Joint Optimization of Class-Specific Training- and Test-Time Data Augmentation in Segmentation

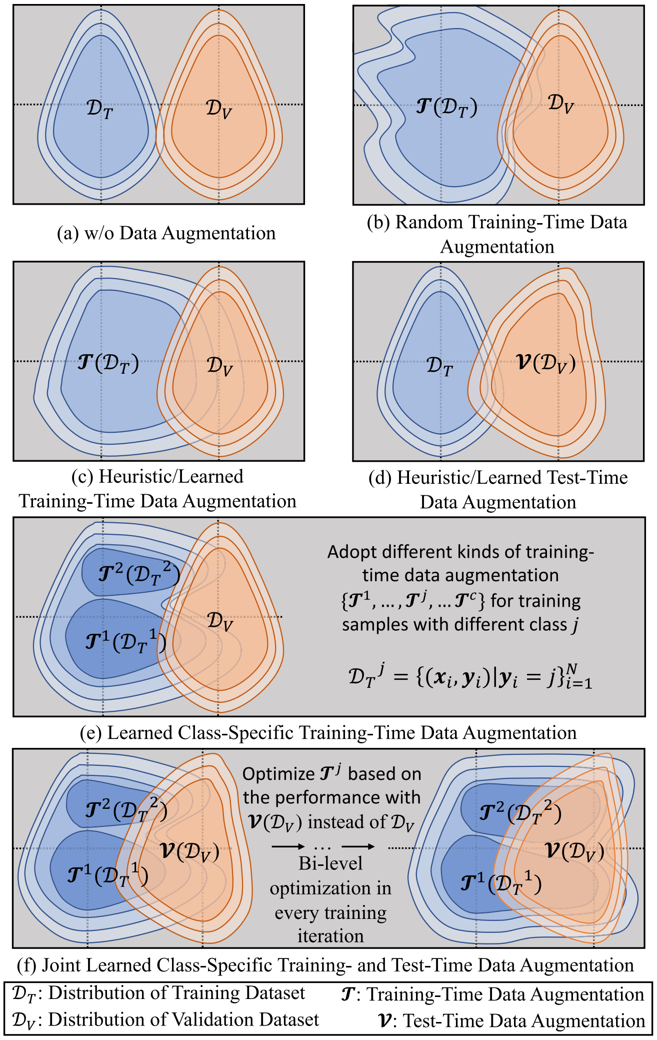
Zeju Li, Konstantinos Kamnitsas, Qi Dou, Chen Qin, and Ben Glocker

***Abstract*— This paper presents an effective and general data augmentation framework for medical image segmentation. We adopt a computationally efficient and data-efficient gradient-based meta-learning scheme to explicitly align the distribution of training and validation data which is used as a proxy for unseen test data. We improve the current data augmentation strategies with two core designs. First, we learn class-specific training-time data augmentation (TRA) effectively increasing the heterogeneity within the training subsets and tackling the class imbalance common in segmentation. Second, we jointly optimize TRA and test-time data augmentation (TEA), which are closely connected as both aim to align the training and test data distribution but were so far considered separately in previous works. We demonstrate the effectiveness of our method on four medical image segmentation tasks across different scenarios with two state-of-the-art segmentation models, DeepMedic and nnU-Net. Extensive experimentation shows that the proposed data augmentation framework can significantly and consistently improve the segmentation performance when compared to existing solutions. Code is publicly available at https://github.com/ZerojumpLine/JCSAugment.**

***Index Terms*— Data augmentation, meta-learning, image segmentation.**

I. Introduction

DATA augmentation is a de facto technique in neural networks and has shown to improve model generalization [34]. It is essential for medical image segmentation algorithms to perform well on unseen test data. Depending on when it is performed, we can divide data augmentation into training-time data augmentation (TRA) and test-time data augmentation (TEA). TRA aims to increase the variation


$\mathcal{D}_T$
$\mathcal{D}_V$
$\mathcal{T}(\mathcal{D}_T)$
$\mathcal{V}(\mathcal{D}_V)$
$\mathcal{T}^2(\mathcal{D}_T{}^2)$
$\mathcal{T}^1(\mathcal{D}_T{}^1)$
Bi-level optimization in every training iteration

