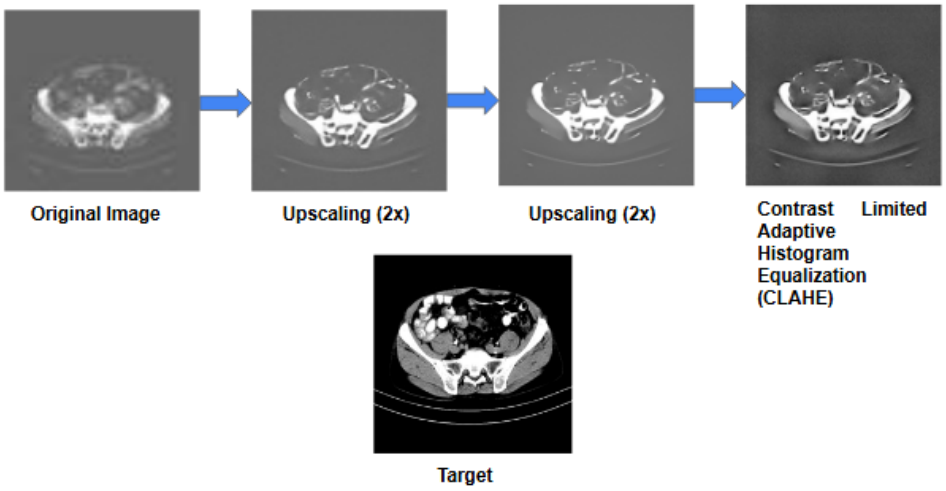


Figure 14: Transformation of AbdomenCT

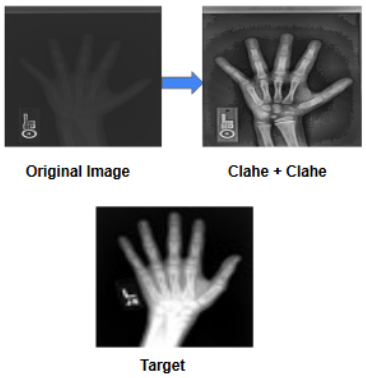


Figure 15: Transformation of HandXR

All the datasets, as shown in Figures 10,11, 12, and 13 were tested in VefNet CNN feature extractor on DANN DA algorithm with a source pre-training and the usage of best checkpoints. This test had 130 samples per class in both the fully labelled source and the unlabelled target datasets. All the datasets were obtained from Kaggle. All the transformations in the datasets and the experiments were performed in Python. Every dataset had gone through replacements, transformations, flips (placement of source images in target from the previous set and vice-versa as required), and all such amendments were made through the classifications on Confusion Matrices, as shown in Figure 19. The description of the source and target subsets are given below-

- **Dataset-1:** In the Dataset-1, shown in Figure 10, which was used in the previous trimester, the source was obtained from the MedMNIST page, and the target was obtained from different Kaggle pages from the source.
- **Dataset-2:** The Dataset-2, as shown in Figure 11, retained the setup for source and target from Dataset-1 for the classes AbdomenCT, ChestXR, and HeadCT, replaced from alternative Kaggle pages for the classes BreastMRI, ChestCT, and HandXR for source, retained the target from Dataset-1 for ChestCT, and flipped the previous Dataset-1 source sets to the targets of BreastMRI and HandXR. The AbdomenCT of the source, as shown in Figure 14, had gone through upscaling (2x) to a high-quality resolution twice, as the image features had been in low resolution through an Enhanced Super Resolution Generative Adversarial Network (ESRGAN) [34]. This ESRGAN was cloned from GitHub, and weights from the repository were used for upscaling. In order to match the

lighting, clarity and background with those of the target set, Contrast Limited Adaptive Histogram Equalisation (CLAHE) [16] has been performed once on the upscaled AbdomenCT, and twice on the HandXR, as shown in Figures 14, 15.

- **Dataset-3:** The set-3, as shown in Figure 12, was retained from the set-2 fully in only BreastMRI, and flipped the source and target sets in AbdomenCT, ChestCT, and HeadCT. ChestXR's source and HandXR's targets were retained from set-1 and set-2. And, ChestXR's target, and HandXR's source were replaced from set-2 with an alternative Kaggle dataset.
- **Dataset-4:** The set-4, as shown in Figure 13, was retained from the set-2 fully for AbdomenCT, BreastMRI, ChestCT, ChestXR, and from set-3 for the HandXR, and HeadCT.

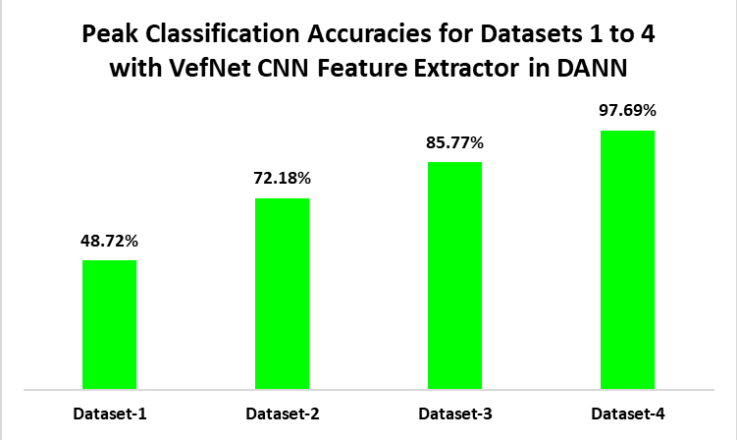


Figure 16: Results of DA with VefNet Feature Extractor in DANN

All these four datasets above were improvements of each other, as the peak accuracies suggest in Figure 16. Due to a peak accuracy of 97.69%, application of transformations only to the source set, and collection of images from various pages as expected in DA setups, the Dataset-4 was chosen as the final dataset for the DA studies further with the selected and proposed CNN architectures on a bigger dataset.

4.2 Preprocessing

Class	Count in Source	Count in Target	Size & Channels
AbdomenCT	500	500	224X224X3
BreastMRI	500	500	224X224X3
ChestCT	500	500	224X224X3
ChestXR	500	500	224X224X3
HandXR	500	500	224X224X3
HeadCT	500	500	224X224X3
Total	3000	3000	

Table 2: Summary of the Final Dataset after Preprocessing

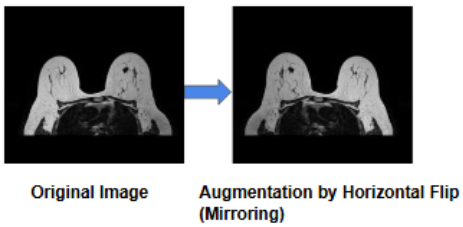


Figure 17: Augmentation of Breast MRI in the Source Set of the Final Dataset

The preprocessing, as shown in the Workflow subsection earlier, in Figure 1, mainly contained the resizing of images from various dimensions to 224x224 with three channels, as it was the required size of the selected CNNs, and normalisation was done for both the source and target datasets, by extracting the three-channel pre-normalisation means and standard deviations of both the sets and transforming the sets with the post-normalisation mean of 1, and standard deviation of 0, for a faster training, and to avoid issues like explosion/vanishing gradients [37]. As the number of images in the Final Dataset for BreastMRI in the source was low (286), they were augmented to 500 with the horizontal flip method to retain the features as it is, as shown in Figure 17, and all the labelled source and unlabelled target images contained 500 images in each of the classes, as noted in Table 2. The unlabelled dataset has been validated with the performance indicators through DA algorithms with a CSV file that contains the image names and class names for each image.

4.3 Setup and Evaluation Metrics

The preprocessing, source dataset pretraining, and DA experiments were performed through an A100 Graphical Processing Unit(GPU) in Google Colab on Python. The A100 GPU has a support system for Deep Learning tasks that require heavy computing [19], and a short feature summary is shown in Table 3-

GPU Architecture	Ampere
CUDA Cores	6,912
Memory	80 GB HBM2e
Memory Bandwidth	1,935 GB/s

Table 3: Specifications of the NVIDIA A100 GPU

As seen earlier, the source dataset has been split into 75% for training, 10% for validation, and 15% for testing in source pretraining with the CNNs. The source dataset is kept without any such split while being fed into DA algorithms with CNNs as feature extractors. The Key Performance Metrics (KPIs) were extracted from the test split set (10%) of the source dataset during pretraining and from the target dataset during prediction in DA experiments with DANN and CDAN+E. All the metrics were extracted with the scikit-learn library [4]. The confusion matrices, as shown in Figures 18 and 19, played an important role in the extraction of the KPIs. The extracted metrics were saved as an Excel sheet using the Pandas library. The batch size was fixed as 32 for all the CNN pretraining experiments, and for DA experiments with DANN, and CDAN+E, had the same batch size, and the epochs, learning rate, and optimiser were fixed as 10, 1e-4, and Adam.

