Lab Report (WiSe 2021/22)
Visualization and Medical Image Analysis

Entropy Guided Unsupervised Domain Adaptation for Segmentation of Brain MRI Scans

Sanket Rajendra Shah, Rohil Prakash Rao
University of Bonn, Department of Computer Science

Abstract

Image segmentation is an important part of most medical applications ranging from diagnostics to medicine research. With the advancement in data and computational resources, deep learning methods have been very effective to solve the segmentation task. However, deep learning based segmentation suffers from the problem of domain shift. A segmentation model trained on scans from a particular center or device (source domain) performs poorly when used for a novel center or device (target domain). To that end, in this lab, we implement a paper for Entropy guided domain adaptation by Zeng et. al. that introduces an adversarial learning based domain invariant segmentation model. The proposed model consists of a segmenter that is trained in a supervised manner to segment scans from the source domain. Additionally, the model contains two auxiliary discriminator networks, that discriminate between the segmenter features maps and the output entropy maps respectively, for the source and target domain. They propose using these discriminators to adversarially train the segmenter to produce domain invariant features maps and low entropy predictions. DICE Score and Surface Dice score were used to evaluate the segmentation performance and corresponding domain shift. In this work, we show a successful implementation of the original paper with a few crucial changes. Unlike the original paper, we use the U-Net architecture for segmentation. We also use a different dataset (Calgary Campinas Brain MRI data). In addition, we contribute to this work by experimenting with some other approaches. We investigate the effect of training the feature discriminator on both the source and target domains, unlike the original paper. We also show results with the addition of direct entropy minimization. Moreover, for comparison, we also produce results by jointly training the entire network from scratch, instead of fine-tuning the segmenter, as suggested in the original paper. In general, we present various successful domain adaptation methods, especially when the domain shift is high.

Keywords: *Unsupervised Domain Adaptation, Segmentation, Brain MRI*

1. Introduction

Magnetic resonance imaging (MRI) is a medical imaging technique that uses a magnetic field and digital radio waves to create digital images of the anatomical elements of the human body like organs, bones, tissues. Neuroscience researchers need to segment brain MRI images to quantitatively detect, analyze and study brain diseases. According to research, abnormal shape or volume of certain anatomical regions of the brain is related to brain diseases, such as Alzheimer's disease and Parkinson's disease. [AGH*17] Generally, accurate segmentation of brain tissue is obtained by manual segmentation by experienced brain experts. However, when faced with a large number of datasets, manual segmentation methods become quite expensive, time-consuming, and impractical. Moreover, due to differences in experience and knowledge among experts, the segmentation results are not uniform. Therefore, it is necessary to develop an automatic segmentation method for brain MRI.

For brain MRI segmentation, methods based on pattern recognition algorithms such as the support vector machine [CR08], random forest [Bre01] and neural network [Nie18] using features such as age, gender, ethnicity etc., and the combined methods of both are popularly used. However, most of the currently developed methods require explicit spatial and intensity information. Furthermore, it is required to extract feature vectors from the intensity information for more accurate segmentation performance. Hence, deep learning based methods [KSH12] are more preferable to solve this problem.

However, deep learning methods face a lot of challenges. Firstly, they typically require a large amount of training data. Specifically, in medical image analysis, it is difficult to obtain such a large amount of labeled training data. Secondly, a well-trained model on MRI scans from one center (source domain) can lead to severe performance losses when tested on data from another center (target domain) [YWG*19]. This domain shift problem can be due to differences in the imaging device or the image acquisition process. An intuitive but inefficient way to address this problem is to create new annotations and train a new model for each center. A model that can be trained with annotations from only the source domain, which can be generalised to the target domain and without the need for additional annotations, would greatly reduce the adaptation workload and is thus highly desirable. To that end, recent works have used unsupervised domain adaptation to tackle the problem of misalignment between source and target domains.

In scenarios, when the model is trained only on the source domain, and one wants to achieve good performance on target domains (with no target annotations), domain adaptation is really useful. Recent works in the field of domain adaptation are either carried out through image adaptation [CDCH18] or through feature adaptation [DOC*18]. Image adaptation methods synthesize "source-like" images from target images by pixel-to-pixel transformation, typically realized by synthesizing source-like images from target images in an unsupervised way [ZPIE17]. Since the target images are now more "source-like", the models trained on the source can be directly used for inference. However, since the performance directly depends on the image synthesis quality, that creates a substantial bottleneck. To alleviate the issues caused by high dependence on image generation, recent works have attempted to align the feature generated for the target domains along with the source features. Recent studies use an adversarial training strategy to realize this. [DOC*18]. The paper which we have implemented in this lab is based on Entropy guided Unsupervised Domain Adaptation [ZSL*20]. The authors implement the adversarial training strategy to implement the unsupervised domain adaptation. Specifically, we first train the model on supervised loss with labeled data, which would lead to good performance on the source-like images. For training the segmenter, we have utilized the U-Net [RFB15] architecture. When this model is applied to data from the target domain, it results in severe performance loss. To alleviate the effect, a feature map discriminator is integrated to reduce the domain shift. For adversarial training, the feature discriminator labels on the target domain are flipped which causes the segmentation network to generate domain invariant features.

2. Method and Architecture

2.1. Overview of Method

In the following section, we use notation similar to the original paper [ZSL*20] for convenience, consistency and because it forms a basis for our contributions. For our task, we are given the labeled source data X^s and unlabeled target data X^t. Source images and ground-truth data are denoted as x_s and y_s respectively. Likewise, x_t represents target images with no associated ground truth. For labelling the two domains, as in the original paper, we also use 1 for the source label and 0 for the target label.