(a) w/o Data Augmentation

(b) Random Training-Time Data Augmentation

(c) Heuristic/Learned Training-Time Data Augmentation

(d) Heuristic/Learned Test-Time Data Augmentation

Adopt different kinds of training-time data augmentation $\{\mathcal{T}^1, \ldots, \mathcal{T}^j, \ldots \mathcal{T}^c\}$ for training samples with different class j

$\mathcal{D}_T{}^j = \{(\boldsymbol{x}_i, \boldsymbol{y}_i) | \boldsymbol{y}_i = j\}_{i=1}^N$

(e) Learned Class-Specific Training-Time Data Augmentation

Optimize $\mathcal{T}^j$ based on the performance with $\mathcal{V}(\mathcal{D}_V)$ instead of $\mathcal{D}_V$

(f) Joint Learned Class-Specific Training- and Test-Time Data Augmentation

$\mathcal{D}_T$: Distribution of Training Dataset $\mathcal{T}$: Training-Time Data Augmentation
$\mathcal{D}_V$: Distribution of Validation Dataset $\mathcal{V}$: Test-Time Data Augmentation

Fig. 1. Data augmentation improves segmentation model performance by aligning the training and validation/test data distribution. As illustrated in (b) [8], (c) [7], [16], [18], [25], [28] and (d) [20], [33], current methods optimize the data distributions by using a validation set as a proxy for unseen test data. Our framework brings improvements by integrating two conceptually simple and intuitive ideas: (e) We adopt different kinds of training-time data augmentation (TRA) for training samples from different classes, effectively extending the training data distribution and alleviating the class imbalance issue. (f) We jointly optimize TRA and test-time data augmentation (TEA) during every training iteration, making the data distributions overlaps more.

Manuscript received 8 April 2023; accepted 28 May 2023. Date of publication 5 June 2023; date of current version 27 October 2023. This work was supported by the ERC under the EU's Horizon 2020 Research and Innovation Program under Grant 757173. The work of Zeju Li was supported by the China Scholarship Council (CSC) Imperial Scholarship. *(Corresponding author: Zeju Li.)*

Zeju Li and Ben Glocker are with the BioMedIA Group, Department of Computing, Imperial College London, SW7 2AZ London, U.K. (e-mail: zeju.li18@imperial.ac.uk; b.glocker@imperial.ac.uk).

Konstantinos Kamnitsas is with the BioMedIA Group, Department of Computing, Imperial College London, SW7 2AZ London, U.K., also with the Department of Engineering Science, University of Oxford, OX3 7DQ Oxford, U.K., and also with the School of Computer Science, University of Birmingham, B15 2TT Birmingham, U.K. (e-mail: konstantinos.kamnitsas12@imperial.ac.uk).

Qi Dou is with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong 999077 (e-mail: qidou@cuhk.edu.hk).

Chen Qin is with I-X and the Department of Electrical and Electronic Engineering, Imperial College London, SW7 2AZ London, U.K. (e-mail: chen.qin@ed.ac.uk).

Digital Object Identifier 10.1109/TMI.2023.3282728

captured by the training dataset by adding perturbed samples with the goal to capture the unseen test data distribution. TEA robustifies the final prediction by averaging predictions of predefined, assumed non-causal variations of test data, to which the model should be robust [35]. An alternative approach for TEA is to modify the test data to achieve higher accuracy with the pretrained model by transforming the test samples to match the distribution of the training data, which is the opposite direction of TRA. Here, we consider to make TRA and TEA complement each other towards the goal of more accurate and robust predictions.

Common data augmentation strategies are usually designed based on heuristics and manually tuned configurations with respect to reducing validation error [16], [18]. However, strategies designed for one task may not be optimal for another task or dataset. Consequently, data augmentation without considering data and task characteristics may not always improve the model performance. In particular, different medical image segmentation tasks may require different data augmentation settings, due to changes in image acquisition protocols, modalities, and anatomical structures of interest [16]. It is tedious to hand-engineer suitable augmentation strategies for each individual task. Therefore, methods have been proposed to automatically learn effective augmentations directly from the available training data [7], [20], [25], [28], [33], [36], [37].