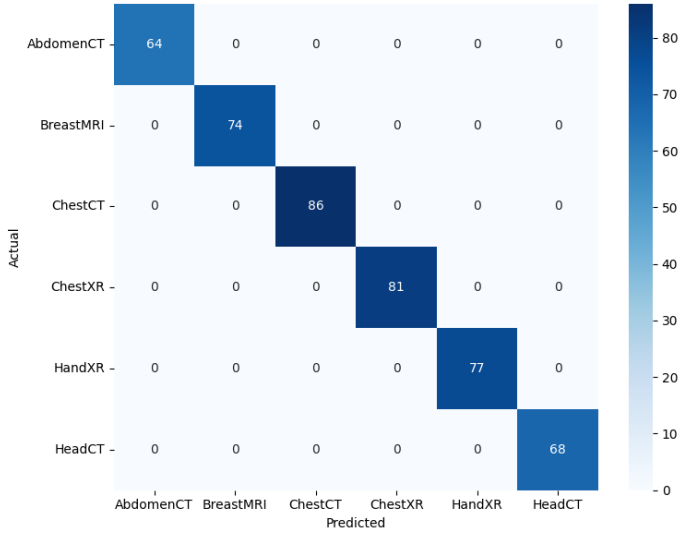


Figure 18: A sample Confusion Matrix of Source Pretraining (Test Set) from EfficientNet-B0 CNN for Adagrad Optimiser- 40 Epochs

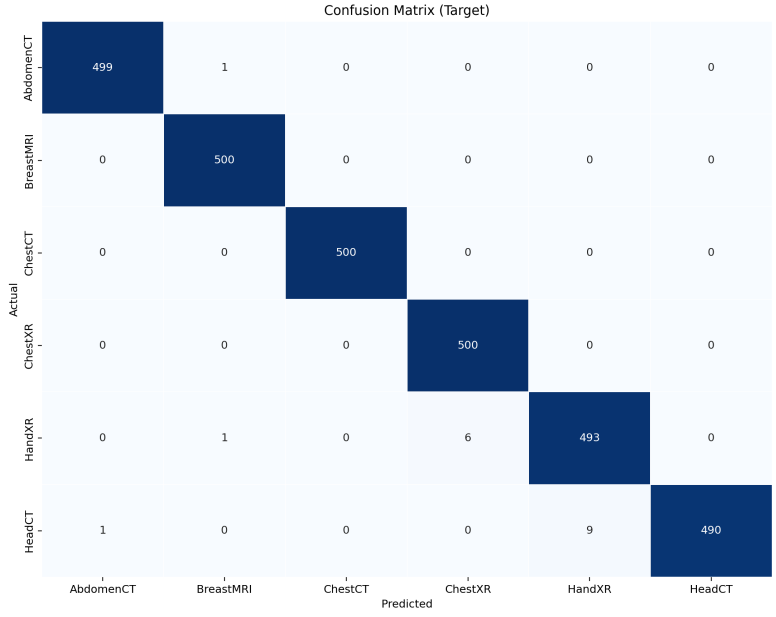


Figure 19: A sample Confusion Matrix of Domain Adaptation (Target Set) from VefNet CNN in CDAN+E for 0.5 lambda

The descriptions of the KPIs are given below, and it is to be noted that the confusion matrix-extracted prediction terms are expressed as True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN).

- **Train, Test Accuracies:** The ratio of the number of correctly classified images to the total number of images in the Train and Test split sets in pretraining, and the target set in DA. Expressed in %s.

- **Number of Correct Classifications:** The total number of correct classifications in DA algorithms (DANN, CDAN+E) with the CNN feature extractors for the target dataset. This is calculated through numpy as it goes through the CSV file for the target data, and compares the algorithm's prediction for every image with the original image and its respective class.

- **Precision:** The ratio of the number of TPs to the total number of positives for the test set of pretraining and the target set of DA [22]. A score close to 1 is expected for a good performance. The precision is expressed as

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

- **Sensitivity:** The ratio of the number of TPs to the number of TPs and FNs for the test set of pretraining and the target set of DA [22]. A score close to 1 is expected for a good performance. The sensitivity is expressed as

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

- **Specificity:** The ratio of the number of TNs to the number of TNs and FPs for the test set of pretraining and the target set of DA [22]. A score close to 1 is expected for a good performance. The specificity is expressed as

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

- **AUC-ROC:** The AUC-ROC is a score for the ability to distinguish between the classes. The Receiver Operating Characteristic (ROC) curve plots sensitivity in the Y-axis against the False Positive Rate (FPR) in the X-axis at various thresholds of unique softmax scores (processed class probability scores from logits, which are unprocessed) [5]. A score close to 1 is expected for a good performance.

The FPR is expressed as

$$\text{False Positive Rate (FPR)} = \frac{FP}{FP + TN} = 1 - \text{Specificity}$$

- **Inference Time:** The inference time is a measure of how long an algorithm takes to predict an image on the target dataset [18]. An average inference time is calculated for each of the checkpoints for predicting 100 images, and the mean of all the average inference times of the checkpoints is measured. Expressed in milliseconds.

4.4 Results of Source Pretraining with CNNs

Opt-Epch	**Tr.Acc**	**Te.acc**	**Prec**	**Sens**	**Spec**	**AUC-ROC**
Adam-40	100%	100%	1	1	1	1
RMSprop-80	100%	99.78%	0.998	0.998	1	0.999
Adagrad-40	100%	99.78%	0.998	0.998	1	1
SGD-40	100%	99.56%	0.996	0.995	0.999	1
Adagrad-5	99.91%	99.56%	0.996	0.995	0.999	0.999

Table 4: Top-5 Source Pretraining Results of VGG-19 stored as .pth checkpoints (Arranged by Test Accuracy Scores)

Opt-Epch	**Tr.Acc**	**Te.Acc**	**Prec**	**Sens**	**Spec**	**AUC-ROC**
Adam-20	100%	100%	1	1	1	1
Adam-40	100%	100%	1	1	1	1
RMSprop-40	100%	100%	1	1	1	1
Adagrad-20	100%	100%	1	1	1	1
Adagrad-40	100%	100%	1	1	1	1

Table 5: Top-5 Source Pretraining Results of EfficientNet-B0 stored as .pth checkpoints (Arranged by Test Accuracy Scores)

Opt-Epch	Tr.Acc	Te.Acc	Prec	Sens	Spec	AUC-ROC
SGD-80	100%	100%	1	1	1	1
Adam-20	100%	100%	1	1	1	1
RMSprop-20	100%	100%	1	1	1	1
Adagrad-10	100%	100%	1	1	1	1
AdamW-40	100%	100%	1	1	1	1

Table 6: Top-5 Source Pretraining Results of EfficientNet-V2-S stored as .pth checkpoints (Arranged by Test Accuracy Scores)

Opt-Epch	Tr.Acc	Te.Acc	Prec	Sens	Spec	AUC-ROC
Adam-20	100%	100%	1	1	1	1
Adam-40	100%	100%	1	1	1	1
RMSprop-40	100%	100%	1	1	1	1
Adagrad-10	100%	100%	1	1	1	1
AdamW-5	100%	100%	1	1	1	1

Table 7: Top-5 Source Pretraining Results of MobileNet-V2 stored as .pth checkpoints (Arranged by Test Accuracy Scores)

Opt-Epch	Tr.Acc	Te.Acc	Prec	Sens	Spec	AUC-ROC
Adam-20	100%	100%	1	1	1	1
Adam-80	100%	100%	1	1	1	1
RMSprop-80	100%	100%	1	1	1	1
Adagrad-10	100%	100%	1	1	1	1
AdamW-40	100%	100%	1	1	1	1

Table 8: Top-5 Source Pretraining Results of ResNet-50 stored as .pth checkpoints (Arranged by Test Accuracy Scores)

Opt-Epch	Tr.Acc	Te.Acc	Prec	Sens	Spec	AUC-ROC
SGD-40	100%	100%	1	1	1	1
Adam-40	100%	100%	1	1	1	1
RMSprop-10	100%	100%	1	1	1	1
RMSprop-20	100%	100%	1	1	1	1
Adagrad-10	100%	100%	1	1	1	1

Table 9: Top-5 Source Pretraining Results of VefNet stored as .pth checkpoints (Arranged by Test Accuracy Scores)

The top performances across the hyperparameters (Optimisers, Epochs) of the source dataset pretraining in the CNNs were stored as checkpoints.

Though, these performances may not be the actual top-5 in the original results, while sorted by test accuracies, these checkpoints were hand-picked among the best performances by looking into the train and test accuracies to avoid overfitting. Despite VGG-19 having an almost close to perfect performance, as observed in Table 4, all the other models had perfect performances with 100% accuracies, and a whole score of 1 for precision, sensitivity, specificity, and Auc-Roc in the chosen checkpoints as found in the Tables 5, 6, 7, 8, and 9. Each of these checkpoints will be initialised for both DANN and CDAN+E's CNN feature extractors for a pretrained deep knowledge about the source dataset, and will be evaluated on the target dataset with the variations in lambda from 0.01 to 1.

