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

Reducing Domain Shift on the Calgary-Campinas Dataset

Morris Klasen
University of Bonn, Department of Computer Science

Abstract

Convolutional Neural Nets performance can be very susceptible to intensity variations of MRI scans. These variations, induced by different MRI-scanner manufacturers and scanning protocols, can result in significant reductions in segmentation quality. Many recently proposed domain adaptation methods attempt to minimize the domain shift by intelligently fine-tuning specific domains to overcome segmentation quality shortcomings. In this context, we evaluate the existing SpotTUNet domain adaptation method and its generalization capacities under stronger regularization by introducing intensity and geometric transformations. Further, we present our own optimized training procedure that reduces the domain shift through simple unsupervised model regularization from 37.8% to 9.3% surface Dice score. We minimize the performance reduction on unseen scanners further to 2.8% using a multi-level intensity and geometry-based augmentation pipeline. And lastly, we propose a multi-orientation ensemble reducing the average domain shift to only 1.5%. We show, that our combination of methods yields an unsupervised domain adaptation system that almost eliminates domain shift and therefore avoids the need for any costly scanner-specific annotations that prior methods proposed to use. The code is available at https://github.com/m-klasen/lab_med_viz.

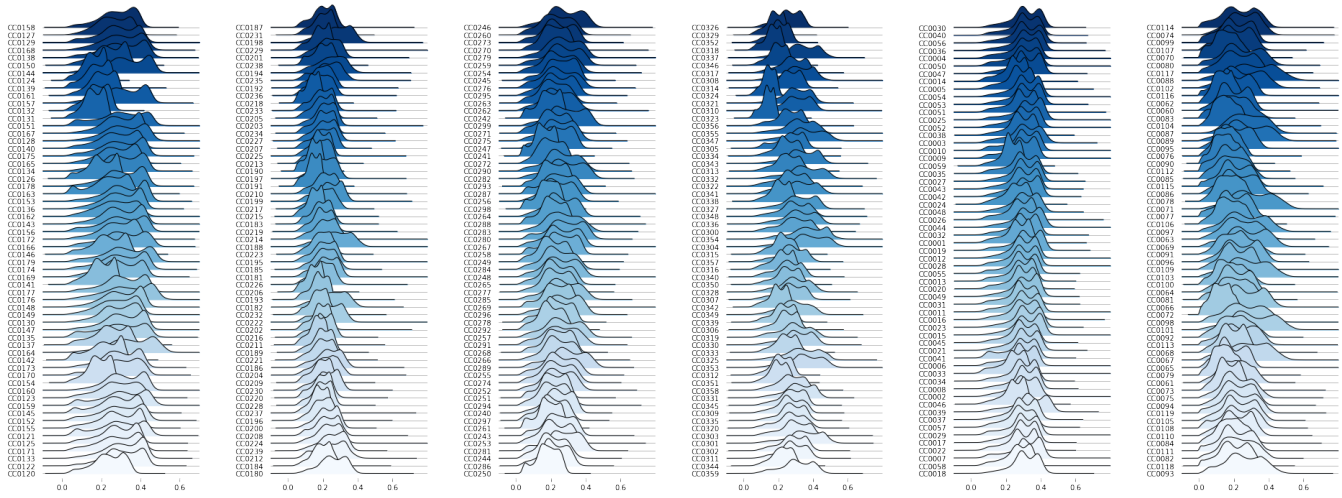
1. Introduction

The advent of Convolutional Neural Networks (CNNs) and their displacement of other algorithms used for medical image analysis has been swift [LKB*17]. CNNs show good generalization capabilities when trained on highly variant datasets but can fall short on unexpected distribution shifts. In this specific case, we are interested in using CNNs on Magnetic Resonance Images (MRI) to extract a binary segmentation mask of brain tissue (*"skull stripping"*). Unlike computer tomography (CT), MRI images yield uncalibrated intensity values, which can lead to distribution differences. This variance in intensity values can stem from a range of factors such as MRI-machine manufacturer, field strengths, device settings but also parameters of acquisition and reconstruction techniques [BRH*18]. MRI scans can also differ in slice orientation and voxel spacing. The intensity distribution shift between MRI scans is the so-called *domain shift*, which can result in substantial performance reduction in predictive power if scanner variations cannot be captured. To tackle the issue, there are several commonly used *domain adaptation* (DA) approaches: The simplest but most costly solution is to sample MRI scans with high-quality expert-level annotations from a maximal number of domains for the training set. Another approach is to adapt existing models, trained on a specific source domain, using new data from another target domain. For this transfer learning or supervised domain adaptation (sDA) approach, we want to minimize the annotation requirements from target domains by intelligently transfer learning [ZSCB21]. Previous work tried to answer the question of which layers should be fine-tuned and in which layers the domain shift manifests itself, concluding that it mostly occurs in the early layers [DOC*18, SZC*20]. SpotTunet [ZSCB21, GSK*19] is a transfer learning approach that intelligently provides adaptive fine-tuning for specific target domains in an attempt to reduce the domain shift.
In this work, we analyze the effects of augmentations on the domain shift and the changes to the SpotTUnet policy. When the originally proposed models showed poor generalization capacities, we developed a new system that is better configured for training binary segmentation tasks. The model we propose acts as an *unsupervised domain adaptation* approach that minimizes the average *domain shift* through relatively simple but powerful training pipeline optimizations yielding a more generalized model manifold. We will use the publicly available CC359 dataset [SLG*18] with 259 patient scans from 6 different MRI scanner types.
We want to highlight the following contributions:

- We evaluated the impact of augmentations on the SpotTUNet and the fine-tune policy.
- The introduction of a multi-level augmentation approach, capturing possible domain shifts during training and substantially improved prediction quality on target domains.
- We upgraded the training pipeline yielding significantly higher generalization capability and increased performance on target domains.
- The development of a multi-orientation ensemble to partially reintroduce 3D neighborhood information, increasing the prediction quality.

2. Related Work

The Deep visual domain adaptation survey [WD18] highlights three possible approaches to tackle this heterogeneous domain adaptation problem: (1) Supervised, (2) semi-supervised and (3)



(a) Siemens 1.5T (b) Siemens 3T (c) GE 1.5T (d) GE 3T (e) Philips 1.5T (f) Philips 3T

Figure 1: The brain tissue distribution plots of individual MRI scans for each source domain. The raw intensities were normalized. We can see the typical white-matter, grey-matter and cerebrospinal fluid peaks in the distributions. While there are noticeable intra-domain variances, there is not obvious inter-domain pattern in the overall distributions.

unsupervised settings. Much of the previous work in the area of medical image analysis focuses on the supervised configuration. [KVGV*19] conclude that very few samples suffice to adequately transfer to specific target domains while fine-tuning the last layers of the network. [VLB*18] similarly only fine-tuned the last layer for domain adaptation, seemingly indicating domain shift in higher-level features. However, [AT16] argues that MRI images, in general, tend to display lower level domain shift and concludes initial network layers should be targeted. To this extent [ZSCB21] analysed the anatomy of domain shift, confirming that early layer fine-tuning is better to address *domain adaptation*.

A method that uses *semi-supervised domain adaptation* is [MMKSM18], who propose using a Conditional Generative Adversarial Network (c-GAN) for chest X-Ray image classification that allows for the incorporation of additional unlabeled samples into the training process.

When it comes to fully *unsupervised domain adaptation*, they will generally follow one of two goals: *Image-alignment* and/or *Feature-alignment*. Image-alignment of samples from differing domains mostly use GAN's [KBL*17, YDZ*19, DOC*18]. [WER18] propose using a cycle-GAN as a DA method for breast-cancer classification, transforming images from one medical centre domain into that of another medical centre domain followed by a CNN for classification. Most *feature-alignment* methods aim to learn only those image features that are invariant to the domain. The most prominent concept is using a siamese network like a Domain Adversarial Neural Network (DANN) [GUA*16]. [KBL*16] propose such a system for brain lesion segmentation in which they train their segmentation network in parallel to a discriminator connected to multiple layers of the segmentation network.