The network architecture (as depicted in Figure [1]) consists of three major components: i) A segmentation network S, ii) a Feature Discriminator network D_F and iii) an Entropy Discriminator Network D_E. The segmenter S is trained using the supervised loss on the source domain and adversarial loss on both source and target domains. The adversarial loss is implemented by flipping labels for target domains. The two discriminators learn to discriminate the feature embeddings and the output entropy maps from two different domains. The objective of the segmenter is to generate embeddings that both the discriminators are unable to discriminate. In the following sections, we will explore each of these components in detail.

2.2. Segmenter Network

The segmenter takes an image x as input and outputs a probability map after the last softmax layer : $S(x) = P_x \subset \mathrm{R}^{HxW}$ where H, W represent the height and width of the segmented images. Since our input images are grayscale and since our task is to perform binary segmentation therefore the output probability map can be depicted using a single channel. In case of additional classes, the output probability map will have as many channels as there are classes.

The segmenter takes the source image as input with associated ground truth and produces predicted segmentation masks as output. The segmenter is based on the U-Net [RFB15] network as shown in Figure[2]. The network essentially consisting of two halves, wherein the first half is encoding layer and the second half is decoding layer. The former follows a generic convolutional network. Each block in the encoding half consists of 3x3 kernels followed by Rectified Linear Unit (ReLU) and 3 Resnet Blocks. After each block, the number of channels are doubled. In total there are 3 such blocks in the first half leading to the number channels becoming 8 times. Each block also consists of dropout layers. In the decoding layers, the number of channels are halved at each block. Each block consists of Transposed Convolutions which leads to increase in the size of the feature map back to the original size with the correct padding and pooling size. Finally the network outputs a single channel map with the height and width of the original input. Moreover, the embedding generated after the encoding layer is also returned as output to train the feature discriminator.

For training the segmenter, the weights of the entire U-Net are changed by supervised loss on source domain. While the weights updates for adversarial loss only effects the first half of the U-Net. To calculate this adversarial loss, the target labels for feature discriminator and entropy discrimiator are flipped. This forces the segementor to generate embeddings and entropy maps for targets that are very much similar to source. And hence, this would in turn result in domain invariance. For supervised loss, we have used a combination of the Binary-cross entropy loss and the dice coefficient loss which is defined as follows :

$$L = -\sum y_s^{(h,w,c)}.log(p_s^{(h,w,c)}) - \lambda \sum \frac{2\hat{y_s}^{(h,w,c)}.2y_s^{(h,w,c)}}{\hat{y_s}^{(h,w,c)}.2y_s^{(h,w,c)} + \hat{y_s}^{(h,w,c)}.2y_s^{(h,w,c)}} \tag{1}$$

where $y_s{}^{(h,w,c)}$ and $y_s{}^{(h,w,c)}$ represent the predicted and ground truth segmentation on source domain. For unsupervised adversarial loss , the labels of target are flipped from 0 to 1 and the model tries to fool the discriminator. Also, to measure the domain shift we use the Surface Dice Co-efficient [SDC] which measures the differences between the contours of the prediction and ground truth volumes.

2.3. Feature Map Discriminator Network

The feature map discriminator aims to learn and discriminate the feature maps generated by the segmenter between the source domain and target domain. The network takes the feature maps generated by the first half of the U-Net as input. The source and target scans are passed through the first half of U-Net and the feature maps hence generated are fed into the discriminator network. Unlike the