4.5 Results of DANN with CNN Feature Extractors

Checkpoint	**Lambda**	**Te.Acc**	**Prec**	**Sens**	**Spec**	**AUC-ROC**
Adam_ep40	0.1	99.97%	0.999	0.999	0.999	1
SGD_ep40	0.1	99.97%	0.999	0.999	0.999	1
Adam_ep40	0.05	99.87%	0.999	0.999	0.999	0.999
Adam_ep40	0.01	99.80%	0.998	0.998	0.999	0.999
Adagrad_ep40	0.1	99.63%	0.996	0.996	0.999	0.999

Table 10: Top-5 Domain Adaptation Performances of DANN with VGG-19 Feature Extractor (Arranged by Test Accuracy Scores)

Checkpoint	**Lambda**	**Te.Acc**	**Prec**	**Sens**	**Spec**	**AUC-ROC**
Adam_ep20	0.01	80.20%	0.712	0.802	0.960	0.948
Adagrad_ep20	0.01	75.83%	0.746	0.758	0.952	0.891
Adam_ep40	0.5	75.10%	0.685	0.751	0.950	0.867
RMSprop_ep40	0.01	71.30%	0.676	0.713	0.943	0.912
Adagrad_ep40	0.01	70.67%	0.647	0.707	0.941	0.909

Table 11: Top-5 Domain Adaptation Performances of DANN with EfficientNet-B0 Feature Extractor (Arranged by Test Accuracy Scores)

Checkpoint	Lambda	Te.Acc	Prec	Sens	Spec	AUC-ROC
AdamW_ep40	0.05	99.53%	0.995	0.995	0.999	0.999
Adam_ep20	0.01	99.53%	0.995	0.995	0.999	0.999
Adam_ep20	0.05	99.53%	0.995	0.995	0.999	0.999
Adam_ep20	0.1	99.13%	0.991	0.991	0.998	0.999
RMSprop_ep20	0.01	98.73%	0.988	0.987	0.997	0.999

Table 12: Top-5 Domain Adaptation Performances of DANN with EfficientNet-V2-S Feature Extractor (Arranged by Test Accuracy Scores)

Checkpoint	Lambda	Te.Acc	Prec	Sens	Spec	AUC-ROC
Adam_ep20	0.01	82.47%	0.716	0.825	0.965	0.976
Adagrad_ep10	0.01	81.93%	0.733	0.819	0.964	0.892
RMSprop_ep40	0.01	80.60%	0.785	0.806	0.961	0.970
RMSprop_ep40	0.05	76.33%	0.714	0.763	0.953	0.934
AdamW_ep5	0.05	73.87%	0.715	0.739	0.948	0.802

Table 13: Top-5 Domain Adaptation Performances of DANN with MobileNet-V2 Feature Extractor (Arranged by Test Accuracy Scores)

Checkpoint	Lambda	Te.Acc	Prec	Sens	Spec	AUC-ROC
Adagrad_ep10	0.5	77.67%	0.703	0.777	0.955	0.937
Adam_ep20	0.5	77.20%	0.719	0.772	0.954	0.956
Adagrad_ep10	0.1	72.70%	0.693	0.727	0.945	0.878
Adam_ep20	0.1	66.53%	0.666	0.665	0.933	0.877
Adam_ep80	0.05	66.50%	0.663	0.665	0.933	0.838

Table 14: Top-5 Domain Adaptation Performances of DANN with ResNet-50 Feature Extractor (Arranged by Test Accuracy Scores)

Checkpoint	Lambda	Te.Acc	Prec	Sens	Spec	AUC-ROC
RMSprop_ep20	0.5	90.37%	0.916	0.904	0.981	0.992
RMSprop_ep10	0.01	77.30%	0.675	0.773	0.955	0.950
SGD_ep40	0.05	70.07%	0.694	0.701	0.940	0.731
Adagrad_ep10	0.05	66.60%	0.666	0.666	0.933	0.754
RMSprop_ep20	1	66.60%	0.655	0.666	0.933	0.904

Table 15: Top-5 Domain Adaptation Performances of DANN with VefNet Feature Extractor (Arranged by Test Accuracy Scores)

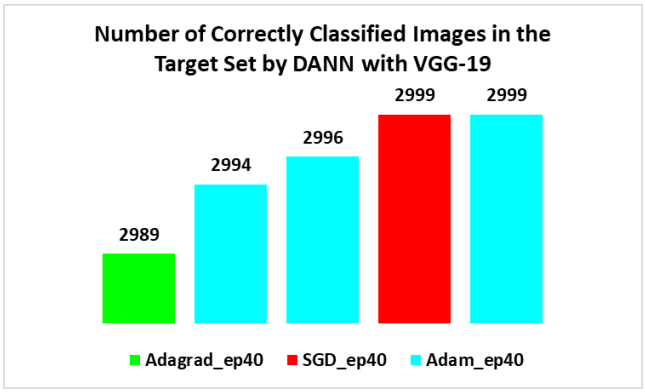


Figure 20: Number Correct Classifications of Images by DANN with VGG-19 for the Top-5 Domain Adaptation Performances

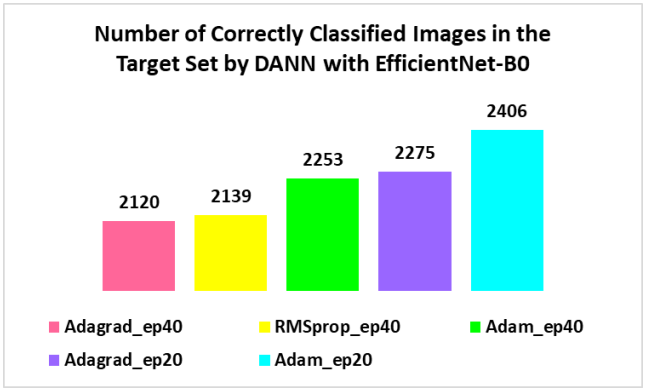


Figure 21: Number Correct Classifications of Images by DANN with EfficientNet-B0 for the Top-5 Domain Adaptation Performances

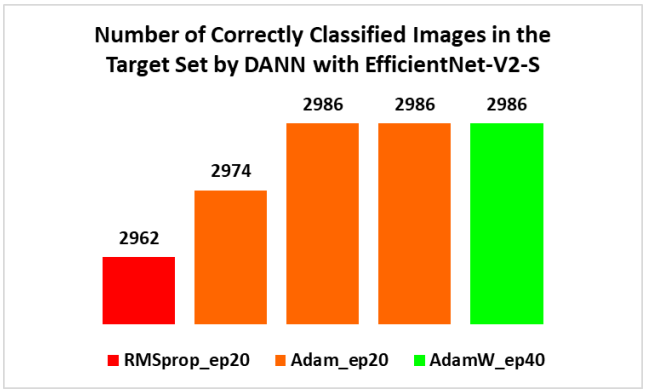


Figure 22: Number Correct Classifications of Images by DANN with EfficientNet-V2-S for the Top-5 Domain Adaptation Performances

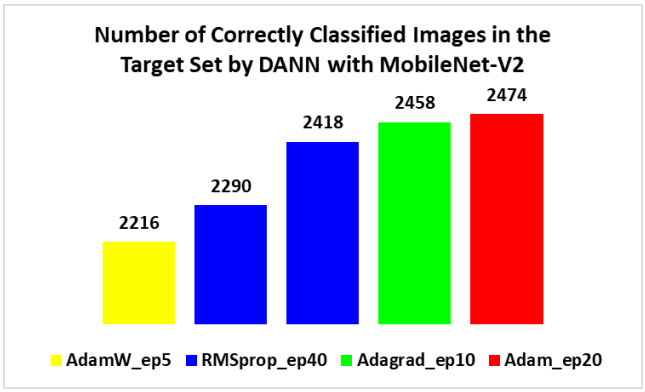


Figure 23: Number Correct Classifications of Images by DANN with MobileNet-V2 for the Top-5 Domain Adaptation Performances

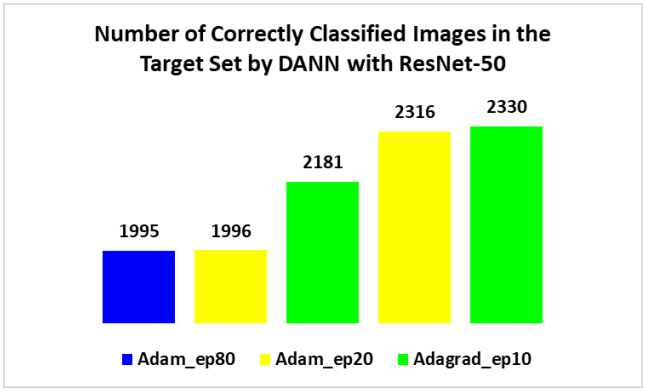


Figure 24: Number Correct Classifications of Images by DANN with ResNet-50 for the Top-5 Domain Adaptation Performances