3. Data

All experiments in this work are done on the CC359 [SLG*18] dataset. The dataset contains 359 MRI scans of patients heads from three different scanner manufacturers: Siemens, Philips and General Electric. Besides the differences in acquisition protocols by differing vendors, they also acquire the scans using varying field strength, further diving them into 1.5 Tesla and 3 Tesla groups. This yields a total of 6 different domains which are equally represented in the number of scans (except Philips1.5T with only 59 scans). For annotations, pixel-wise binary masks are provided for every scan. The scans feature various scales of T1-weighted raw intensities, voxel resolutions and sizes in general. Figure

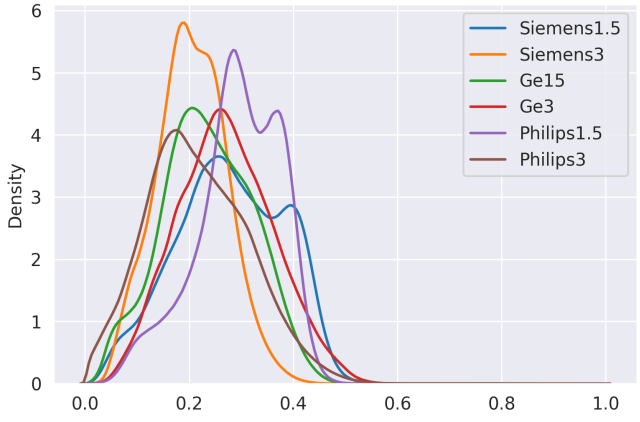


Figure 2: Plot of the average brain matter intensity distribution of every scan. For each of the six domains, we can see the characteristic CSF, white-matter and grey-matter distribution peaks.

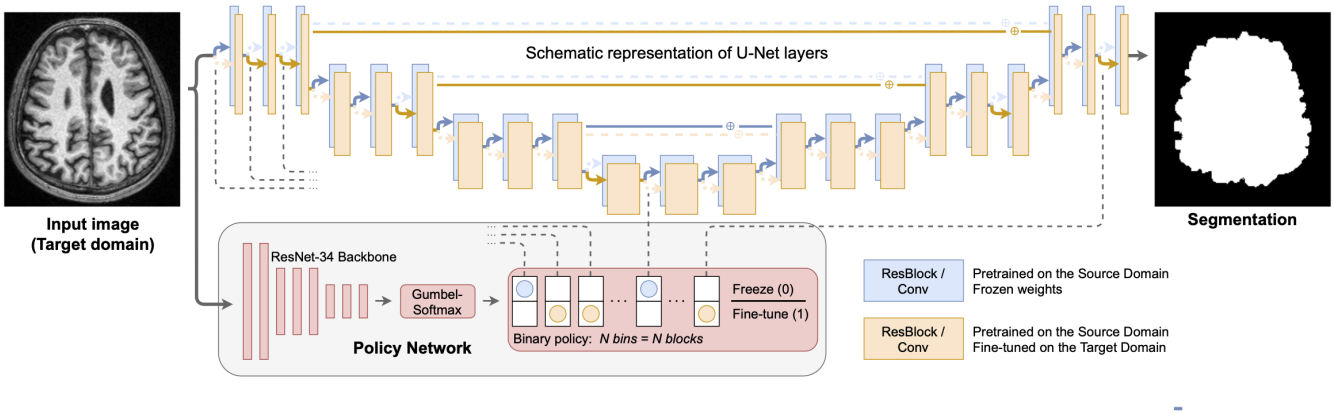


Figure 3: The figure and architecture by [ZSCB21] describe the model architecture for their supervised domain adaptation approach. The blue U-Net weights are frozen, while the orange copy can be fine-tuned. The fine-tuning is specific layers is decided by an policy network as proposed in [GSK*19]. The policy network decides for each U-Net layer if the blue frozen weights are kept or the orange fine-tunable weights are to be used.

2 shows the intensity distributions of the brain matter for each domain. We observe that they all follow the general homogeneity of brain scans as indicated by the density estimations being approximately multimodal distributions with three modes. We know they are the three predominant brain matter types: white matter, grey matter and cerebrospinal fluid. The figure also highlights the differences between scanners, where we see varying density means and deviations causing inconsistent, underlying matter-specific intensities. That can result in a substantial overlap of different tissue types with similar intensities. The white matter peaks are also pronounced differently between each scanner-type, as we barely see any peak for the general electric 1.5T and 3T models, while the Siemens 1.5T and Philips 1.5T white matter peak is unmistakable. Interestingly, the overall domain distributions show no definitive and consistent pattern between scanner manufacturers or scanner field strengths.

Going into more detail with Figure 1, we plot the individual MRI-scan distributions of each domain. Immediately we can see that, within domains, there are large variations between patient scans. Some scans have widely varying distribution widths and peak densities at different intensities.

The challenge, in this, could then be characterized as the classic variance-bias trade-off problem. Training a model that captures the variance on the domains it is being trained on, while simultaneously having to avoid overshooting in the other direction of bias and overfitting the model, causing poor performance on other domains.

To normalize the various input format as inputs for neural network training, we initially re-scale the voxel spacing of every scan to $1 \times 0.95 \times 0.95$ using 3rd order spline interpolation as done in [ZSCB21]. Following that, we transform the input shapes to $(256, 256, 256)$ by padding and cropping the full scans using the MONAI medical imaging framework [C*20].

4. Methods

4.1. Architecture

2D U-Net. For qualitative comparisons, we use the same 2D U-Net proposed by Zakazov et al. [ZSCB21]. The 2D U-Net consists of 29 layers and three skip connections. In the layers, we convolve and downsample in 3 blocks from an initial resolution of 256 to 32 and vice versa. The most significant deviations from the original U-Net implementation is changing the simple convolution block to residual blocks [HZRS16] and using convolutions as skip-connections with a $1 \times 1 \times 1$ kernel. The skip connections within the residual block and the change of the U-Nets skip-connections themselves aim at improving feature encoding and decoding with respect to fine details that were often lacking in the final binary segmentation of U-Nets.

SpotTUnet. Zakazov et al. [ZSCB21] proposed the SpotTUnet. In the case that we have additional data from target domains, we want to optimally fine-tune the existing model trained on a specific source domain. As Fig. 3 details, an additional ResNet-34 is employed, extracting input features which get reduced to a 2×32 dimensional output. Each of the 64 output logits is passed through a Gumbel-Softmax [JGP16] and then corresponds to a U-Net layers probability to be chosen for fine-tuning or remaining frozen, exactly reproducing [GSK*19]. The U-Nets layers, which are frozen by default, are then unfrozen once a ResNet-34 output probability is above 0.5. The original SpotTune authors also proposed a regularization constraint for the global policy aimed at consolidating the global fine-tuning policy to fewer layers, reducing the memory requirements. The SpotTUNet loss with added penalty term is defined as

$$\mathcal{L} = \mathcal{L}_{segm} + \lambda \sum_{l=1}^{6} 4(1 - I_l(x)), \tag{1}$$

where $I_l(x)$ is the binary indicator for the l-th frozen layer based on the image input x. The penalty parameter λ is being optimized via grid-search using a subset of the source/target domains. Since the we want to require as few samples as possible, [ZSCB21] only

use a fraction of the target domain slices for fine-tuning: We evaluate SpotTune performance for 5, 48 and 800 target domain slices available for fine-tuning.

4.2. Augmentations

We want to introduce transformation of the slices during training to regularize the optimization processes more strongly and improve the generalization. As a baseline we chose the well established *RandAugment* method [CZSL19] that combines a multitude of image-level intensity and spatial transformations. We also test several more types of augmentation that were not included in *RandAugment*. We adjusted the original implementation and used the intensity and geometric transformations shown in Table 1.