However, we argue that there are two major limitations constraining the performance of current data augmentation strategies. First, previous studies [7], [20], [28], [33] mostly focus on either TRA or TEA separately, without considering their connections, despite the two being closely linked. This could lead to suboptimal results as the test condition can be adapted through TEA which is not taken into account by TRA when the two are considered in isolation. Second, most TRAs adopt the same transformations for all the samples without considering the different properties existing in different classes. Specifically, the foreground samples in segmentation are more prone to overfitting than background samples because they underrepresented due to class imbalance [26]. Current data augmentation strategies fail to model the heterogeneity of samples from different classes and the resulting model performance may suffer from overfitting under class imbalance.

In this study, we aim to bridge the gap between TRA and TEA by presenting a gradient-based meta-learning framework to automatically discover optimal TRA and TEA strategies, simultaneously. As illustrated in Fig. 1(a,b,c,d), data augmentation improves model generalization by aligning the training and underlying test data distribution. Our data augmentation framework (c.f. Fig. 1(e, f)) further takes class properties and test condition into account, fundamentally restructuring the data distributions aiming for an increased overlap. We validate our method with medical image segmentation because of its imbalanced nature and clinical importance.

The contributions of this study can be summarized as follows: 1) We build a bridge between TRA and TEA through joint optimization of data augmentation policies during the training process, which improves alignment of training and test sample distributions and yields better generalization. 2) We introduce a method that automatically finds different TRA policies for training samples from different classes, implicitly addressing the class imbalance problem. 3) We design a transformation set for TRA with 15 cascaded transformations and 47 operations in total, as well as a transformation set with 83 operations for TEA. These transformation sets cover most transformations in medical image segmentation and can also be easily extended and applied to other applications. 4) Extensive experiments performed on four datasets with two state-of-the-art segmentation models show that our method can consistently improve segmentation performance in various applications and demonstrate the potential to replace the heuristically chosen augmentation policies currently used in most previous works.

II. Related Work

A. Data Augmentation Model

The majority of data augmentation strategies consist of a set of transformations defined based on domain knowledge to represent the heterogeneity of the test data. Examples include rotations, flipping, and intensity shifts [6], [21]. On the other hand, there are also heuristic perturbation techniques such as cutout [9] and mixup [38] that, even though they lead to unrealistic synthetic samples, have been empirically found to improve model generalization. More realistic transformations can be generated based on properties matching [39] or generative adversarial networks [13]. Although these techniques showed promising performance, the design of data augmentation is difficult because it requires prior knowledge about the task at hand. Optimal strategies, however, may differ significantly between different tasks, datasets and types of input modalities, and thus will be difficult to hand engineer [7], [16]. In this study, we aim to automate the process of designing data augmentation.

Currently most TRA methods adopt the same transformations to all the training samples except [26], which proposed to increase the variance of foreground samples by heuristically reducing the number of transformed samples for the background classes in order to alleviate class imbalance. However, they found different hyper-parameters are optimal for different datasets, and the chosen transformations and hyper-parameters were based on heuristics. In contrast, our method automatically learns different transformations for different classes and discovers the rules from the training data by itself.

B. Learning Based Training-Time Data Augmentation

There have been many attempts to optimize TRA along with the training process to obtain task-specific TRA policies. Most of the studies are developed based on the idea of adversarial training [12]. The basic adversarial augmentation might not improve generalization on real data as the constructed samples are not realistic. Recent methods attempt to improve real data heterogeneity by adopting an advanced augmentation model [4] or restricting the search space [31], which require strong prior knowledge. Different from these, some methods were proposed to generate artificial samples with task constraints [3], which encourage a generative model to produce additional well-classified images with class properties to enlarge the training data distribution. However, the well-classified samples might not be very useful when the training data is sufficient as they would not make significant changes to the learning of the decision boundary.

Our method is closely related to the line of research which optimizes TRA based on the validation performance such that the learned model can best generalize. Those methods find the sets of augmentation policies that are optimal for a specific training database, out of a pool of possible transformations, based on reinforcement learning [7], [37], meta-learning [25], [36], or density matching [28]. In our study, we consider to learn the parameters of a probability distribution over TRA with a meta-learning scheme. The meta-learner parameters are optimized with the aim of enabling the task segmentation network to perform better on a validation set. In this way, the meta-learner is explicitly trained to select augmentations that improve generalization. Our method improves existing solutions by the joint optimization of TRA and TEA as well as learning a separate augmentation per class. In addition, the defined transformation pool in our work is more comprehensive than previous studies for medical image segmentation, making it more practical to improve upon current heuristic solutions.