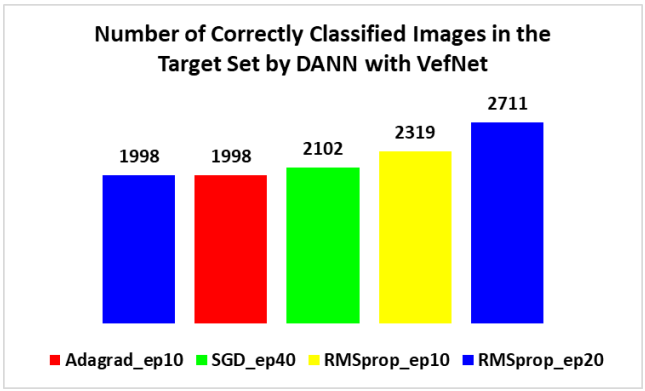


Figure 25: Number Correct Classifications of Images by DANN with VefNet for the Top-5 Domain Adaptation Performances

The VGG-19, as observed from Table 10, had a peak test accuracy of 99.97% in DANN, and a peak correct classification score of 2999 images in the 3000 image target set, as seen in Figure 20. The lambda value 0.1, for the checkpoints Adam and SGD with 40 epochs, had perfect sensitivity, specificity, precision, and AUC-ROC. The EfficientNet-B0, as shown in Table 11 and Figure 21, performed less well compared to VGG-19, achieving a peak accuracy of 80.20% (correct target-set classification score: 2406 images) and yielding precisions ranging from 0.55 to 0.75 across various lambdas and checkpoints. The specificities (always above 0.85) were better than sensitivities (0.32 to 0.70), and the AUC-ROCs were decent from 0.65 to around 0.95. The EfficientNet-V2-S had two of its checkpoints, AdamW, and Adam, with 40 and 20 epochs, achieving the peak accuracy of 99.53% (2986 images) as seen from Table 12 and Figure 22. The specificities and AUC-ROCs were in the range of 0.78 to 1.

As observed from Table 13 and Figure 23, the ResNet-50 has slightly disappointed with a peak test accuracy of 77.67% (2330 images) for Adagrad with 10 epochs, and the signs of overfitting were observed in precisions, sensitivities in the range of 0.3 to 0.8, and had good sensitivities, and AUC-ROCs in the range of 0.74 to 0.94. The MobileNet-V2, with lower parameters than ResNet-50, has surprisingly performed better with a peak test accuracy of 82.47% (2474 images), as seen in Table 14, and Figure 24 and had precisions, sensitivities, specificities in the range of 0.3 to 0.97 and AUC-ROCs were always above 0.71. The VefNet has performed well with a test accuracy of 90.37% (2711 images), as found in Table 15, and Figure 25. VefNet also had specificities and precisions in the range of 0.29 to 0.91, and sensitivities, AUC-ROCs in the range 0.85 to 0.99. It was also noted that the lambda values of 0.01 to 0.5 had given the decent to best performances for EfficientNet-B0, EfficientNet-V2-S, ResNet-50 and VefNet, though VGG-19, and MobileNet-V2 did well from 0.01 to 0.1. For EfficientNet-B0, the lambda value of 1 had always suffered from a lacklustre performance, while the dull performances of the other CNNs took lambda values from 0.5 to 1 mostly.

4.6 Results of CDAN+E with CNN Feature Extractors

Checkpoint	Lambda	Te.Acc	Prec	Sens	Spec	AUC-ROC
Adagrad_ep5	0.05	100%	1	1	1	1
SGD_ep40	0.1	100%	1	1	1	0.999
Adagrad_ep5	0.01	99.97%	0.999	0.999	0.999	1
Adam_ep40	0.1	99.97%	0.999	0.999	0.999	1
Adam_ep40	0.5	99.97%	0.999	0.999	0.999	1

Table 16: Top-5 Domain Adaptation Performances of CDAN+E with VGG-19 Feature Extractor (Arranged by Test Accuracy Scores)

Checkpoint	Lambda	Te.Acc	Prec	Sens	Spec	AUC-ROC
Adam_ep20	1	98.33%	0.984	0.983	0.997	0.999
Adam_ep20	0.1	89.63%	0.926	0.896	0.979	0.982
Adam_ep20	0.5	88.07%	0.929	0.881	0.976	0.982
Adam_ep20	0.05	81.63%	0.734	0.816	0.963	0.848
Adagrad_ep40	0.1	81.40%	0.735	0.814	0.963	0.901

Table 17: Top-5 Domain Adaptation Performances of CDAN+E with EfficientNet-B0 Feature Extractor (Arranged by Test Accuracy Scores)

Checkpoint	Lambda	Te.Acc	Prec	Sens	Spec	AUC-ROC
AdamW_ep40	0.05	99.87%	0.999	0.999	0.999	0.999
Adam_ep20	1	99.83%	0.998	0.998	0.999	0.999
Adam_ep20	0.1	99.77%	0.998	0.998	0.999	0.999
Adam_ep20	0.5	99.73%	0.997	0.997	0.999	0.999
RMSprop_ep20	0.01	99.73%	0.997	0.997	0.999	0.999

Table 18: Top-5 Domain Adaptation Performances of CDAN+E with EfficientNet-V2-S Feature Extractor (Arranged by Test Accuracy Scores)

Checkpoint	Lambda	Te.Acc	Prec	Sens	Spec	AUC-ROC
RMSprop_ep40	1	99.10%	0.991	0.991	0.998	0.999
AdamW_ep5	1	98.87%	0.989	0.989	0.998	0.999
Adam_ep40	1	96.53%	0.971	0.965	0.993	0.999
RMSprop_ep40	0.5	95.03%	0.954	0.950	0.990	0.999
Adam_ep20	0.1	83.27%	0.746	0.833	0.967	0.997

Table 19: Top-5 Domain Adaptation Performances of CDAN+E with MobileNet-V2 Feature Extractor (Arranged by Test Accuracy Scores)

Checkpoint	Lambda	Te.Acc	Prec	Sens	Spec	AUC-ROC
Adam_ep20	0.1	82.13%	0.744	0.821	0.964	0.898
Adam_ep20	0.05	78.87%	0.744	0.789	0.958	0.862
Adagrad_ep10	0.1	66.63%	0.601	0.666	0.933	0.796
Adam_ep80	0.01	66.60%	0.666	0.666	0.933	0.873
AdamW_ep40	0.01	66.57%	0.597	0.666	0.933	0.871

Table 20: Top-5 Domain Adaptation Performances of CDAN+E with ResNet-50 Feature Extractor (Arranged by Test Accuracy Scores)

Checkpoint	Lambda	Te.Acc	Prec	Sens	Spec	AUC-ROC
RMSprop_ep20	0.5	99.40%	0.994	0.994	0.999	0.999
RMSprop_ep20	1	98.40%	0.985	0.984	0.997	0.999
Adagrad_ep10	1	82.40%	0.866	0.824	0.965	0.955
RMSprop_ep20	0.05	81.83%	0.734	0.818	0.964	0.802
RMSprop_ep10	0.5	69.03%	0.664	0.690	0.938	0.881

Table 21: Top-5 Domain Adaptation Performances of CDAN+E with VefNet Feature Extractor (Arranged by Test Accuracy Scores)

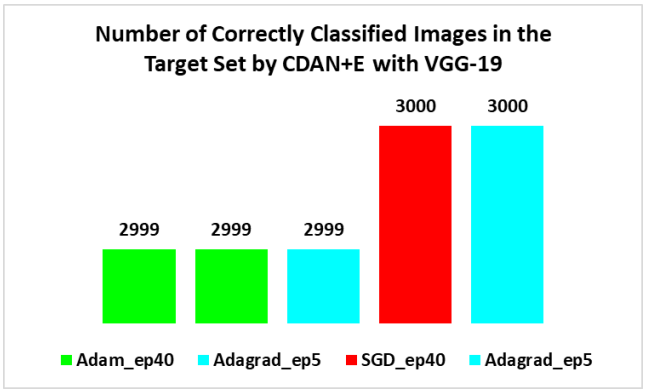


Figure 26: Number Correct Classifications of Images by CDAN+E with VGG-19 for the Top-5 Domain Adaptation Performances

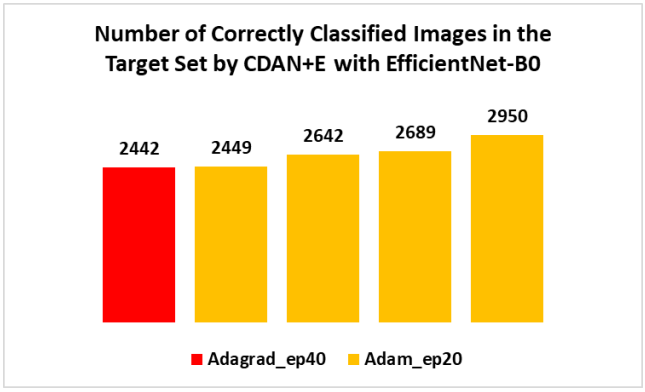


Figure 27: Number Correct Classifications of Images by CDAN+E with EfficientNet-B0 for the Top-5 Domain Adaptation Performances