The detailed functionality of the method is as follows: *RandAugment* calls the underlying function for each sample in a batch-iteration individually. The function-call gives each of the specified augmentations an application probability of 50%. A predefined number of augmentations are, consecutively, chosen at random. The number of randomly chosen augmentations is by default 2. As a consequence, each sample is subject to the chance of having (1) no augmentation applied, (2) one intensity or spatial transformation applied, (3) two intensity/spatial transformation applied or (4) one intensity and one spatial transformation applied simultaneously. To further diversify the augmentations, the parameter of individual augmentation functions are drawn from a Gaussian distribution for every sample. The filter quality (bilinear, bicubic, lanczos, ...) associated with some of the geometric augmentations, e.g. rotations, are chosen at random as well. Figure 4 visualizes the augmentations and their parameters impact on an MRI slice.

Intensity Based Augs.		**Spatial Augs.**	
Augmentation	Params.	Augmentation	Params.
AutoContrast	-	Rotate	(-30,30)
Posterize	(0,4)	ShearX	(-0.3,0.3)
Solarize	56	ShearY	(-0.3,0.3)
Contrast	(0.1, 1.9)	TranslateXRel	(-0.2,0.2)
Color	(0.1, 1.9)	TranslateYRel	(-0.2,0.2)
Brightness	(0.1, 1.9)		
Sharpness	(0.1, 1.9)		

Table 1: List of Augmentations used in our customized *RandAugment* augmentation strategy.

4.3. Metrics

To evaluate the prediction quality of models, we use two metrics: The volumetric dice coefficient score defined as

$$DSC = \frac{2|X \cap Y|}{|X| + |Y|}, \tag{2}$$

with X and Y being two sets of masks (i.e. prediction and ground truth), which measure twice the intersection of prediction and ground truth normalized by the combined total cardinality of both masks that will result in a score between 0 and 1.

However, as [NBZ*18] describes, the volumetric dice score has issues in the case of medical image segmentation as it penalizes all brain-mask edge delineation equally and independently of their distance to the actual edge. [NBZ*18] propose a new surface dice score, that instead of measuring the overlap of the whole volume, first calculates its surface and measures the surface overlap instead. We derive surfaces $\mathcal{S}_X$ and $\mathcal{S}_Y$ from their volumes X and Y with points $\sigma_X \in \mathcal{S}_X$ on the surface. One can define a border region $\mathcal{B}_i^\tau \subset \mathbb{R}^3$ for the surface $\mathcal{S}_i$ with tolerance τ as

$$\mathcal{B}_i^\tau = \left\{ x \in \mathbb{R}^3 | \exists \sigma_i \in \mathcal{S}_i, \|x - \sigma_i\| \leq \tau \right\}, \tag{3}$$

such that we can write the surface dice score at tolerance τ as

$$SDSC = \frac{|\mathcal{S}_X \cap \mathcal{B}_Y^\tau| + |\mathcal{S}_Y \cap \mathcal{B}_X^\tau|}{|\mathcal{S}_X| + |\mathcal{S}_Y|}. \tag{4}$$

4.4. Losses

The original implementation by Zakazov et al. [ZSCB21] uses a regular binary cross-entropy (BCE) loss to maximize the likelihood distribution. The BCE loss works by calculating the mean over the whole volumes. Given a predicted segmentation $\mathbf{x}_i$ and corresponding ground truth $\mathbf{y}_i$ the BCE of a batch is defined by:

$$BCE(\mathbf{X}, \mathbf{Y}) = -\frac{1}{N} \sum_i^N \left[\mathbf{y}_i \cdot \log(\mathbf{x}_i) + (1 - \mathbf{y}_i) \cdot \log(1 - \mathbf{x}_i) \right]. \tag{5}$$

While the BCE loss generally works well for segmentation problems, we face a similar domain-specific issue as we did with the metrics. The provided MRI scans are generally around 85% background pixels containing mostly air and a small degree of other matter like bone structures. Only 15% is actual brain matter denoted with positive annotations. While this imbalance can be compensated with a weighted BCE loss, the Focal Loss has been shown to address the imbalance of classes a lot better [LGG*17]. The focal loss adds a factor of $(1 - \mathbf{x}_i)^\gamma$ to the standard cross-entropy, where the choice of γ is controlling the relative loss for well-classified pixels. Gamma values larger than zero reduce the influence of well-classified pixels (i.e. predictions with probabilities 0.6-1.), focusing on the harder to classify pixels. The focal loss for a binary classification problem defined as:

$$FL(\mathbf{X}, \mathbf{Y}) = -\frac{1}{N} \sum_i^N \left[\alpha \mathbf{y}_i (1 - \mathbf{x}_i)^\gamma \log(\mathbf{x}_i) + (1 - \mathbf{y}_i) \mathbf{x}_i^\gamma \cdot \log(1 - \mathbf{x}_i) \right], \tag{6}$$

with γ being the mentioned *focusing parameter* and α governing the precision-recall trade-off by weighing the positive classification samples.

One additional step we want to take is introduce a IoU-based loss. So far both the BCE and the FL are only pixel-wise losses but do not consider the overall volume overlaps. While the dice loss or jaccard loss were typically used criterions, since the proposal of the Lovasz loss it has gained substantial popularity. Berman et al. [BTB18] showed that the Jaccard loss is submodular and we therefor can optimize it in a continuous optimization framework by using the smooth Lovasz extension of the discrete loss. The Lovász loss is defined by

$$LovaszLoss(\mathbf{F}) = \overline{\triangle_{J_1}}(\mathbf{m}(\mathbf{F})), \tag{7}$$

where $\mathbf{F}$ are the Jaccard output scores, $\mathbf{m}$ are the hinge loss error vectors associated with the individual predictions and ground

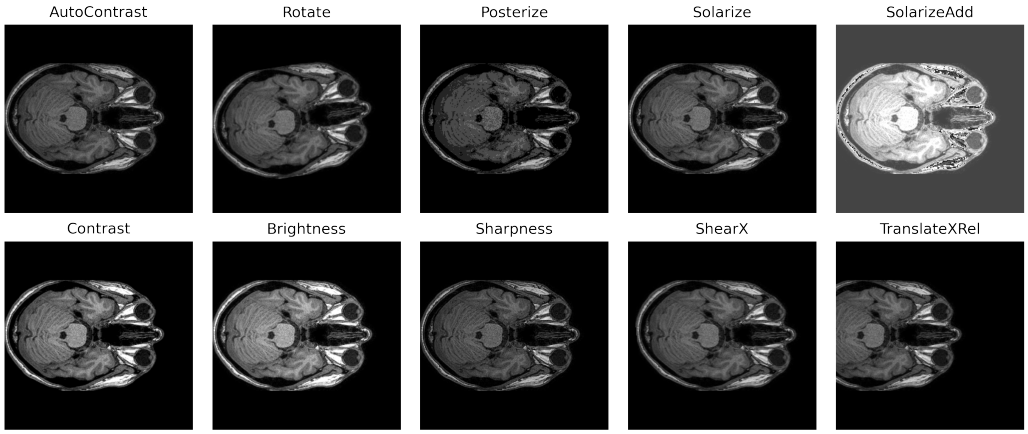


Figure 4: Visualization of a single MRI-scan slice with each individual augmentation contained in the *RandAugment* method applied to it.

truths and $\overline{\triangle_J}$ denotes the submodular Jaccard loss that is tightly convex closed and computable in polynomial time. For specific details visit [BTB18]. In our own implementation we use the combination of FocalLoss and LovaszLoss as a robust criterion: $\mathcal{L}_{tot} = \mathcal{L}_{Focal}(\mathbf{X}, \mathbf{Y}) + \mathcal{L}_{Lovasz}(\mathbf{X}, \mathbf{Y})$.