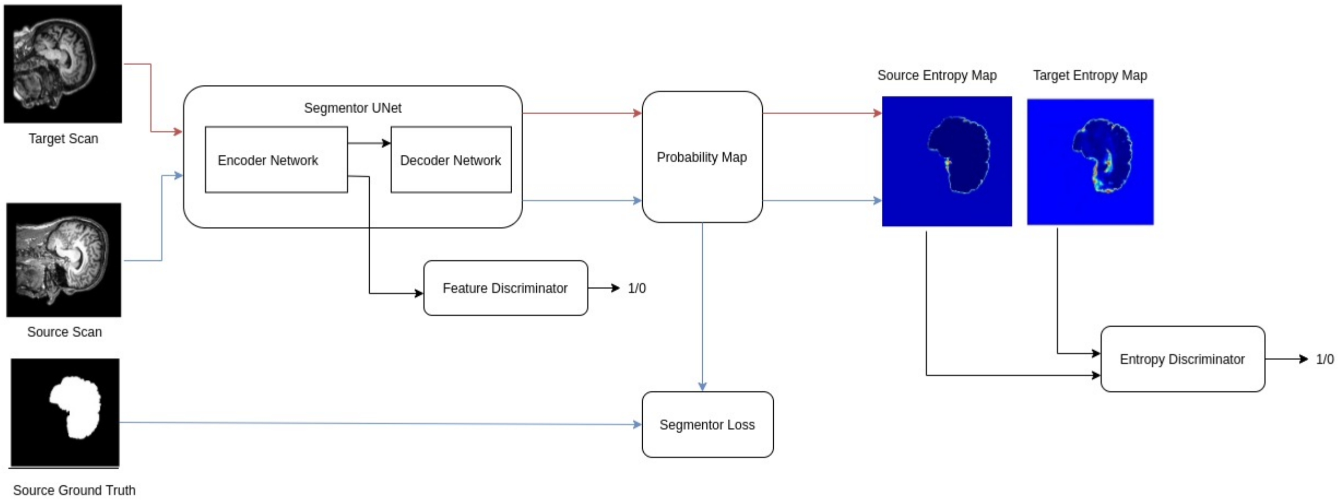

Figure 1: *Network Architecture*

paper, our network consists of 3 convolutional layers instead of 5. This was decided based on the size of the input feature maps to be given to the network. In the original paper, the feature maps are of size 50x50 whereas we have feature maps of size 36x36. This is because we use a different dataset with a different size. Each convolution layer has a stride of 2. Also each of them are followed by rectified linear unit activation layers. Finally, we also use a fully connected layer with 64 neurons mapping to a single output neuron which is followed by a sigmoid layer to yield an output between 0 and 1. Ideally, the network should take feature maps from source and yield 1, and 0 for target feature maps. In order to propel feature alignment between source and target feature using adversarial training, the labels for target scans are flipped from 0 to 1. In simple terms, the discriminator would take target scan feature maps as input but would be forced to produce 1 as output. Thus in principle, the adversarial loss tries to fool the discriminator and forces the segmenter to generate feature maps similar to that of the source. This would in turn ensure that the segementer is domain invariant. Let L_D be the cross-entropy loss. The loss function for D_F is given by :

$$L_{D_F} = \frac{1}{|X_s|}\sum_{X_s} L_D(S_F(x_s),1) + \frac{1}{|X_t|}\sum_{X_t} L_D(S_F(x_t),0) \tag{2}$$

2.4. Entropy Map Discriminator Network

The entropy map discriminator aligns the probability distribution from source and target domains. The architecture of the entropy map discriminator is similar to the feature map discriminator. The only difference is that it contains 5 convolutional layers, which is similar to the architecture used in the original paper. Like the feature map discriminator, we use fully-connected layers at the end of the entropy discriminator as well. However, we use two such layers with 512 neurons and 64 neurons respectively. The output entropy map $E_x \subset \mathrm{R}^{HxWxC}$ is defined based on probability output map as follows :

$$E_x^{(h,w,c)} = -p_x^{(h,w,c)}.log(p_x^{(h,w,c)}) \tag{3}$$

where $p_x^{(h,w,c)}$ represents each voxel in the probability map $P_x \subset \mathrm{R}^{HxWxC}$. The entropy attains minimum when probability is at 0 or at 1. While the entropy is maximum when the probability is 0.5. The model trained on source domain gives low entropy for source-like images but high entropy on target domain scans. Using adversarial training, the model is forced to make similar entropy maps from different domains. To that end, another discriminator D_E is constructed to align entropy distribution. D_E takes the entropy map of both domains as inputs and predicts the classification label (1 for source domain and 0 for target domain) as output.The loss function for D_E is given by :

$$L_{D_E} = \frac{1}{|X_s|}\sum_{X_s} L_D(E_{x_s},1) + \frac{1}{|X_t|}\sum_{X_t} L_D(E_{x_t}),0) \tag{4}$$

In addition to the above equation we also use an additional direct entropy minimization loss term as suggested in [VJB*18]. It is defined as the sum of all the pixel-wise entropies. The purpose of this is to further enforce the model to produce high-confidence (low-entropy) predictions.

$$L_{ent_x} = \sum_{h,w} E_x^{(h,w)} \tag{5}$$

The final loss is defined as follows :

$$\begin{aligned} L_S = \frac{1}{|X_s|}\sum_{X_s} L_{seg}(x_s,y_s) + \frac{1}{|X_t|}\sum_{X_t}((\lambda_2 L_D(E_{x_t}),1)+ \\ (\lambda_3 L_D(S_F(x_t)),1)+(\lambda_4 L_{ent_{x_t}}) \end{aligned} \tag{6}$$