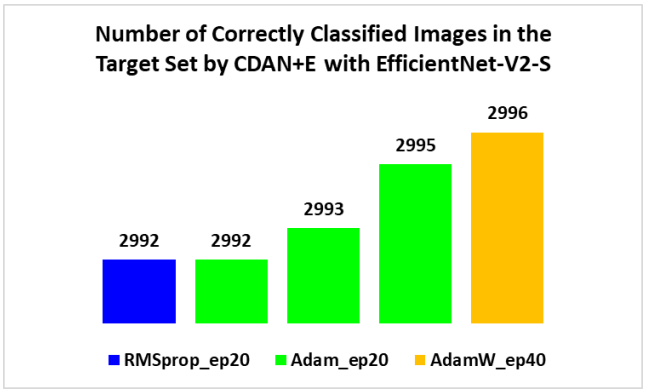


Figure 28: Number Correct Classifications of Images by CDAN+E with EfficientNet-V2-S for the Top-5 Domain Adaptation Performances

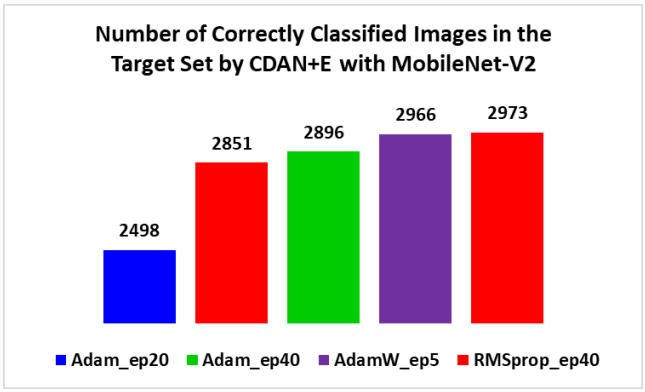


Figure 29: Number Correct Classifications of Images by CDAN+E with MobileNet-V2 for the Top-5 Domain Adaptation Performances

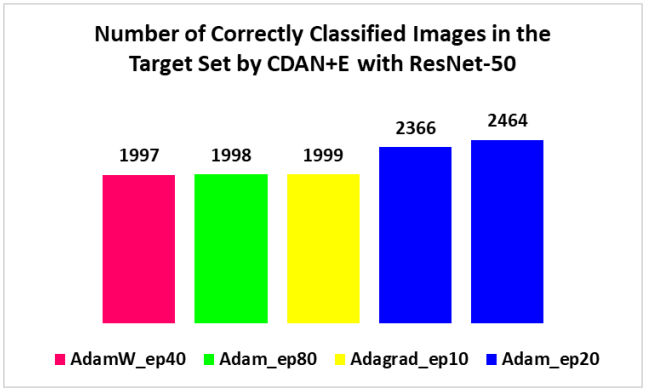


Figure 30: Number Correct Classifications of Images by CDAN+E with ResNet-50 for the Top-5 Domain Adaptation Performances

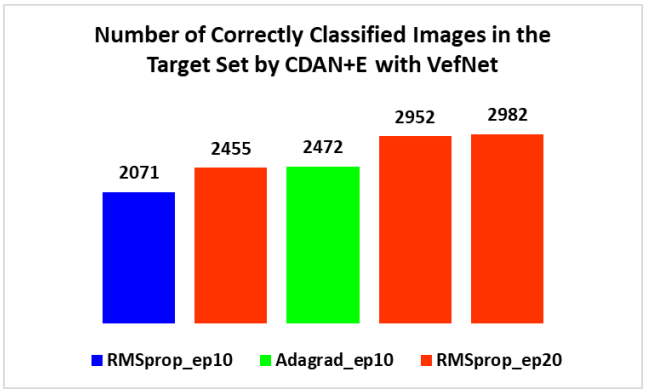


Figure 31: Number Correct Classifications of Images by CDAN+E with VefNet for the Top-5 Domain Adaptation Performances

The improved CDAN+E algorithm, for every CNN architecture, gave better performances than DANN. The VGG-19, as observed in Table 16 and Figure 26, had 100% test accuracy twice in classifying the 3000 images target. Like VGG-19, the SGD optimiser with a 40-epoch checkpoint was one of the top performers, and specificities were always above 0.895, which were better than precision and sensitivity across various lambdas and checkpoints. The EfficientNet-B0, as seen in Table 17 and Figure 27, also showed a tremendous improvement in classifying 2950 target images (98.33% test accuracy), with Adam optimiser for 20 epochs. The precisions and sensitivities were in the range of around 0.375 to 0.99, and the specificities and AUC-ROCs were always above 0.61 across the lambda values and checkpoints. The EfficientNet-V2-S, as noted in Table 18 and Figure 28, also had a slight improvement to DANN, with a peak test accuracy of 99.87% (2996 images) for AdamW with 40-epoch checkpoint. The sensitivities and precisions were always above 0.59, and the sensitivities, AUC-ROCs were always above 0.76 across checkpoints and lambdas.

The MobileNet-V2, as seen in Table 19 and Figure 29, had a peak test accuracy of 99.10% (2973 images). As repeated in various methods, the sensitivities and precisions were above 0.41, and the specificities and AUC-ROCs were always above 0.89 and 0.55 across various test cases. The ResNet-50 in CDAN+E, as noted in Table 20 and Figure 30, had an improvement to DANN, with a peak test accuracy of 82.13% (2464 images), and sensitivities, precisions were in the range of 0.34 to 0.82, and specificities and AUC-ROCs were in the range of 0.57 to 0.96 across various cases. VefNet had a slight improvement over the contemporaries MobileNet-V2 and EfficientNet-B0, as seen in Table 21 and Figure 31, with a peak accuracy of 99.40% (2982 images), and precisions and sensitivities were in the range of 0.26 to 0.99 and specificities, AUC-ROCs were in the range of 0.48 to 0.99 across all the checkpoints and lambda val. The VGG-19, EfficientNet-V2-S, and MobileNet-V2 had almost all the lambda values from 0.01 to 1 in both good and bad performances, while EfficientNet-B0, and VefNet CNNs had good performances with the lambda values from 0.05 to 1, and the worst performances had lambda values of mostly 0.01 and 0.05.

4.7 Overall Trends

Model	Lambda	Te.Acc	Prec	Sens	Spec	AUC-ROC
VGG-19	0.1	99.97%	1	1	1	1
EfficientNet-V2-S	0.05	99.53%	0.995	0.995	0.999	0.999
VefNet	0.5	90.37%	0.916	0.904	0.981	0.992
MobileNet-V2	0.01	82.47%	0.716	0.825	0.965	0.976
EfficientNet-B0	0.01	80.20%	0.712	0.802	0.960	0.948
ResNet-50	0.5	77.67%	0.703	0.777	0.955	0.937

Table 22: Best Domain Adaptation Performances of DANN with all the Experimented CNN Feature Extractors (Arranged by Test Accuracy Scores)

Model	Lambda	Te.Acc	Prec	Sens	Spec	AUC-ROC
VGG-19	0.05	100%	1	1	1	1
EfficientNet-V2-S	0.05	99.87%	0.999	0.999	0.999	0.999
VefNet	0.5	99.40%	0.994	0.994	0.999	0.999
MobileNet-V2	1	99.10%	0.991	0.991	0.998	0.999
EfficientNet-B0	1	98.33%	0.984	0.983	0.997	0.999
ResNet-50	0.1	82.13%	0.744	0.821	0.964	0.898

Table 23: Best Domain Adaptation Performances of CDAN+E with all the Experimented CNN Feature Extractors (Arranged by Test Accuracy Scores)

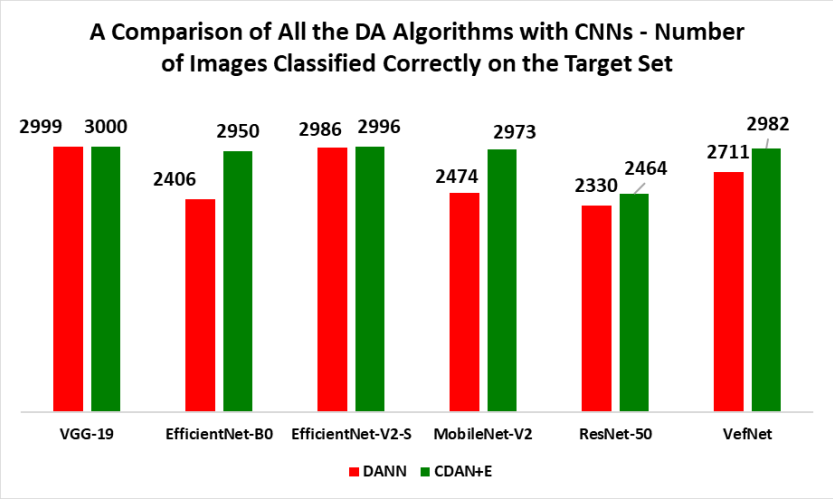


Figure 32: The Peaks of Number Correct Classifications of Images by all the CNN Feature Extractors in DANN and CDAN+E

Model	**DANN**	**CDAN+E**
VGG-19	**1.60**	**1.80**
EfficientNet-B0	7.88	7.81
EfficientNet-V2-S	**16.39**	**16.11**
MobileNet-V2	5.29	5.44
ResNet-50	5.90	5.94
VefNet	5.43	5.48

Table 24: Results of the Average Inference Times of CNNs in DA Algorithms for Predicting 100 Target Data Images (Expressed in Milliseconds)

The proposed VefNet architecture has performed with a 90.37% on DANN (2711 images) and 99.40% on CDAN+E (2982 images) with a moderate number of trainable parameters (~16.7 M), as seen from Figure 32 and Tables 22 and 23. The direct contemporary model MobileNet-V2, with the lowest number of trainable parameters (~2.23 M), has significantly challenged the VefNet with peak test accuracies of 82.47% in DANN (2494 images), and 99.10% in CDAN+E (2973 images). While the inspiration VGG-19 (~140 M parameters) achieves peak test accuracies of 99% and 100% in DANN and CDAN+E, due to a slight underperformance in source pretraining, a strong sign of underfitting is indicated. A vast number of training parameters make it a memory-hostile architecture under limited hardware.