C. Learning Based Test-Time Data Augmentation

In TEA, class-posterior probabilities from multiple predictions are averaged after applying predefined transformations to the test sample, which was found to be effective to improve accuracy. Recently, some methods were proposed to learn TEA by choosing the transformations obtaining low loss values on the validation set based on a pre-trained model [20], [33]. Test-time adaptation is another kind of learning based TEA where the pre-trained model is adapted to fit a single test sample based on denoising autoencoder [19] or self-supervision [14]. These learning based TEA strategies can be seen as a post-processing to the segmentation and do not contribute to the learning of the model. In contrast, our method proposes to combine the optimization of TRA and TEA during training, which leads to not only learning the optimal TRA and TEA transformations that complement each other, but also learning optimal model parameters given the specific set of data transformations.

III. Method

A. Preliminaries

We consider the image segmentation problem with c total number of classes. A training dataset $\mathcal{D}_T = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^N$ with N samples is given, where $\boldsymbol{x}_i$ is a training image and $\boldsymbol{y}_i$ corresponds to the segmentation label map with individual labels $y_{ip} \in \{1, \ldots, c\}$ for each image pixel p. Assuming a segmenter f_θ parameterized by θ, our aim is to learn optimal θ^* parameters, such that $f_{\theta^*}(\cdot)$ minimizes the empirical risk over the training data. For any training loss function $\mathcal{L}_{train}$, the empirical risk of the segmentation model f_θ is defined as $R_{\mathcal{L}_{train}}(f_\theta) = \frac{1}{N}\sum_{i=1}^N \mathcal{L}_{train}(f_\theta(\boldsymbol{x}_i), \boldsymbol{y}_i)$. Apart from $\mathcal{D}_T$, we usually have a validation dataset $\mathcal{D}_V = \{(\tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{y}}_i)\}_{i=1}^M$ with M samples along with a validation loss $\mathcal{L}_{val}$, which is taken as a proxy for unseen test data and used to tune the hyper-parameters including learning rates [24], network architecture [40] and data augmentation policies [7]. Note that $\mathcal{D}_V$ could come from a different distribution from $\mathcal{D}_T$, based on different assumptions of unseen test data.

B. Sampling Transformations

For a sample $\boldsymbol{x}_i$ (or $\tilde{\boldsymbol{x}}_i$), we will apply transformation $\mathcal{T}_i(\cdot)$ which is specific to the i-th sample. $\mathcal{T}_i$ is obtained by sampling from a set of K operations $\{\mathcal{O}^1, \ldots, \mathcal{O}^K\}$ based on the corresponding probability distribution $\boldsymbol{p} = [p_1, \ldots, p_K]^\intercal$. In this study we represent with a different $\mathcal{O}^j$ operation not only transformations of different type (for example rotations, contrast enhancement, etc.) but also transformations of the same type but different magnitudes (for example rotations of different degree). We do not further optimize the predefined magnitudes of transformations during training. Our method will optimize during training the sampling distribution $\boldsymbol{p}$ of different transformations for data augmentation, so that we learn which transformations are most appropriate for the given dataset and task.

In order to include the distribution into the gradient based optimization through the non-differentiable sampling process, we reparameterize the categorical distribution using the Gumbel-Softmax trick [17]. We calculate the probability of assigning sample $\boldsymbol{x}_i$ (or $\tilde{\boldsymbol{x}}_i$) with operation $\mathcal{O}^j$ as:

$$s_{ij} = \frac{\mathrm{e}^{p_j + g_{ij}}}{\sum_{v=1}^K \mathrm{e}^{p_v + g_{iv}}}, \qquad \text{for } j = 1, \ldots, K, \tag{1}$$