4.5. Multi View Ensemble

Finally, we want to address the compromise we made by breaking up the full 3D MRI scans into individual 2D slices. The reason for this was a multitude of issues concerning increased model complexity of 3D CNNs, GPU memory constraints and general model convergence due to reduced numbers of total samples. Using a slice-wise 2D U-Net approach, however, removes a lot of context information contained in neighbouring slices and the model only sees a fraction of the overall 3D brain structure at a time.
We propose a new multi-orientation ensemble method to address the issue (Fig. 5). As training is done on a specific but fixed volume orientation, we only ever get neighbourhood information along one axis. If we, however, train one model on each axis orientation, we can combine the models by registering them to the same axis orientation and averaging the prediction logits. By ensembling the three-axis orientation models we reintroduce a weak approximation of the full 3D neighbourhood for each pixel in the volume.

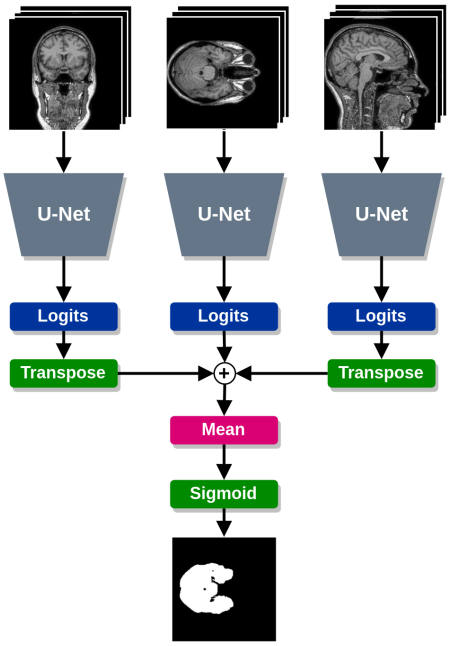


Figure 5: Workflow of the multi-orientation ensemble. We use 3 different 2D U-Net models, trained on different volume orientation slices. The inferred scan logits are re-transposed into equal orientations and ensembled.

5. Experiments

This section can broadly be divided into the two segments: Our initial experiments using the original implementation by [ZSCB21], testing the baseline, benefits of augmentations and their effect on the resulting SpotTUNet policy. And then, after that, we describe the details of our own implementation and present the evaluation results of the baseline, augmentations and SpotTune performance.

5.1. Experimental Setup

Baseline training. In this section, we first compare the different training configurations of the baseline models. In this setting, models are trained on one source domain and evaluated on the five corresponding target domains. The training, validation and test split of the dataset are then defined as follows: The baseline experiments

train one model for each source domain. As each SD contains 60 scans, four complete scans are for validation, and the remaining 56 make up the training set. The 300 remaining scans of other domains will be in the test set to evaluate the target domain performance. Given that we have six domains, we train six models to evaluate all source domain to target domain combination. Additionally, to check for individual domains' upper bound, we train the *oracle*, using 3-fold cross-validation on each of the domains, resulting in a total of 18 models to be trained.

SpotTUNet training. The six domains result in a total of 30 individual source-target combinations, [ZSCB21] propose the separation of Siemens1.5T source domain for optimizing regularization parameters, which leaves 25 remaining to evaluate the DA performance.

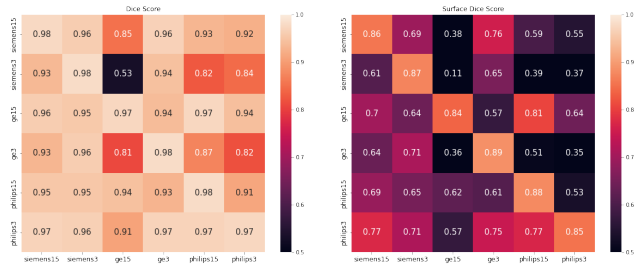
5.2. Original Implementation

5.2.1. Default Original Implementation

The baseline procedure and its configuration as proposed by Zakazov et al. [ZSCB21] is detailed in Table 3. It uses stochastic gradient descent (SGD) with Nesterov momentum at $m = 0.9$ with no weight decay. The model is trained for 40 epochs, with a learning rate of $1e-3$ and a reduction step at epoch 30 to $1e-4$. The data input pre-processing is defined by two steps: (1) Scaling of the raw intensity values to an $[0, 1]$ interval and (2) clipping the normalized intensities outside of the 1st and 99th percentile to values at the percentiles. The baseline does not use any augmentations.

Fig. 6 shows the combined evaluation w.r.t. their dice and surface-dice scores. The heatmaps main diagonal represents the *oracle* where we report the mean performance of the 3-fold cross-validation within each source domain. The remaining entries report the source to target domain performance where each row represents the source and each column one target domain.

In Figure 6a we see an average target domain dice score of 88.17% with individual outliers like the Siemens1.5 to Ge1.5 domain of only 61% but also the Ge15 to Philips15 domain with 97%. The average *oracle* dice score is 97.69%. However, looking at the stricter surface dice in Fig. 6b we obtain a substantially lower scores with an average TD score of 59.02% SDice and 86.35% oracle SDice score, but the individual outlier domains in performance remained the same.

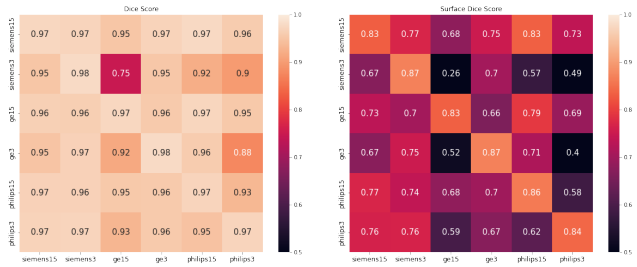


(a) Dice Scores (b) SDice Scores

Figure 6: Oracle and target domain performance on dice and SDice metrics using the original baseline.

5.2.2. Original Implementation with augmentations

For the baseline with augmentations we use the same hyperparameter configuration as seen in Table 3. For augmentations we use *RandAugment* strategy described in Sec. 4.2. Overall, this results in models with an average dice score of 94.35% for the target domains and 97.37% for the *oracle*. The augmentations method increases the surface Dice score on the target domains from 59.02% to 66.55%. However, the *oracle* performance did not increase but stayed on a similar level: 86.35% SDice score compared to now 85.04%. That indicates that we have some degree of overfitting with the *oracle* which were not mitigated by augmentations. However, we did succeed in reducing the domain shift by using image augmentations as regularization.



(a) Dice Scores (b) SDice Scores

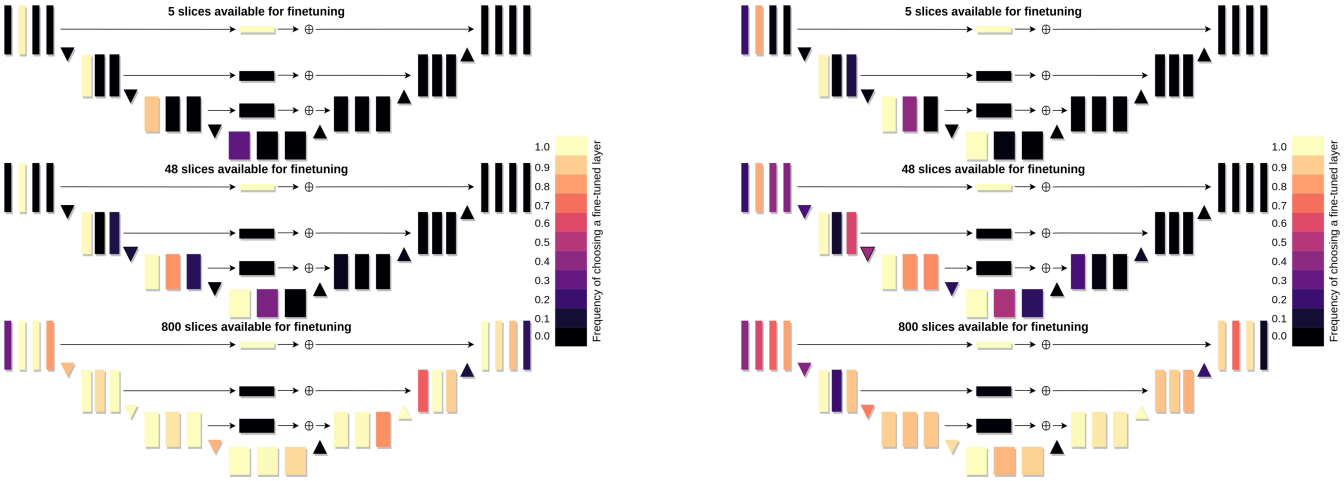
Figure 7: Oracle and target domain performance on dice and SDice metrics using the original baseline with augmentations.