The EfficientNet-V2-S (~38.17 M) offers a strong performance of above 99% in both algorithms, almost double the trainable parameters of VefNet, making it another model prone to memory issues under constrained hardware conditions. EfficientNet-B0 (~7.16 M) shows a great improvement in CDAN+E with 98.33% compared to DANN with 80.20%, and ResNet-50 (~23.53 M parameters) struggles to cross 82.13% as seen from CDAN+E. The precisions and sensitivities have been slightly lower than specificities, and AUC-ROCs usually, although they have been above 0.7, mostly in the peak performances. The average inference times (in milliseconds) taken for predicting 100 images in the target set by all the checkpoints indicate that EfficientNet-V2-S took the highest time of more than 16ms, and VGG-19 had the lowest inference times of around 1.7ms in both algorithms. The MobileNet-V2's inference time takes a slight lead ahead of VefNet in both the DANN and CDAN+E, while both of them are around 5.29 to 5.48ms.

5 Conclusion

To conclude, this research showed, the memory-friendly VefNet CNN as a feature extractor in DANN and CDAN+E algorithms has successfully classified 2982 images on a 3000-image target dataset in the CDAN+E algorithm, and 2711 images in the DANN algorithm, although another memory-friendly MobileNet-V2 had a close identification of 2973 images in CDAN+E. The CDAN+E's multilinear and entropy conditioning has proved better for all the CNN feature extractors in the DA algorithms. The other tested CNN architectures for medical images either produced results with underfitting symptoms or proved unfit for memory-constrained hardware. Apart from that, the closest replacements of medical image datasets in source/target domains, data transformations have shown their abilities to perform well if the identifiable features are presented in images. As the experiments were run through an A100-GPU (the most advanced GPU in Colab), the GPUs with reduced efficiencies may increase the training time and inference time. Though most of the CNNs with DA algorithms predict all the classes, the AbdomenCT in the target suffers from slightly lower predictions in some cases due to the HD quality of images in the target data, and the target data expects more proximity to the source data, even in terms of quality.

To improve this research further, the latest image detection benchmarks can be integrated along with the DA algorithms for an effective Unsupervised Domain Adaptation, even without multiple data transformations and replacements. Furthermore, the study can be extended for the prediction of a closer class (eg, chest to lungs), or with different modalities (eg, CT to MRI), or for a different organism (eg, HandXR scans of humans in source, HandsXR scans of animals in target), which could be useful in disease/condition analysis and veterinary diagnosis. Apart from collecting the images of scans online for the target dataset, the scans of diseases without labels, from a medical centre, can be kept in the target, and doctors can diagnose diseases with the learning of the source datasets collected from online, thus providing a wake-up call for the patient's recovery or further assessment.

6 Code and Repository

The execution codes of source-pretraining and domain adaptations can be found at - **UDA with VefNet CNN for Medical Images.**

Acknowledgements

Dr Xinyu Zhang, Project Supervisor and Lecturer at University of Auckland, has given me guidance and support to complete the research and experiments.

A Appendices

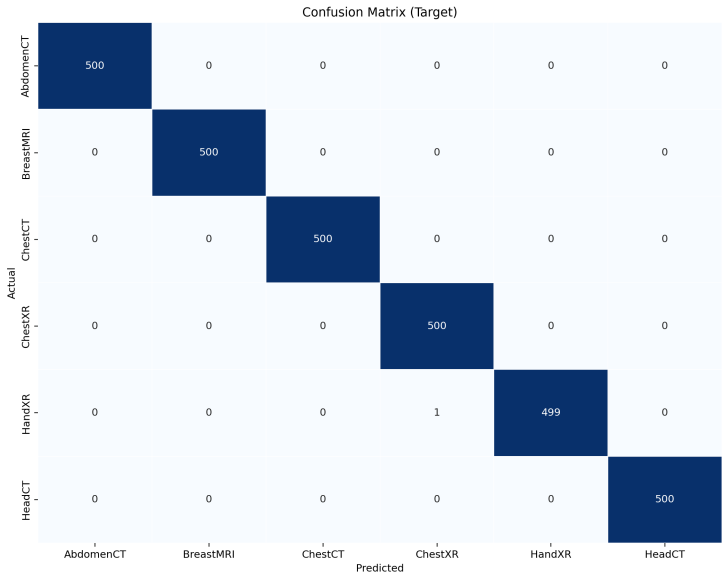


Figure 33: The Confusion Matrix of the DANN with VGG-19 Feature Extractor for the Peak Classification

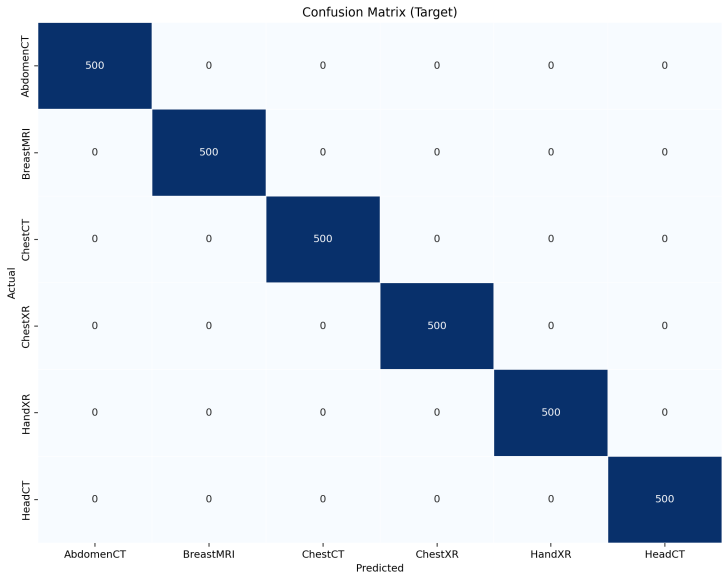


Figure 34: The Confusion Matrix of the CDAN+E with VGG-19 Feature Extractor for the Peak Classification

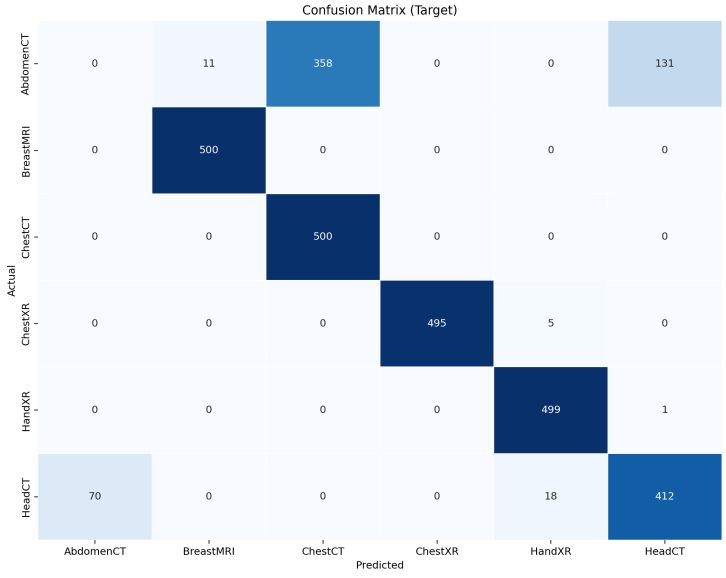


Figure 35: The Confusion Matrix of the DANN with EfficientNet-B0 Feature Extractor for the Peak Classification