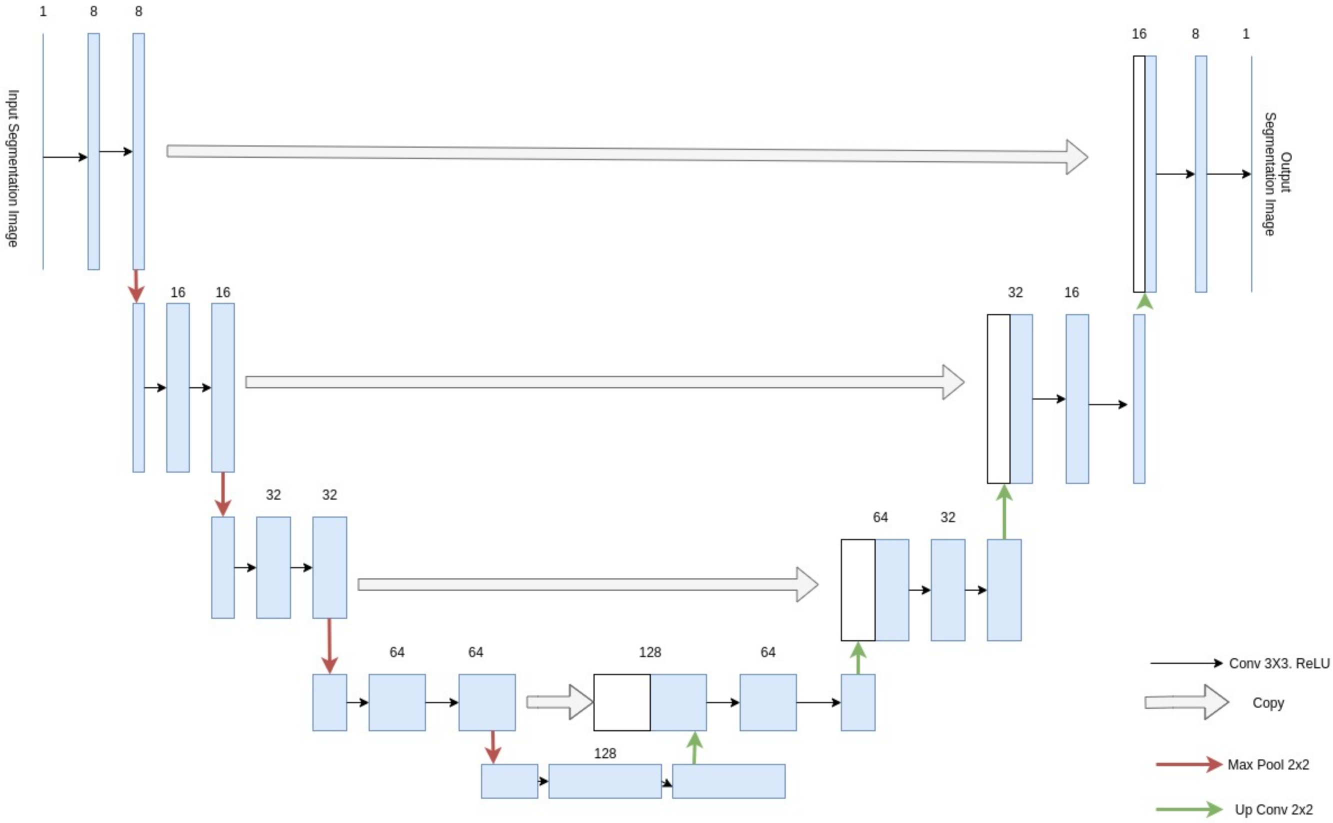

Figure 2: *U-Net Architecture*

3. Dataset and Preprocessing

We have used the Calgary-Campinas-359 (CC-359) Public dataset [SLG*17] consisting of MR Images for the brains of 359 subjects. The MR image volumes are obtained at different magnetic field strengths (1.5T and 3T) from different vendors (GE, Philips, and Siemens). This results in 6 different combinations (or domains). We arbitrarily chose GE-3(GE at 3T) as our source domain. We report all domain shift results on the remaining 5 combinations. On exploring the data we saw that some of the domains have different image sizes. We also observed stark differences in intensity ranges for volumes between domains as well as within the same domain. Therefore our pre-processing consisted of the following steps for each scan volume: a) Clipping outlier intensity values to a range between the 1st and 99th percentile, b) min-max scaling to the range 0 to 1, and lastly c) zero-padding all images to a size of 288 x 288 (maximum across domains). The previously mentioned operations (a and b) were inspired from the pre-processing steps of Shirokikh et. al. [SZC*20]. All the datasets used in our implementations were split into random train (70%), validation(20%), and test(10%) sets based on the number of volumes.

4. Implementation Details

For implementing this work we have used Python (version 3.8) and the Pytorch library. NiBabel library [†] was used for reading the MRI image scans. DeepMind's surface distance library was used for calculating the surface dice metrics [‡]. We use the same implementation of the U-Net segmenter used by Shirokikh et. al [SZC*20] also linked in the footnotes below. Similarly, we also use the deep-pipe library [§]. All our supporting source code is available on our Github [¶]. Experiments were run using Google Colaboratory and on a Desktop environment. Specifically, we used the Pro version for Google Colab since it allows longer GPU availability and higher memory. This allowed us to load the entire data (source and target) into the RAM which allowed faster runtimes and larger batch sizes (up to size 64). For the desktop environment, we used a computer with an Intel i7 processor, 16 GB RAM and Nvidia graphic card

[†] https://nipy.org/nibabel/
[‡] https://github.com/deepmind/surface-distance
[§] https://github.com/neuro-ml/deep_pipe
[¶] https://github.com/sanket-pixel/unsup-segmentation

(GeForce RTX 2060). This had longer runtimes and we had to use smaller batches (of size 4).

5. Experiment Details

Our implementation broadly consisted of five stages or milestones. They are as follows: Stage-I) Initially, we implemented and tested separately the segmenter and discriminator components of our architecture. We also produce baseline segmentation results. Stage-II) We then combined these components and implement the unsupervised domain adaptation (UDA) approach explained in the original paper [ZSL*20]. Stage-III) We modified the adversarial feature discriminator loss and use it for both the source and the target domain (instead of using it only on the target domain as in the original paper). Stage-IV) We also tried to train the model from scratch to see if this produces better results and lastly, in Stage-V) we add a direct Entropy Minimization loss inspired from [VJB*18] to the experiments carried out before.

Similar to the original paper we perform experiments for a few combinations of the individual components of the network. We perform experiments for 3 combinations: 1) Using only the feature discriminator (FeatDisc) 2) Using both feature and entropy discriminator (BothDisc) and 3) Using both discriminators along with the direct entropy minimization (Combined). Further, we use the postfix 'FS' for a network trained from scratch (only used in stage-IV) and 'FT' for a network where the segmenter was only fine-tuned. Lastly, to distinguish the adversarial method used in the original paper (where the adversarial loss, using the feature discriminator, is calculated only on the target domain) we further postfix the method with '(OG)'. All other approaches use the adversarial method specified in Stage-III below. As an instance 'CombinedFT(OG)' denotes a model that was fine-tuned using the adversarial feature and entropy discriminator as in the original paper along with direct entropy minimization.