5.2.3. SpotTune of the Original Implementation

We then SpotTune the target domains based on the baseline model we described in Section 5.2.1. We specifically retained the baseline configuration parameters. The policy network is trained with a learning rate of $1e-2$ for 60 epochs. The SpotTune parameters were $\lambda = 0.05$ and the gumbel-softmax temperature was $\tau = 0.1$. We SpotTune using an existing model trained on a source domain and fine-tune it on a specific target by providing some annotated data for training. Each of the 30 possible source/target domain combinations can individually be fine-tuned by training the SpotTune policy.

We analyze the SpotTune performance for three levels of annotated target domain slices being provided for fine-tuning the existing source domain model on each target domain: 5, 48 and 800. Table 2 shows the performance increase in SDice score on one exemplary source domain Siemens3. Each of the target domain columns is then a uniquely trained domain-specific model. On the baseline, we can observe that the average target domain performance of 42.66% SDice score substantially improves to 74.36% by merely providing five target domain slices. Adding further slices increases the performance by +35.64% for 48 slices and +35.76% for 800 slices, indicating an upper limit reached with around 48 target domain slices. Looking at the corresponding policy visualizations in Fig. 8a we see the same results as in [ZSCB21], the MRI are overall semantically homogeneous, which does not require tuning of later layers, but early layers need to adapt to the domain shift.

The augmented model described in section 5.2.2, however, perform



(a) SGD Base SpotTUNet

(b) SGD RandAugment SpotTUNet

Figure 8: Visualization of the SpotTUnet learned policy. We visualize the default configuration (left) against the addition of RandAugment (right) for 3 different settings: SpotTune with 5,48 or 800 available target domain slices.

slightly different when SpotTUned. While the initial RandAugment SDice performance of 53.88% is +11.22% higher than the default (see Tab. 2), the performance increase with additional target domain slices reduces to +19.94% for 5 slices and +23.90% for 800 slices, which put the target domain averages within one percent of the initial SpotTune results without any augmentations. Visualization of the SpotTUNet policy in Figure 8b shows the fine-tuned layer choice more distributed than the baseline. It indicates that, by augmenting, the domain shift is no longer located in specific early layers and tends to be more distributed across them.
Overall, when looking at the original implementation, we can say that the poor baseline DA performance can be improved by augmentations, which can be helpful if no target domain annotations are available. But the improved model generalization diminishes when training domain-specific SpotTUNet models as their average upper bound performances converge towards very similar levels.

	TD slices	Sms1.5	GE1.5	GE3	Phi1.5	Phi3	Avg.	SDice-Δ
Siemens3 SDice - Original Baseline & RandAugment SpotTune								
Default SpotTune	-	0.610	0.107	0.655	0.389	0.372	0.4266	-
	5	0.766	0.734	0.824	0.734	0.660	0.7436	+0.3170
	48	0.798	0.739	0.840	0.799	0.739	0.7830	+0.3564
	800	0.797	0.713	0.847	0.840	0.724	0.7842	+0.3576
RandAugment SpotTune	-	0.670	0.256	0.703	0.573	0.492	0.5388	-
	5	0.755	0.737	0.753	0.796	0.650	0.7382	+0.1994
	48	0.793	0.762	0.807	0.799	0.728	0.7778	+0.2390
	800	0.797	0.755	0.811	0.821	0.705	0.7778	+0.2390

Table 2: Evaluating the SpotTune performance from the Siemens3 source domain to 5 target domains on the original baseline and RandAugment version. We evaluate the performance for 3 different amount of target domain slices available for fine-tuning.

5.3. Upgraded Framework

5.3.1. Implementation

In this section, we introduce and describe our re-implementation of the training pipeline with several upgrades and configuration changes (see Tab. 3). Since we train on 2D slices and not the full volumes, on-demand slice data-loaders can quickly saturate the input/output instructions per-second capacities of SSD hard drives. To avoid the bottleneck of disk read speeds, we cache the data in RAM using a multi-processing shared array for optimized multi-CPU data-loader accessing in $float32$ as individual slices (train-set) and whole volumes (validation/test-set). We also pre-compute the voxel spacing normalization to $(1, 0.95, 0.95)$ using third-order interpolation and save it to disk, avoiding costly interpolations upon loading MRI volumes.
For the improved pipeline, we switched away from the SGD optimizer and towards an adaptive gradient-based Adam optimizer as experiments showed significantly improved target domain generalization capabilities. Besides changing the optimizer, we optimized the configuration and hyper-parameter choices in the code. We switched the loss function from a log-likelihood maximization (binary-cross entropy) to a combination of Focal loss (with $\alpha = 10., \gamma = 2.$) and Lovasz loss. The learning rate schedule is defined as follows: Of the 40 total epochs, we have five warmup epochs with a learning rate of $1e-5$ followed by a *cosine decay* scheduler starting at $1e-4$. The *weight decay* is set to $1e-2$ and we clip gradients to 1.0 ensuring numerical stability. We trained with a batch size of 64. Using a Nvidia RTX 3090, the code by [ZSCB21] take about 1 hour 50 minutes for each source domain, our optimized implementation trained each domain in circa 1 hour 10 minutes.
Opposed to the prior data-preprocessing pipeline, we diverted from input scaling and outlier-clipping to initial scaling followed by normalization using the datasets global calculated mean and standard deviation values. The clipping could potentially skew the

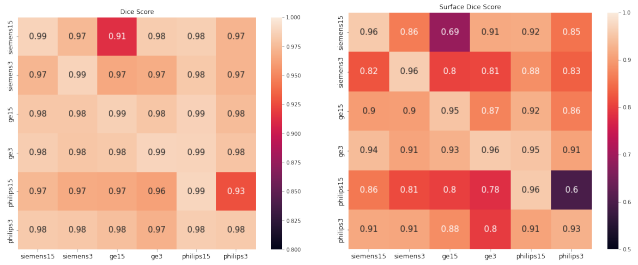
Table 3: Configurations and hyper-parameters used for training. We compare the existing baseline with our own implementations.

	Zakazov et al. approach [ZSCB21]		Ours		
Procedure →	Base	Aug Base	A1	A2	A3
Resolution	256	256	256	256	256
Voxel Spacing	$1 \times .95 \times .95$	$1 \times .95 \times .95$	$1 \times .95 \times .95$	$1 \times .95 \times .95$	$1 \times .95 \times .95$
Percentile Clip.	(1,99)	(1,99)	✗	✗	✗
Normalization	✗	✗	✓	✓	✓
Epochs	100	100	40	40	40
Batch Size	16	16	64	64	64
Optimizer	SGD	SGD	Adam	Adam	Adam
LR	1e-3	1e-3	1e-4	1e-4	1e-4
LR decay	step	step	cosine	cosine	cosine
decay rate	0.1	0.1	-	-	-
Weight decay	-	-	1e-2	1e-2	1e-2
Warmup Epochs	✗	✗	5	5	
Gradient Clip.	✗	✗	1.0	1.0	1.0
Rand Augment	✗	✓	✗	✓	✓
Orient. Ensemble	✗	✗	✗	✗	✓
BCE Loss	✓	✓	✗	✗	✗
Focal Lovasz Loss	✗	✗	✓	✓	✓

intensity distribution, and it is generally good practice to normalize inputs for numerical stability.