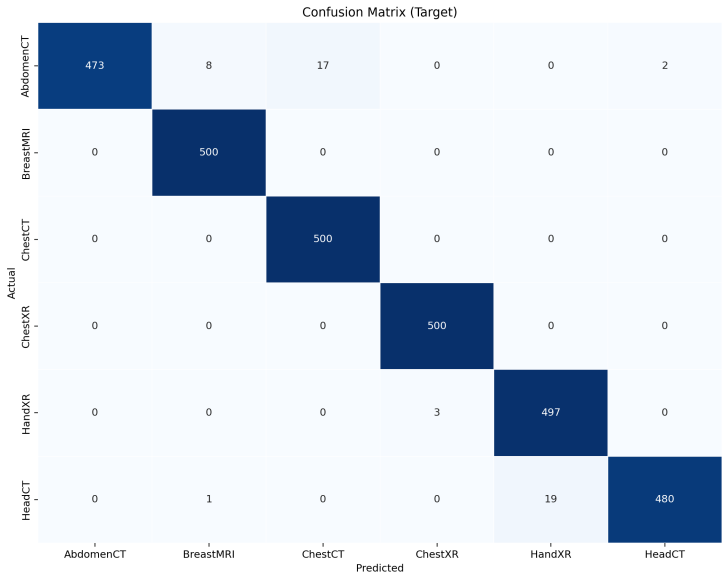


Figure 36: The Confusion Matrix of the CDAN+E with EfficientNet-B0 Feature Extractor for the Peak Classification

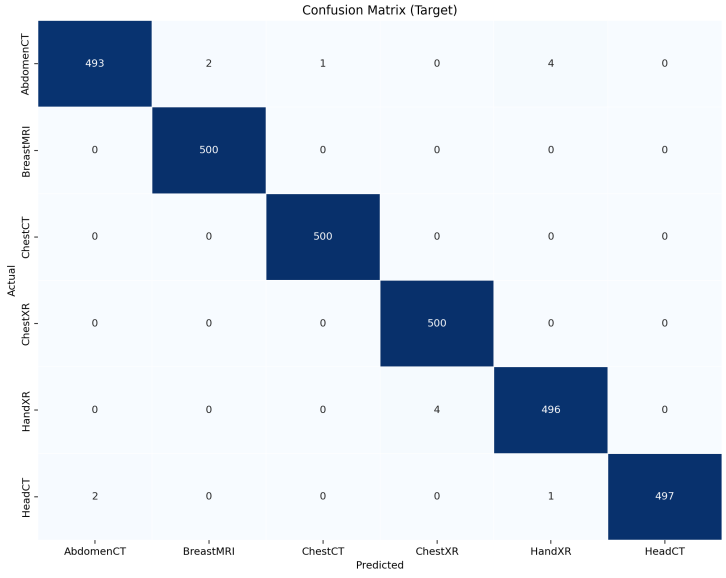


Figure 37: The Confusion Matrix of the DANN with EfficientNet-V2-S Feature Extractor for the Peak Classification

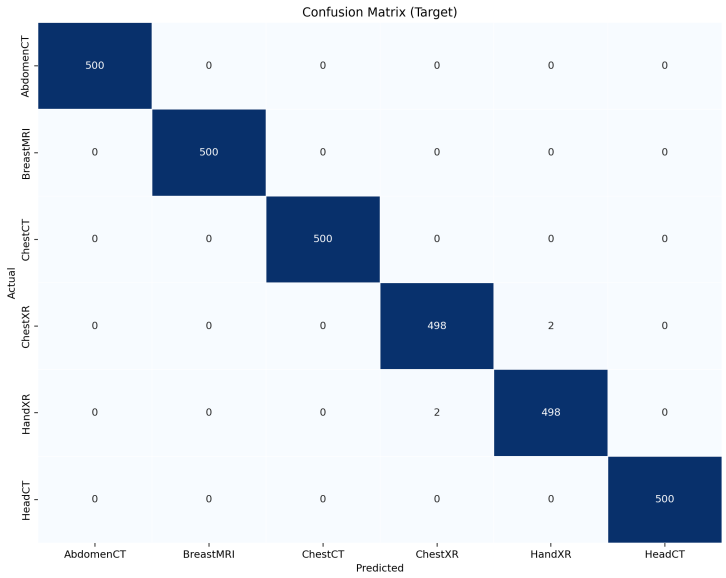


Figure 38: The Confusion Matrix of the CDAN+E with EfficientNet-V2-S Feature Extractor for the Peak Classification

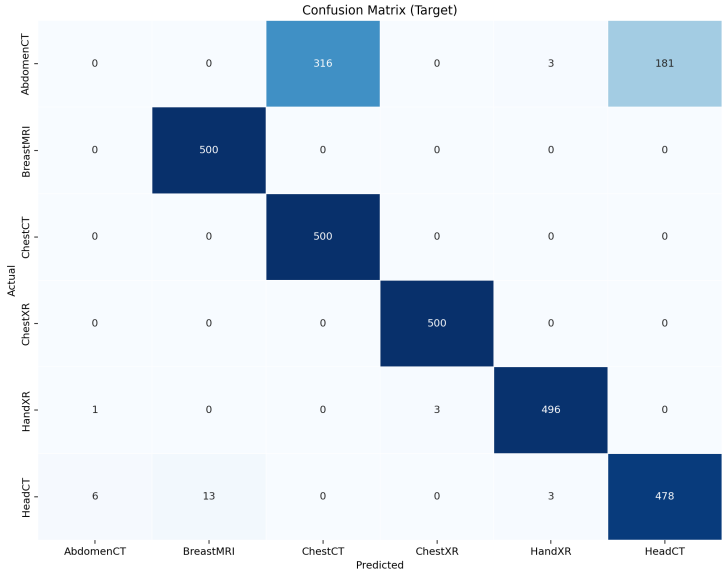


Figure 39: The Confusion Matrix of the DANN with MobileNet-V2 Feature Extractor for the Peak Classification

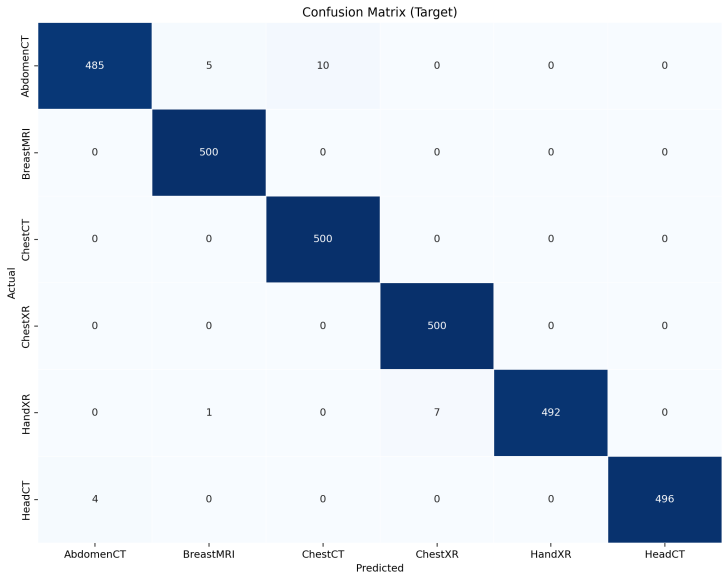


Figure 40: The Confusion Matrix of the CDAN+E with MobileNet-V2 Feature Extractor for the Peak Classification

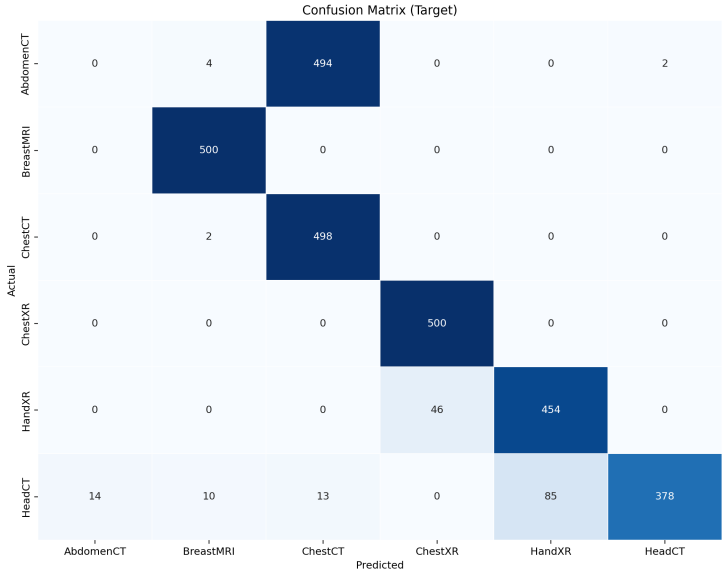


Figure 41: The Confusion Matrix of the DANN with ResNet-50 Feature Extractor for the Peak Classification

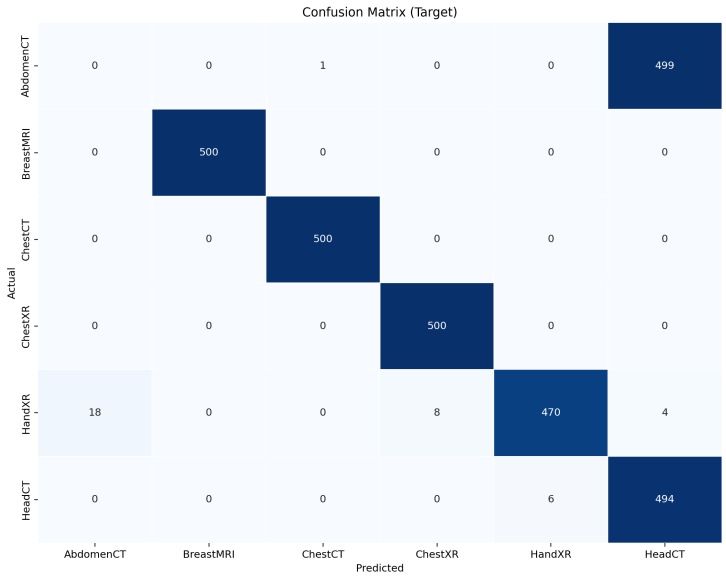


Figure 42: The Confusion Matrix of the CDAN+E with ResNet-50 Feature Extractor for the Peak Classification