where g_{ij} is a sample drawn from the Gumbel distribution, i.e., $g_{ij} = -\log(-\log(\varepsilon))$, in which ε is a random number by drawing $\varepsilon \sim \text{Uniform}(0, 1)$. It then holds that $\sum_{j=1}^K s_{ij} = 1$ and $0 \le s_{ij} \le 1, \forall j$. In this way, the stochasticity involved in the sampling process is removed from the computational graph of network's training and the process of choosing augmentation $\mathcal{T}_i$ based on probability distribution $\boldsymbol{p}$ now becomes differentiable. Specifically, $\mathcal{T}_i$ is chosen as $\mathcal{O}^{j^*}$ where $j^* = \text{argmax}_j(s_{ij})$. The sampling probability $\boldsymbol{p}$, which we would like to optimize, can still not be updated via backpropagation, both due to the non differentiable argmax and because the transformations are non-differentiable in the general case. To work around this, we also calculate a weight w_i that corresponds to the sample $\boldsymbol{x}_i$ (or $\tilde{\boldsymbol{x}}_i$) with:

$$w_i = \max_j\ (s_{ij}) + \underbrace{(1 - \max_j\ (s_{ij}))}_{\text{does not require gradient}}, \tag{2}$$

which is a function of the sampling probability s_{ij}. We then incorporate the weight into the empirical risk as $R'_{\mathcal{L}_{train}}(f_\theta) = \frac{1}{N}\sum_{i=1}^N w_i \mathcal{L}_{train}(f_\theta(\mathcal{T}_i(\boldsymbol{x}_i)), \boldsymbol{y}_i)$. In this manner, w_i and s_{ij} are part of the total loss and hence can be straightforwardly