Stage I- Baselines: For the baseline results without any domain adaptation, we trained the Segmentation network on the arbitrarily chosen source domain GE-3. We then test it on the remaining 5 domains to produce baseline scores. The segmenter model was trained for 15 epochs (450 iterations per epoch, and batch size 16). We used a learning rate of 1 x 10^{-3}. We experimented with different optimizers like Stochastic Gradient Descent (SGD), Adam and AdamW, however, we report all our results only for the Adam optimizer (with default parameters) since these results were the best. At this stage, the discriminator networks were also trained independently to test their capability. The default Adam optimizer with a learning rate of 1 x 10^{-3} was used for both. Generally, both the feature and entropy discriminators required a maximum of 5 epochs to achieve very good discrimination accuracy.

Stage II- Implementing the Original Paper: In this stage, we combined the different components of the network. We use a pre-trained segmentation network as suggested in the original paper [ZSL*20]. Both the discriminator networks were untrained and used to fine-tune the segmenter with the adversarial loss. The task is to fine-tune the pre-trained segmenter such that it produces domain invariant features that can fool both the discriminators. The goal for fine-tuning is to improve segmentation performance on the target domain without significantly affecting performance on the source domain. This requires a lot of experimentation to find the right balance, for learning rates of the different networks, as well as, for the weighting factors for the adversarial loss of the feature discriminator and entropy discriminator terms. Also, the original paper does not specify details for balancing the various components of the network. There are different ways to optimize this network with many hyperparameters. Our goal was to find a general combination of hyperparameters that could be applied for fine-tuning all domains. After a lot of experimentation, we found the following methods. We use a separate optimizer (Adam with default parameters) for the segmenter, feature discriminator, and entropy discriminator. All the components are optimized jointly in every epoch. We found that a learning rate of 1 x 10^{-4} for all components was most suitable for jointly training the network. As noted by Vu et. al. [VJB*18], higher values for the adversarial loss weighting factors prevent the discriminator from learning and produce a bias towards a single domain. We, therefore, use a fixed value of 0.001 for all adversarial weighting factors. However, in cases with a small domain shift, we found 0.0001 to be more suitable. We also experimented with various stopping conditions for the training procedure. The most reliable for fine-tuning was training the model for 5 epochs with a weight decay of 0.75 after each epoch. For cases with a relatively small domain shift, we found training only for 3 epochs was enough. Also, for all experiments, we use a single dataloader such that each training batch can contain both source and target examples. Note that we do not use the target segmentation masks for training. Only the target images are used to train the discriminator networks and for the adversarial loss.

Domain	DICE score	Surface Dice at 1mm
Train	0.99	0.96
Validation	0.99	0.95
Test	0.99	0.95

Table 1: *Segmentation performance metrics (DICE score and Surface Dice score at 1mm Tolerance) on the source domain GE-3*

Domain	DICE score	Surface Dice at 1mm
GE-3 (SRC)	**0.99**	**0.95**
GE-1.5	0.86	0.51
Philips-3	0.87	0.63
Philips-1.5	0.97	0.83
Siemens-3	0.98	0.93
Siemens-1.5	0.95	0.80

Table 2: *Baseline segmentation metrics (DICE score and Surface Dice score at 1mm Tolerance) without any domain adaptation (using GE-3T as the source domain)*

As a side note, we also tried implementing the same segmenter model (inspired from Dilated Residual Networks [DRN]) as described in the original paper. However, we saw some gridding artifacts in the output segmentation. Due to time constraints, we could not explore making modifications to this model.

Stage III- Modifying the adversarial loss: Based on our ex-

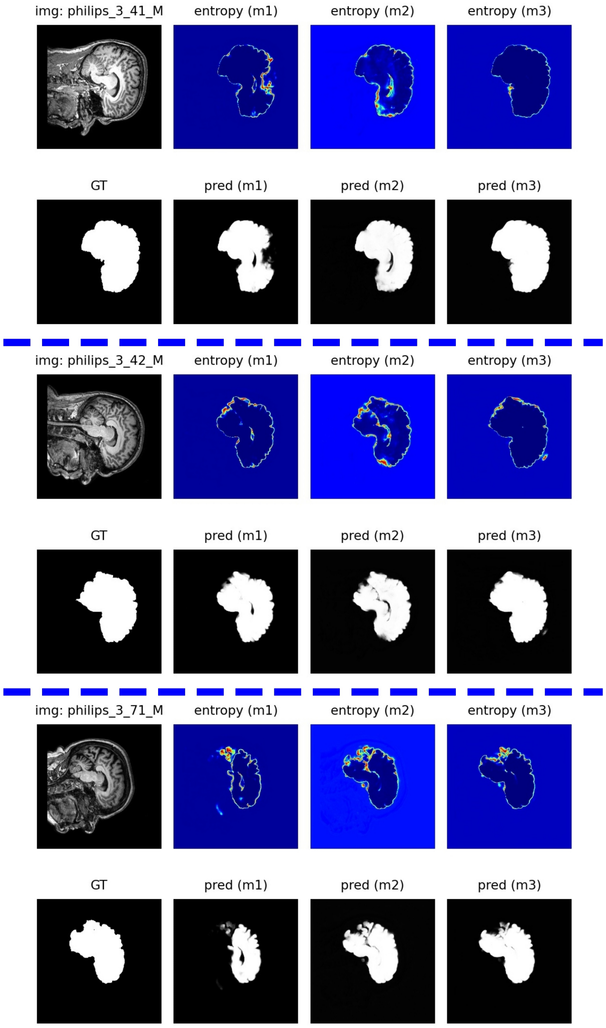

Figure 3: *Plots for results of the models trained from scratch with source domain as GE-3 and target domain as Philips-3. In the above figure model m1 refers to results without domain adaptation. Model m2 refers to results using only a feature discriminator (FeatDiscFS). Model m3 refers to results using both the feature and entropy discriminator (BothDiscFS).*