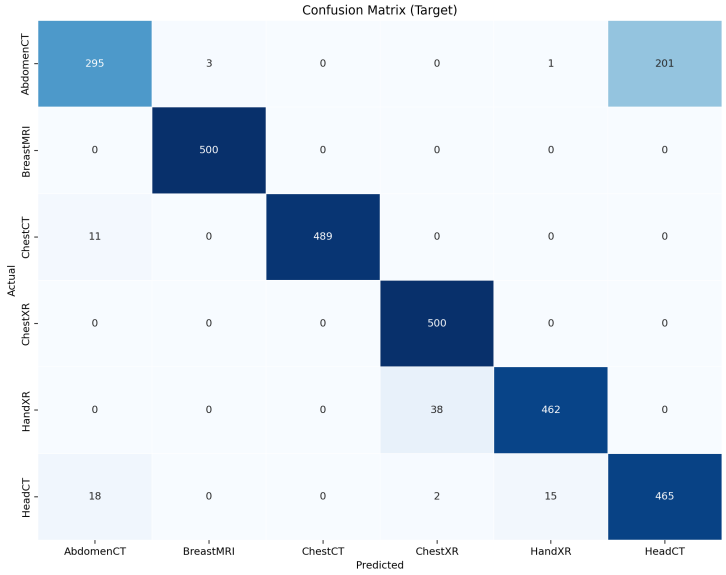


Figure 43: The Confusion Matrix of the DANN with VefNet Feature Extractor for the Peak Classification

References

[1] Abulikemu Abuduweili and Changliu Liu. Revisiting the initial steps in adaptive gradient descent optimization. *arXiv preprint arXiv:2412.02153*, 2024.

[2] Doğay Altınel. Development of deep learning optimizers: Approaches, concepts, and update rules. *arXiv preprint arXiv:2509.18396*, 2025.

[3] Aditi Anand, Sarada Krithivasan, and Kaushik Roy. Romia: a framework for creating robust medical imaging ai models for chest radiographs. *Frontiers in Radiology*, 3:1274273, 2024.

[4] Ekaba Bisong. Introduction to scikit-learn. In *Building machine learning and deep learning models on google cloud platform: a comprehensive guide for beginners*, pages 215–229. Springer, 2019.

[5] Tom Fawcett. An introduction to roc analysis. *Pattern recognition letters*, 27(8):861–874, 2006.

[6] 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.

[7] Xiaoting Han, Lei Qi, Qian Yu, Ziqi Zhou, Yefeng Zheng, Yinghuan Shi, and Yang Gao. Deep symmetric adaptation network for cross-modality medical image segmentation. *IEEE transactions on medical imaging*, 41(1):121–132, 2021.

[8] Bishi He, Yuanjiao Chen, Darong Zhu, and Zhe Xu. Domain adaptation via wasserstein distance and discrepancy metric for chest x-ray image classification. *Scientific Reports*, 14(1):2690, 2024.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[10] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018.

[11] Shih-Cheng Huang, Anuj Pareek, Malte Jensen, Matthew P Lungren, Serena Yeung, and Akshay S Chaudhari. Self-supervised learning for medical image classification: a systematic review and implementation guidelines. *NPJ Digital Medicine*, 6(1):74, 2023.

[12] Sian Jin, Chengming Zhang, Xintong Jiang, Yunhe Feng, Hui Guan, Guanpeng Li, Shuaiwen Leon Song, and Dingwen Tao. Comet: a novel memory-efficient deep learning training framework by using error-bounded lossy compression. *arXiv preprint arXiv:2111.09562*, 2021.

[13] Sorin Liviu Jurj, Sina Banasaz Nouri, and Jörg Strutwolf. Snntrainer3d: Training spiking neural networks using a user-friendly application with 3d architecture visualization capabilities. *Applied Sciences*, 14(13):5752, 2024.

[14] Adrian Shuai Li, Elisa Bertino, Xuan-Hong Dang, Ankush Singla, Yuhai Tu, and Mark N Wegman. Maximal domain independent representations improve transfer learning. *arXiv preprint arXiv:2306.00262*, 2023.

[15] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I Jordan. Conditional adversarial domain adaptation. *Advances in neural information processing systems*, 31, 2018.

[16] Akshansh Mishra. Contrast limited adaptive histogram equalization (clahe) approach for enhancement of the microstructures of friction stir welded joints. *arXiv preprint arXiv:2109.00886*, 2021.

[17] Aminu Musa, Rajesh Prasad, and Monica Hernandez. Addressing cross-population domain shift in chest x-ray classification through supervised adversarial domain adaptation. *Scientific Reports*, 15(1):11383, 2025.

[18] Canicius Mwitta, Glen C Rains, and Eric Prostko. Evaluation of inference performance of deep learning models for real-time weed detection in an embedded computer. *Sensors*, 24(2):514, 2024.

[19] NVIDIA Corporation. NVIDIA A100 Tensor Core GPU. `https://www.nvidia.com/en-au/data-center/a100/`, 2020. Accessed: 2025-11-23.

[20] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.

[21] Ferhat Sarikaya. Batch size selection in deep learning: A comprehensive analysis of training dynamics and performance optimization. *arXiv preprint / Zenodo*, 2024.

[22] B. R. N. Sathyanarayanan and Tantri Roopa. Confusion matrix–based performance evaluation metrics. *African Journal of Biomedical Research*, 2025. Available online via ResearchGate.

[23] Aiza Shabir, Khawaja Tehseen Ahmed, Khadija Kanwal, Arif Mehmood, Nagwan Abdel Samee, Muhammad Tahir Naseem, and Imran Ashraf. Advanced multilevel feature fusion framework for enhanced image retrieval using convolutional neural network and benchmark datasets. *Journal of Big Data*, 12(1):240, 2025.

[24] Mehwish Shaikh, Isma Farah Siddiqui, Qasim Arain, Jahwan Koo, Mukhtiar Ali Unar, and Nawab Muhammad Faseeh Qureshi. Mdev model: A novel ensemble-based transfer learning approach for pneumonia classification using cxr images. *Computer Systems Science and Engineering*, 46(1):287–302, 2023.

[25] Hyungseob Shin, Hyeongyu Kim, Sewon Kim, Yohan Jun, Taejoon Eo, and Dosik Hwang. Cosmos: cross-modality unsupervised domain adaptation for 3d medical image segmentation based on target-aware domain translation and iterative self-training. *arXiv preprint arXiv:2203.16557*, 2022.

[26] Seung Yeon Shin, Sungwon Lee, and Ronald M Summers. Unsupervised domain adaptation for small bowel segmentation using disentangled representation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 282–292. Springer, 2021.

[27] 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.

[28] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[29] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019.

[30] Mingxing Tan and Quoc Le. Efficientnetv2: Smaller models and faster training. In *International conference on machine learning*, pages 10096–10106. PMLR, 2021.

[31] Patrick Thiam, Ludwig Lausser, Christopher Kloth, Daniel Blaich, Andreas Liebold, Meinrad Beer, and Hans A Kestler. Unsupervised domain adaptation for the detection of cardiomegaly in cross-domain chest x-ray images. *Frontiers in Artificial Intelligence*, 6:1056422, 2023.

[32] Gijs van Tulder and Marleen de Bruijne. Unpaired, unsupervised domain adaptation assumes your domains are already similar. *Medical Image Analysis*, 87:102825, 2023.

[33] Lianyu Wang, Meng Wang, Daoqiang Zhang, and Huazhu Fu. Model barrier: A compact un-transferable isolation domain for model intellectual property protection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20475–20484, 2023.

[34] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Yu Qiao, and Chen Change Loy. Esrgan: Enhanced super-resolution generative adversarial networks. In *Proceedings of the European conference on computer vision (ECCV) workshops*, pages 0–0, 2018.

[35] Zhenbin Wang, Mao Ye, Xiatian Zhu, Liuhan Peng, Liang Tian, and Yingying Zhu. Metateacher: Coordinating multi-model domain adaptation for medical image classification. *Advances in Neural Information Processing Systems*, 35:20823–20837, 2022.

[36] Ting Xiao, Cangning Fan, Peng Liu, and Hongwei Liu. Simultaneously improve transferability and discriminability for adversarial domain adaptation. *Entropy*, 24(1):44, 2021.

[37] Yuqi Xiong and Wei Duan. Predictive capability evaluation of micrograph-driven deep learning for ti6al4v alloy tensile strength under varied preprocessing strategies. *Metals*, 15(6):586, 2025.