5.3.2. Procedure A1: New Baseline with Adam

The changes in the training pipeline and hyper-parameter configuration are summarized in Tab. 3 and resulted in significantly better domain transfer across the board on both dice and SDice metrics (see Fig. 9). The *oracle* under new configuration achieves around 99% dice and 95% SDice score, which is also significantly higher (+8% avg. SDice) than the *oracle* baseline trained with SGD described in Section 5.2.1. Figure 9a shows average Dice scores of 98.63% for the oracle, but also a target domain average of 97.29%, indicating that we already significantly reduced the domain shift. Looking at the surface Dice score in Fig. 9b we see a similarly strong *oracle* with a 95.05% average, but a ten percentage point drop when considering the mean SDice of 85.74% across all target domains.



(a) Dice Scores (b) SDice Scores

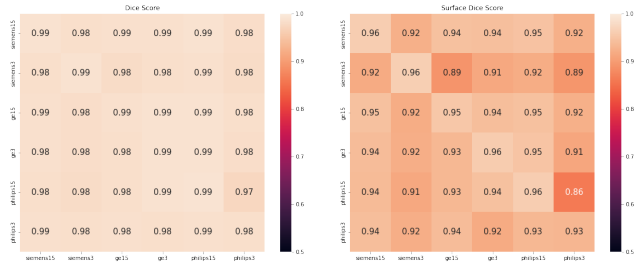
Figure 9: Oracle and target domain performance on Dice and SDice metrics using the A1 procedure without augmentations.

5.3.3. Procedure A2: New Baseline with Augmentations

Building upon our substantial improvements in reducing domain shift in Section 5.3.2, we now want to introduce our proposed augmentation pipeline. We again retain our previously optimized model hyper-parameters and only change the training set dataloader to include our RandAugment method (see Tab. 3). Figure 10 shows the resulting Dice and surface Dice scores. Looking at the Dice scores for the *oracle* in Fig. 10a we do not see significant improvement over the non-augmented Adam baseline (98.63% without augs. vs. 98.64% with augs.). However, the TD dice scores slightly improve from 97.29% to 98.35%. While we observe a similar very marginal change of oracle performance from 95.05% to 95.34% surface dice (see Fig. 10b), we see a substantial increase in target domain performance from 85.74% without augmentations to 92.55% with augmentations.

Overall, through the training pipeline optimization, switching to the Adam optimizer and introducing augmentations, we can observe that: (1) The regular dice metric diminishes in expressive power once we obtain highly performing models; (2) the oracle being close to the upper bound of segmentation precision. The remaining miss-classified pixel can be attributable to ambiguities, incorrect labels and other small variances. And (3) that through the deployment of aforementioned methods, *domain shift* between oracle and target domains has been shrunken significantly.

Evaluating augmentations Individually. To obtain a better understanding of the RandAugment method, we plotted the two worst-performing source domains Siemens1.5T and Philips1.5T. We look at the individual augmentations within RandAugment to



(a) Dice Scores (b) SDice Scores

Figure 10: Oracle and target domain performance on Dice and SDice metrics using the A2 procedure with augmentations.

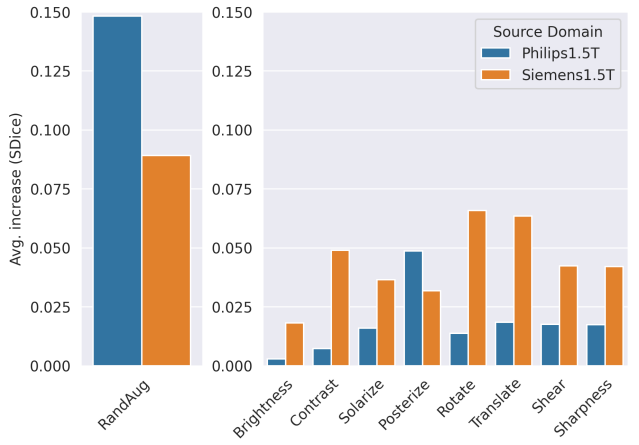


Figure 11: We plot the average target domain SDice score increase for two source domains when introducing RandAugment (left). On the right we plot the performance increase for models trained on the individual augmentations making up RandAugment.

measure their performance. Fig. 11 shows on the left-hand side the surface dice improvements on the two source domains that correspond to the values observed in Fig. 9b,10b and on the right side the models trained with individual augmentations. We see that every augmentation has benefits concerning the SDice performance, while some improvements are marginal (Siemens1.5 brightness), others are +5% and more (philips1.5 rotate & translate). We do not observe any degeneration of models in any augmentation, but neither a clear pattern for specific intensity or geometric transformations that work specifically well for any target domain. Therefore we cannot determine any definitive contributing augmentation for specific MRI domain-shifts rather than a more general observation of the increased generalization through the regularization effect induced by augmentations.

Considering additional Augmentations. While the RandAugment method already employs various intensity and geometrical augmentations, we can still scrutinize the specific augmentation choices made by evaluating additional augmentation types for additional benefits. We evaluated five additional transformations that feature somewhat different intensity or geometry-based regularization characteristics, e.g. grid distortions or cutting out small sections of the image and corresponding annotation (Cutout [DT17]). Table 4 shows that these additional augmentations all perform comparably to the *RandAugment* methods. The Cutout and Mixup methods increase the average SDice score by around 1.5%, the total-variation augmentation only yields +0.5%, while grid distortion (+2.4%) and gamma augmentations (+3.3%) work quite well too.

However, integrating additional augmentations into the RandAugment method did not yield any substantial improvement. In Table 5 we evaluate an example of a revised RandAugment version (*"RandAugment_v2"*) which includes the well-performing augmentations Gamma and GridDistortion of previous Table 4. As we do not see any substantial improvement in this example or any other revision of the RandAugment method, we assume that we hit an upper bound of generalization through the additional regularization effect of augmentations.

	Sm1.5	**Sm3**	**Ge1.5**	**Ge3**	**Phi3**	**Average**	**SDice-Δ**
Baseline	0.858	0.812	0.801	0.780	0.596	0.7694	-
TotalVariation	0.863	0.787	0.823	0.793	0.609	0.7750	+0.0056
GridDistortion	0.882	0.824	0.815	0.804	0.642	0.7934	+0.0240
Gamma	0.881	0.825	0.806	0.833	0.669	0.8028	+0.0334
Cutout [DT17]	0.871	0.807	0.822	0.818	0.600	0.7836	+0.0142
Mixup0.2 [ZCDLP17]	0.870	0.812	0.757	0.880	0.617	0.7872	+0.0178

Table 4: Additional augmentations experiments done on the source domain Philips1.5T. We measure the SDice on its 5 corresponding target domains and compare it against the baseline.

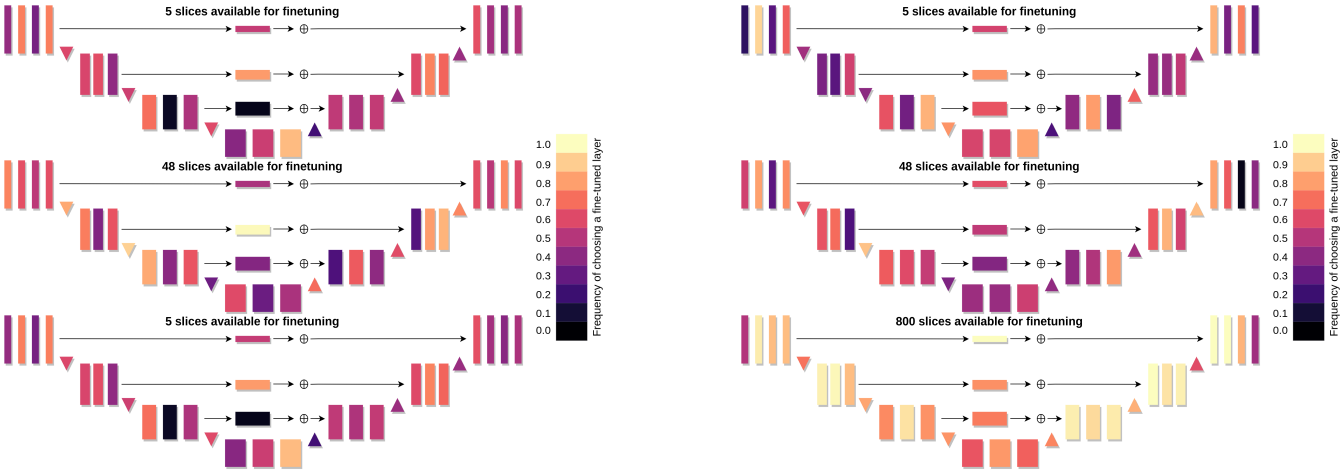
	Sm1.5	**Sm3**	**Ge1.5**	**Ge3**	**Phi3**	**Average**	**SDice-Δ**
Baseline	0.858	0.812	0.801	0.780	0.596	0.7694	-
+ RandomAug	0.936	0.913	0.933	0.945	0.861	0.9176	+0.1482
+ RandomAug_v2	0.937	0.919	0.932	0.945	0.855	0.9176	+0.1482