perience and intuition from implementing the original paper we decided to modify the adversarial loss. In the original paper, the adversarial loss, for the feature discriminator and the entropy discriminator, is calculated only on the target domain. This seems reasonable for the entropy discriminator since our objective for the target domain is to produce low entropy outputs like the source domain. However, the objective of the adversarial loss using the feature discriminator is to enforce the segmenter to produce domain invariant features. Therefore our hypothesis is that the adversarial loss should be calculated for both the source and the target domain. To this end, we modify the adversarial loss and for comparison, we perform experiments with similar settings as in the previous stage.

Stage IV- Training from Scratch: Although the original paper suggests fine-tuning the pre-trained segmentation network, we hypothesize that jointly training the entire network from scratch could be a robust way to produce domain invariant features. Therefore, we repeat the experiments, training all components from scratch. This was done only for experiments from stage III, since the only difference between stage II and III is the adversarial loss of the feature discriminator.

The model with only the feature discriminator (FeatDiscFS) was trained entirely from scratch for 15 epochs and we used a learning rate of 1.5 x 10^{-3} for the Segmentation network and a learning rate of 1 x 10^{-3} for the Discriminator networks. We also use a learning rate scheduler that decays the learning rate by a factor of 0.5 at each step. The scheduler starts decaying only after it achieves a minimum source validation surface dice score of 0.85 and entropy less than 0.15. This was done to ensure that the model at the least performs well on the source domain and produces high confidence predictions. On the other hand for training the model with both

Approach	DICE score	Surface Dice
No domain adaptation	0.80	0.49
FeatDiscFS	0.90	0.61
BothDiscFS	0.95	0.73
FeatDiscFT(OG)	0.91	0.63
BothDiscFT(OG)	0.92	0.64
CombinedFT(OG)	0.94	0.70
FeatDiscFT	0.92	0.64
BothDiscFT	0.94	0.71
CombinedFT	**0.96**	**0.75**

Table 3: *Results for various unsupervised domain adaptation approaches on the unseen test set of the target domain (Philips-3) using a model trained on the source domain (GE-3).*

Approach	Baseline	FeatDisc	BothDisc	Combined
GE-1.5	0.624	0.792	0.793	**0.795**
Philips-1.5	0.855	**0.897**	0.891	0.844
Philips-3	0.496	0.645	0.71	**0.759**
Siemens-1.5	0.808	0.818	0.837	**0.862**
Siemens-3	**0.940**	0.937	0.935	0.937

Table 4: *Average Surface Dice scores on different target domain unseen testing datasets using fine-tuning and trained with GE-3 as source domain. The approaches shown above are FeatDiscFT, BothDiscFT and CombinedFT. The full names are not used due to space constraints.*

discriminators (BothDiscFS) we used a learning rate of 1.5 x 10^{-4} for the Segmentation network and a learning rate of 1 x 10^{-4} for the Discriminator network. We also used a learning rate scheduler that decays the learning rate by a factor of 0.5 at each step (with steps at epochs 15 and 18). The learning rate scheduling method was adapted based on our experience with training the FeatDiscFS. Also, as discussed previously all adversarial loss terms were weighted with a factor of 0.001.

Stage V- Direct Entropy Minimization: Lastly, along with the indirect entropy minimization i.e. the adversarial entropy loss, we examine the effects of adding a direct entropy minimization term to the loss function as suggested by Vu et. al. [VJB*18]. We repeat all previous fine-tuning experiments (Stage II, and III) using this additional loss term. Stage IV was excluded due to time constraints. All the training settings are similar to those specified in stage II.

We produce results for the methods mentioned above for a particular source and target domain pair. Based on these results we then selected a category of models to validate our results on all other target domains as well. The results are discussed in the following section.

6. Results

Below we present results for all the methods mentioned in the previous section.

Baselines As mentioned in section 3 we arbitrarily chose GE-3 as our source domain. We train our Segmentation model on GE-3 using the method described previously and obtain results as seen in Table 1. The baseline U-Net segmenter model performs quite well on the source domain. Also, in Table 2 we compare the performance of this baseline model without any domain adaptation for all the other target domains. We used a random sample of 20 volume scans for each of the target domains. The domain shift is not large when observing only the standard DICE score. However, the Surface Dice at 1mm tolerance shows a large domain shift for GE-1.5 and Philips-3. For our initial experiments, we chose Philips-3 as our target domain.