Table 5: We compare the Baseline against our RandAugment Method and a revised RandAugment method that additionally includes the best performing augmentations: Gamma and Griddistortion augmentations.

5.3.4. SpotTUNet Experiments

We also repeat the SpotTUnet experiment on our upgraded implementation. We employ the same configuration as we did in Section 5.2.3, only changing the learning rate. We fine-tune the SpotTUnet main U-Net with $1e-5$ and the ResNet with $1e-3$.

In Table 6 we plot one example of the Philip1.5T target domains being fine-tuned using SpotTUnet. We observe that the baseline SpotTUNet with a small number of TD slices degenerates in performance as the seemingly learned manifold is being optimized for the worse - we overfit. With more TD slices available and consequently more training data variance, this performance degeneration is reversed, and we see performance increases of +6%/+13% SDice score. Unsurprisingly, compared to the original SpotTUNet, the performance did not increase correspondingly, considering our baseline performance is already significantly higher. Looking at the SpotTUNet performance on our pre-trained model with RandAugment, we see a similar degradation using only 5 TD slices, but we



(a) Our Base SpotTUNet

(b) Our RandAugment SpotTUNet

Figure 12: Another visualization of the learned SpotTUnet. We plot the policy of the A1 procedure (left) and the A2 procedure (right) with augmentations.

do not see any improvements with 48 or 800 TD slices. The most likely cause is that the initial target domain performance is already very good (avg. 91.76%), and the small amount of additional TD data presented to the models do not contain substantially new information.

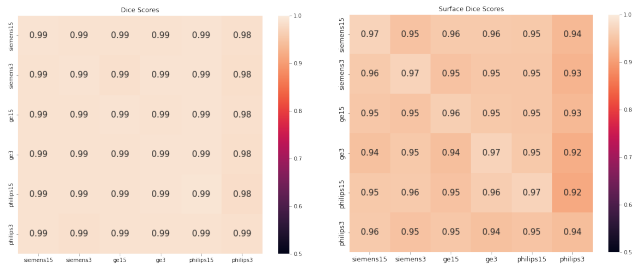
Fig. 12 of the SpotTUNet policy visualizations show us that, opposed to the original implementation, no specific U-Net layers are being chosen for fine-tuning anymore. This indicates that: (1) The Adam optimizer is learning differently, dealing with the domain shift in various layers. And (2) an overall reduction in domain shift relative to the original implementation, demanding smaller amounts of weight adaptations in more layers. The one prior observation that still holds is that we can observe the relative amount of layers being chosen for fine-tuning increases as more TD slices introduce more variance.

	TD slices	Sms1.5	Sms3	GE1.5	GE3	Phi3	Avg.	SDice-Δ
Philips1.5T SDice - Our Baseline & RandAugment SpotTune								
Default SpotTune	-	0.858	0.812	0.801	0.780	0.596	0.7694	-
	5	0.852	0.757	0.777	0.660	0.521	0.7134	-0.0560
	48	0.870	0.850	0.904	0.753	0.795	0.8344	+0.0650
	800	0.897	0.915	0.918	0.922	0.854	0.9012	+0.1318
RandAugment SpotTune	-	0.936	0.913	0.933	0.945	0.861	0.9176	-
	5	0.906	0.816	0.906	0.750	0.780	0.8316	-0.0860
	48	0.916	0.886	0.932	0.923	0.843	0.9000	-0.0176
	800	0.924	0.943	0.926	0.938	0.854	0.9170	-0.0006

Table 6: Evaluating the SpotTune performance from the Philips1.5T source domain to 5 target domains on the original baseline and RandAugment version. We evaluate the performance for 3 different amount of target domain slices available for fine-tuning.

5.4. Procedure A3: Multi-Orientation Ensemble

The Multi-Orientation Ensemble is straightforward, as it simply combines the prior *A2-procedure*, described in Table 3 and Section 5.3.3, by training three models, each with a different axis orientation of the MRI scans. We achieve this by simply changing the MRI scan transpose configuration when caching the dataset before training. The resulting ensemble predictions are shown in Figure 13. Unfortunately, at this level of model performance, the dice scores in Figure 13a do not yield any insights. The metric's upper bound is nearly reached, with an average *oracle* score of 98.6% and a target domain score of 98.6%. But using the surface Dice score, we can observe another slight performance increase on the target domains going from an average 92.5% to 94.7% SDice score with target domain is above 92%. Interestingly, the *oracle* scores also increase almost one percent-point to 96.20% indicating that the added neighbourhood information is more beneficial than augmentations for within-domain performance when no domain shift is present.



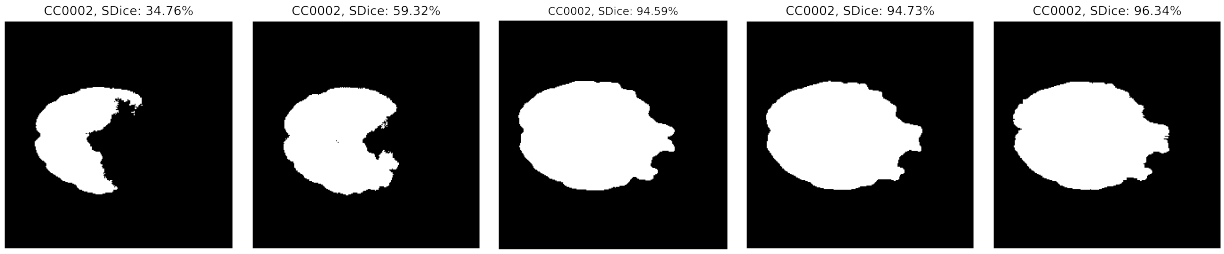
(a) Dice Scores

(b) SDice Scores

Figure 13: Oracle and target domain performance on Dice and SDice metrics using the the A3 procedure using the multi-orientation ensemble.

6. Conclusion

In this work, we showed that our initial reproduction of the SpotTUNet method with additional *RandAugment* augmentations improved generalization and reduced the domain shift on target do-



(a) Orig. Baseline (b) Aug. Baseline (c) Our Baseline (d) Our Aug. Baseline (e) Multi-Orient. Ensemble

Figure 14: We visualize an example of a slice prediction from the scan CC0002. We can visually see the improvements in the prediction. Note: The calculated surface dice here refers to the whole volume, not just the visualized slice.

mains (+16% SDice score) (see Tab. 7). The SpotTUNet with augmentations, however, did not produce any significant new insights or clues in the policy visualization, except that additional target-domain data is as helpful for the augmented model as for the baseline version. The *oracle* performance on the original implementation also failed to improve with augmentations, indicating some general optimization problems with the architecture. We then developed our own system, changing the data-loader, training pipeline and hyper-parameters that immediately improved performance, not only on the *oracle* (+9%) but also the target domains (+21%). Adding the *RandAugment* method to our own architecture reduced the domain shift even further (+6.8%). In this context, we also experimented with individual augmentations and whether any specifc transformation is responsible for the domain shift reduction. As all evaluated augmentations were more or less beneficial, we concluded that the overall regularizing effect of augmentations is the driving force for the uptick in performance. Lastly, we proposed a novel method of ensembling multiple 2D models trained on different axis orientations to reintroduce some 3D information into the predictions, resulting in further domain shift reductions and overall performance improvements (+0.9% *oracle*,+2.2% TD).

	Average. SDice		
	oracle	*target domain*	domain shift (Δ)
Baseline [ZSCB21]	86.3%	48.5%	37.8%
+ Augs.	85.0%	66.55%	18.4%
Our Method	95.0%	85.7%	9.3%
+ Augs.	95.3%	92.5%	2.8%
+ Multi-ori. Ensemble	96.2%	94.7%	1.5%

Table 7: In summary we can plot the major configurations we evaluated here, showing us the significant domain shift reductions improvement made.

Overall we managed to reduce the domain-shift using unsupervised domain adaptation, mostly through better optimization and regularization, from initially close to 40% SDice score between source and targets to a staggering 1.5% difference. This can also be seen very easily in Fig. 14, showing the five major architecture configuration we evaluated here. In future works, one might also experiment with better normalization techniques to control the raw intensities, for example, using Fuzzy C-Means normalization [RDCP19]. We analyzed brain matter-specific intensity augmentations, but they did not yield meaningful benefits in our initial experiments. Otherwise, it might be interesting to apply the lessons learned in this work to other more challenging medical image datasets, for example, brain lesion segmentation.

References

[AT16] ALJUNDI R., TUYTELAARS T.: Lightweight unsupervised domain adaptation by convolutional filter reconstruction. In *European Conference on Computer Vision* (2016), Springer, pp. 508–515. 2

[BRH*18] BLOEM J. L., REIJNIERSE M., HUIZINGA T. W., VAN DER HELM-VAN A. H., ET AL.: Mr signal intensity: staying on the bright side in mr image interpretation. *RMD open 4*, 1 (2018), e000728. 1

[BTB18] BERMAN M., TRIKI A. R., BLASCHKO M. B.: The lovász-softmax loss: A tractable surrogate for the optimization of the intersection-over-union measure in neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2018), pp. 4413–4421. 4, 5

[C*20] CONSORTIUM M., ET AL.: Monai: Medical open network for ai, 2020. 3

[CZSL19] CUBUK E. D., ZOPH B., SHLENS J., LE Q. V.: Randaugment: Practical automated data augmentation with a reduced search space. arxiv e-prints, page. *arXiv preprint arXiv:1909.13719* (2019). 4

[DOC*18] DOU Q., OUYANG C., CHEN C., CHEN H., HENG P.-A.: Unsupervised cross-modality domain adaptation of convnets for biomedical image segmentations with adversarial loss. *arXiv preprint arXiv:1804.10916* (2018). 1, 2

[DT17] DEVRIES T., TAYLOR G. W.: Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552* (2017). 9

[GSK*19] GUO Y., SHI H., KUMAR A., GRAUMAN K., ROSING T., FERIS R.: Spottune: Transfer learning through adaptive fine-tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2019). 1, 3

[GUA*16] GANIN Y., USTINOVA E., AJAKAN H., GERMAIN P., LAROCHELLE H., LAVIOLETTE F., MARCHAND M., LEMPITSKY V.: Domain-adversarial training of neural networks. *The journal of machine learning research 17*, 1 (2016), 2096–2030. 2

[HZRS16] HE K., ZHANG X., REN S., SUN J.: Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2016), pp. 770–778. 3

[JGP16] JANG E., GU S., POOLE B.: Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144* (2016). 3

[KBL*16] KAMNITSAS K., BAUMGARTNER C., LEDIG C., NEWCOMBE V., SIMPSON J., KANE A., MENON D., NORI A., CRIMINISI A., RUECKERT D., ET AL.: Unsupervised domain adaptation in brain lesion segmentation with adversarial networks. *arXiv preprint arXiv:1612.08894* (2016). 2

[KBL*17] KAMNITSAS K., BAUMGARTNER C., LEDIG C., NEWCOMBE V., SIMPSON J., KANE A., MENON D., NORI A., CRIMINISI A., RUECKERT D., ET AL.: Unsupervised domain adaptation in brain lesion segmentation with adversarial networks. In *International conference on information processing in medical imaging* (2017), Springer, pp. 597–609. 2

[KVGV*19] KUSHIBAR K., VALVERDE S., GONZALEZ-VILLA S., BERNAL J., CABEZAS M., OLIVER A., LLADO X.: Supervised domain adaptation for automatic sub-cortical brain structure segmentation with minimal user interaction. *Scientific reports 9*, 1 (2019), 1–15. 2

[LGG*17] LIN T.-Y., GOYAL P., GIRSHICK R., HE K., DOLLÁR P.: Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision* (2017), pp. 2980–2988. 4

[LKB*17] LITJENS G., KOOI T., BEJNORDI B. E., SETIO A. A. A., CIOMPI F., GHAFOORIAN M., VAN DER LAAK J. A., VAN GINNEKEN B., SÁNCHEZ C. I.: A survey on deep learning in medical image analysis. *Medical image analysis 42* (2017), 60–88. 1

[MMKSM18] MADANI A., MORADI M., KARARGYRIS A., SYEDA-MAHMOOD T.: Semi-supervised learning with generative adversarial networks for chest x-ray classification with ability of data domain adaptation. In *2018 IEEE 15th International symposium on biomedical imaging (ISBI 2018)* (2018), IEEE, pp. 1038–1042. 2

[NBZ*18] NIKOLOV S., BLACKWELL S., ZVEROVITCH A., MENDES R., LIVNE M., DE FAUW J., PATEL Y., MEYER C., ASKHAM H., ROMERA-PAREDES B., ET AL.: Deep learning to achieve clinically applicable segmentation of head and neck anatomy for radiotherapy. *arXiv preprint arXiv:1809.04430* (2018). 4

[RDCP19] REINHOLD J. C., DEWEY B. E., CARASS A., PRINCE J. L.: Evaluating the impact of intensity normalization on mr image synthesis. In *Medical Imaging 2019: Image Processing* (2019), vol. 10949, SPIE, pp. 890–898. 11

[SLG*18] 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* (2018), 482–494. 1, 2

[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*. Springer, 2020, pp. 117–126. 1

[VLB*18] VALINDRIA V. V., LAVDAS I., BAI W., KAMNITSAS K., ABOAGYE E. O., ROCKALL A. G., RUECKERT D., GLOCKER B.: Domain adaptation for mri organ segmentation using reverse classification accuracy. *arXiv preprint arXiv:1806.00363* (2018). 2

[WD18] WANG M., DENG W.: Deep visual domain adaptation: A survey. *Neurocomputing 312* (2018), 135–153. 1

[WER18] WOLLMANN T., EIJKMAN C., ROHR K.: Adversarial domain adaptation to improve automatic breast cancer grading in lymph nodes. In *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)* (2018), IEEE, pp. 582–585. 2

[YDZ*19] YANG J., DVORNEK N. C., ZHANG F., CHAPIRO J., LIN M., DUNCAN J. S.: Unsupervised domain adaptation via disentangled representations: Application to cross-modality liver segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention* (2019), Springer, pp. 255–263. 2

[ZCDLP17] ZHANG H., CISSE M., DAUPHIN Y. N., LOPEZ-PAZ D.: mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412* (2017). 9

[ZSCB21] ZAKAZOV I., SHIROKIKH B., CHERNYAVSKIY A., BELYAEV M.: Anatomy of domain shift impact on u-net layers in mri segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention* (2021), Springer, pp. 211–220. 1, 2, 3, 4, 5, 6, 7, 8, 11