Comparison of all approaches: The table 3 shows a comparison for all approaches (mentioned previously) on an unseen test set (of 7 volumes). The source domain is GE-3 and the target domain is Philips-3. We use the DICE score and Surface Dice for comparison. It must be noted that the average Surface dice score without any adaptation is lower than the baseline (which was calculated on a random sample of 20 volumes) shown in table 2. All the presented methods show a successful domain adaptation. We however note that the 'CombinedFT' outperforms all others. In general, we see that the models using only the feature discriminator (FeatDisc) are outperformed by the other approaches (BothDisc and Combined) for each of the categories (FS - train from scratch, FT - fine-tuning, and FT(OG) - fine-tuning as in original paper). We also see that the model BothDisc(FS) that was trained from scratch has quite good results. However, fine-tuning approaches perform almost as good or better at a fraction of the training cost. Also, for the fine-tuning models, we note that all models trained using the method specified in Stage III (FeatDiscFT, BothDiscFT, and CombinedFT) outperform their counterparts from Stage II (FeatDiscFT (OG), BothDiscFT (OG), and CombinedFT (OG) respectively). Therefore, we chose these fine-tuning models to validate on all domains. Results seen in table 4 are discussed below.

Comparison on all domains: Based on the baseline scores in table 2 and 4 we see that GE-1.5 and Philips-3 have a high domain shift. Other domains show a relatively lower domain shift. We accordingly perform domain adaptation using methods previously discussed in section 5. The results for GE-1.5 and Siemens-1.5 are similar to what was obtained for Philips-3. However, we observed that it was difficult to obtain an improvement on Philips-1.5 and Siemens-3, where we have a relatively small domain shift. In fact, for Philips-1.5 we see some drop when using the Combined method. Howeever, we see that in general the approaches perform well for a majority of the cases.

We also show the effect of domain adaptation using figures for a few of the approaches. In Fig.3, we see the results for a few randomly selected test dataset slices for the models trained from scratch. We see the differences in quality between the predictions (as well as the corresponding entropy maps) of the model without any adaptation as compared to the the model with adaptation (using only the feature discriminator or both discriminators). Also in Fig. 4 we see results for a few randomly selected test dataset slices for the best performing models from table 4.

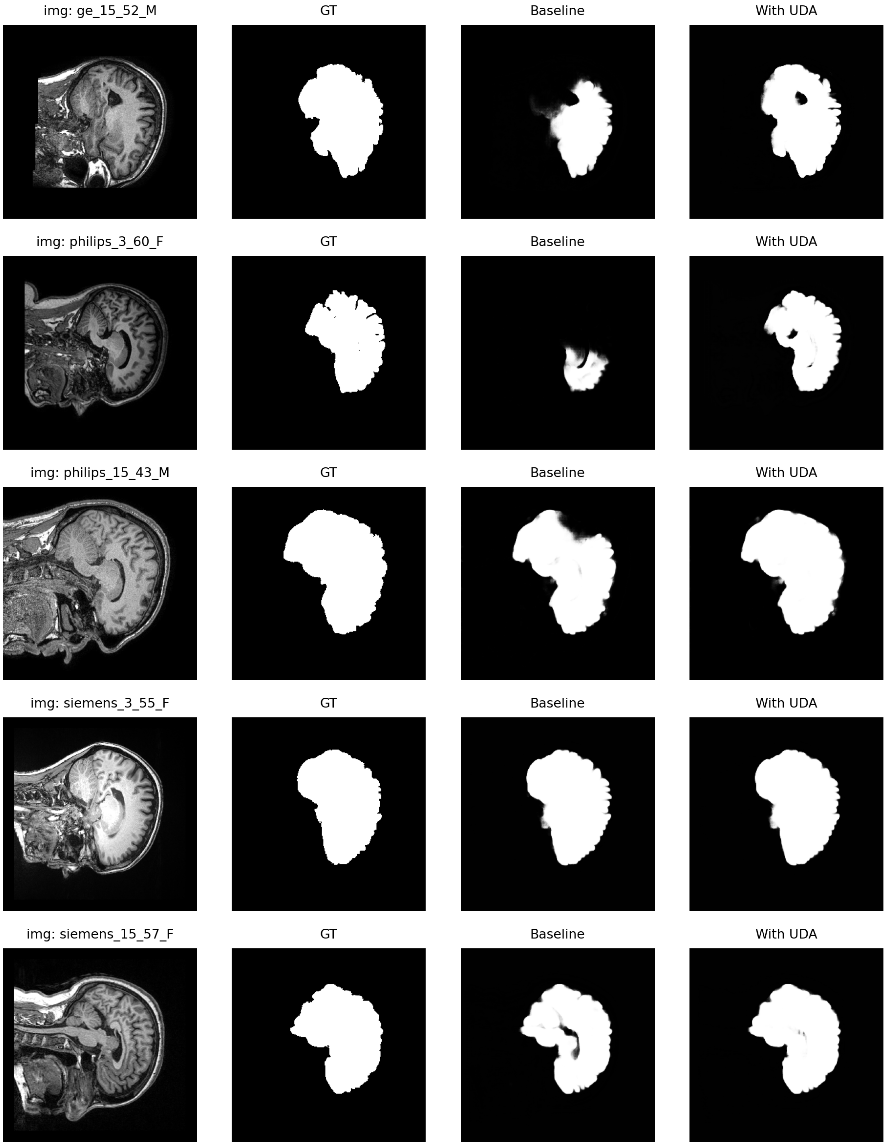

Figure 4: *Plots for results on best performing fine-tuning models from table 4 on all target domains.*

7. Conclusions

In this work we have shown how Unsupervised domain adaptation inspired from [ZSL*20] can be used to successfully reduce the effect of domain shift for a segmentation model trained on a source domain and evaluated on different target domains. Specifically, we show how entropy-based methods (both direct and indirect entropy minimization) can improve domain adaptation, especially in cases where the effect of domain shift is high. Further, we show that the adversarial feature discriminator loss on both domains can perform better than the approach described in the original paper we implemented. Lastly, we show how fine-tuning methods can perform better than jointly training the network from scratch (which is far more time-consuming). In summary, all approaches described in the paper show promising results for unsupervised domain adaptation.

8. Future Work

Based on our experiments with adversarial training, we saw the potential for some variability in the results over different executions with the same parameters. Therefore, we think in the future it is important to obtain confidence measures for the obtained results. Also, as future work, we would like to investigate if the domain shift on this data is significantly different when using models trained with regularization methods or when using deep learning models like transformers. Additionally, methods like histogram matching or Generative Adversarial Network-based style transfer could also be used to augment source domain images to produce images that are closer to the target domain. It would be interesting to see if training on such data can further boost performance. Self-supervised learning approaches based on augmenting the source scans might also be used.

9. Contributions

In the initial phase, the training of the segmenter was done by Rohil and the training for the discriminator was done by Sanket. Both team members worked on combining the components to create the final architecture. Due to different working environments, Sanket implemented a framework for training the network on a personal computer whereas Rohil implemented the same on Google Colab. Therefore, we provide two working code environments that can be used as required. In the later phases the other experiments and approaches mentioned in the report were split up between both team members.

10. Acknowledgements

This lab project is courtesy of Prof. Thomas Schultz and Rasha Sheikh. We thank them for their guidance in the completion of this work.

References

[AGH*17] Akkus Z., Galimzianova A., Hoogi A., Rubin D. L., Erickson B. J.: Deep learning for brain mri segmentation: State of the art and future directions. *Journal of Digital Imaging 30*, 4 (Aug 2017), 449–459. 1

[Bre01] Breiman L.: Random forests. *Machine Learning 45*, 1 (2001), 5–32. 1

[CDCH18] Chen C., Dou Q., Chen H., Heng P.-A.: Semantic-aware generative adversarial nets for unsupervised domain adaptation in chest x-ray segmentation. In *MLMI@MICCAI* (2018). 2

[CR08] Cristianini N., Ricci E.: *Support Vector Machines*. Springer US, Boston, MA, 2008, pp. 928–932. 1

[DOC*18] Dou Q., Ouyang C., Chen C., Chen H., Heng P.: Unsupervised cross-modality domain adaptation of convnets for biomedical image segmentations with adversarial loss. *CoRR abs/1804.10916* (2018). 2

[DRN] 5

[KSH12] Krizhevsky A., Sutskever I., Hinton G. E.: Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25*, Pereira F., Burges C. J. C., Bottou L., Weinberger K. Q., (Eds.). Curran Associates, Inc., 2012, pp. 1097–1105. 1

[Nie18] Nielsen M. A.: Neural networks and deep learning, 2018. 1

[RFB15] Ronneberger O., Fischer P., Brox T.: U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention* (2015), Springer, pp. 234–241. 2

[SDC] 2

[SLG*17] Souza R., Lucena O., Garrafa J., Gobbi D., Saluzzi M., Appenzeller S., Rittner L., Frayne R., Lotufo R.: An open, multi-vendor, multi-field-strength brain mr dataset and analysis of publicly available skull stripping methods agreement. *NeuroImage 170* (08 2017). 4

[SZC*20] Shirokikh B., Zakazov I., Chernyavskiy A., Fedulova I., Belyaev M.: First u-net layers contain more domain specific information than the last ones. In *Domain Adaptation and Representation Transfer, and Distributed and Collaborative Learning - Second MICCAI Workshop, DART 2020, and First MICCAI Workshop, DCL 2020, Held in Conjunction with MICCAI 2020, Lima, Peru, October 4-8, 2020, Proceedings* (2020), Albarqouni S., Bakas S., Kamnitsas K., Cardoso M. J., Landman B. A., Li W., Milletari F., Rieke N., Roth H., Xu D., Xu Z., (Eds.), vol. 12444 of *Lecture Notes in Computer Science*, Springer, pp. 117–126. 4

[VJB*18] Vu T., Jain H., Bucher M., Cord M., Pérez P.: ADVENT: adversarial entropy minimization for domain adaptation in semantic segmentation. *CoRR abs/1811.12833* (2018). 3, 5, 7

[YWG*19] Yan W., Wang Y., Gu S., Huang L., Yan F., Xia L., Tao Q.: The domain shift problem of medical image segmentation and vendor-adaptation by unet-gan. *ArXiv abs/1910.13681* (2019). 1

[ZPIE17] Zhu J.-Y., Park T., Isola P., Efros A. A.: Unpaired image-to-image translation using cycle-consistent adversarial networks. In *2017 IEEE International Conference on Computer Vision (ICCV)* (2017), pp. 2242–2251. 2

[ZSL*20] Zeng G., Schmaranzer F., Lerch T. D., Boschung A., Zheng G., Burger J., Gerber K., Tannast M., Siebenrock K., Kim Y.-J., Novais E. N., Gerber N.: Entropy guided unsupervised domain adaptation for cross-center hip cartilage segmentation from mri. In *Medical Image Computing and Computer Assisted Intervention – MICCAI 2020* (Cham, 2020), Martel A. L., Abolmaesumi P., Stoyanov D., Mateus D., Zuluaga M. A., Zhou S. K., Racoceanu D., Joskowicz L., (Eds.), Springer International Publishing, pp. 447–456. 2, 5, 8