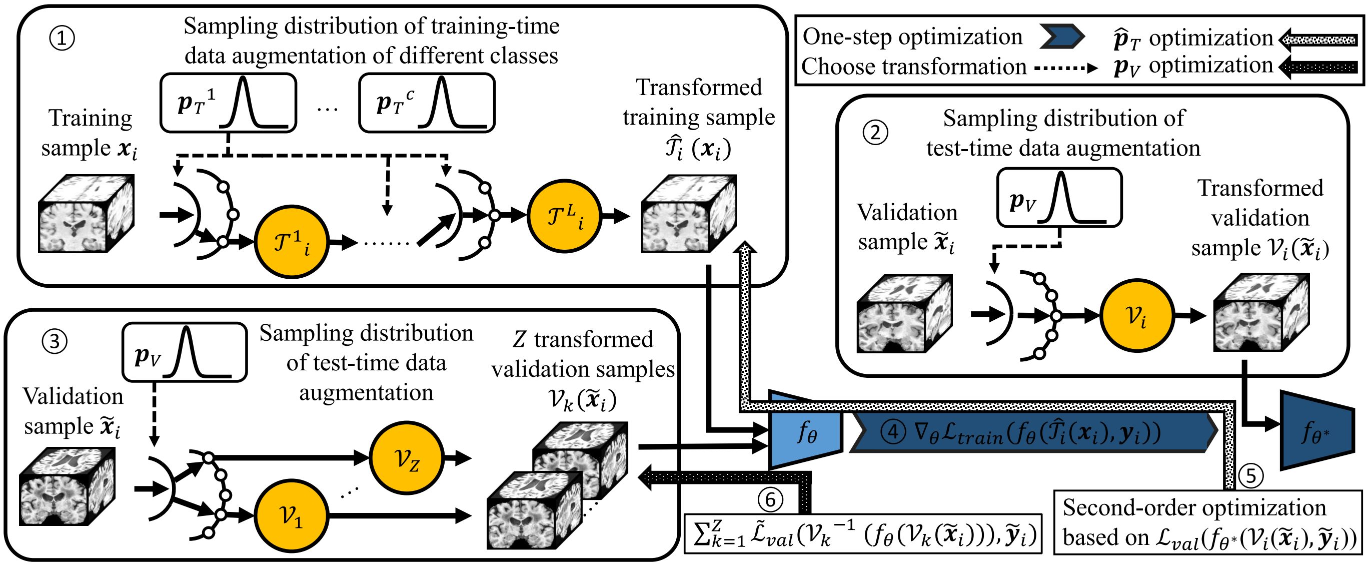

①
Sampling distribution of training-time data augmentation of different classes
Training sample $\boldsymbol{x}_i$
$\boldsymbol{p}_T{}^1$
…
$\boldsymbol{p}_T{}^c$
Transformed training sample $\hat{\mathcal{T}}_i\,(\boldsymbol{x}_i)$
$\mathcal{T}^1{}_i$
……
$\mathcal{T}^L{}_i$
One-step optimization
Choose transformation
$\hat{\boldsymbol{p}}_T$ optimization
$\boldsymbol{p}_V$ optimization
②
Sampling distribution of test-time data augmentation
Validation sample $\tilde{\boldsymbol{x}}_i$
$\boldsymbol{p}_V$
Transformed validation sample $\mathcal{V}_i(\tilde{\boldsymbol{x}}_i)$
$\mathcal{V}_i$
③
$\boldsymbol{p}_V$
Sampling distribution of test-time data augmentation
Validation sample $\tilde{\boldsymbol{x}}_i$
Z transformed validation samples $\mathcal{V}_k(\tilde{\boldsymbol{x}}_i)$
$\mathcal{V}_Z$
$\mathcal{V}_1$
f_θ
④ $\nabla_\theta\mathcal{L}_{train}(f_\theta(\hat{\mathcal{T}}_i(\boldsymbol{x}_i)),\boldsymbol{y}_i))$
f_{θ^*}
⑤
Second-order optimization based on $\mathcal{L}_{val}(f_{\theta^*}(\mathcal{V}_i(\tilde{\boldsymbol{x}}_i),\tilde{\boldsymbol{y}}_i))$
⑥
$\sum_{k=1}^{Z}\tilde{\mathcal{L}}_{val}(\mathcal{V}_k{}^{-1}\,(f_\theta(\mathcal{V}_k(\tilde{\boldsymbol{x}}_i))),\tilde{\boldsymbol{y}}_i)$

Fig. 2. The optimization process of the proposed method. In this study, data augmentation is formulated as the probability distribution of multiple predefined transformations, as demonstrated in ①, ② and ③. During the same iteration, class-specific TRA is optimized based on meta-gradients with ⑤ while TEA is optimized based on the validation losses of Z transformed samples with ⑥.

optimized. During the forward propagation, we utilize w_i to evaluate the chosen transformation $\mathcal{T}_i$ without affecting the training procedure, as w_i is always equal to 1. During back-propagation, w_i that is associated with a relatively effective $\mathcal{T}_i$ is prone to be increased. As we enforce the computation of the second term in Eq. 2 to never require gradient, we can use w_i as a means to optimize s_{ij} and thus $\boldsymbol{p}$ with gradient descend.

For distinction between TRA and TEA, in the following paragraphs we define the probability distribution and transformation of TRA as $\boldsymbol{p}_T$ and $\hat{\mathcal{T}}_i$ while denoting the ones of TEA as $\boldsymbol{p}_V$ and $\mathcal{V}_i$ unless otherwise noted. Note that we are considering the problem of image segmentation, therefore the spatial transformations are always applied to $\boldsymbol{y}_i$ simultaneously but we omit this for simplicity.

C. Overview of the Training Process

We aim to reduce the generalization gap explicitly by optimizing a probability distribution over data augmentations $\boldsymbol{p}_T$ (or $\boldsymbol{p}_V$) based on the gradient from the validation data $\nabla_{\boldsymbol{p}_T}\mathcal{L}_{val}$ (or $\nabla_{\boldsymbol{p}_V}\mathcal{L}_{val}$). Thus, $\boldsymbol{p}_T$ (or $\boldsymbol{p}_V$) is automatically adapted to the underlying task-specific characteristics.

We develop a training framework based on meta-learning via second-order optimization to accomplish this. The optimization process of the proposed method is illustrated in Fig. 2. During the sampling process, we first obtain the transformed training data with ① and optimize the model to f_{θ^*} with ④ based on a single optimization step; then we pass the transformed validation data with ② through f_{θ^*} to compute the second-order gradients ⑤ and backprop to ①, to learn TRA that leads to learning θ^* that best generalizes on validation data transformed with TEA; meanwhile, we also apply varied transformations of TEA to a single validation sample with ③ and we try to learn TEA $\boldsymbol{p}_V$ which can transform a single validation sample $\tilde{\boldsymbol{x}}_i$ to have the lowest validation error with ⑥.

D. Learning of Class-Specific Training-Time Data Augmentation

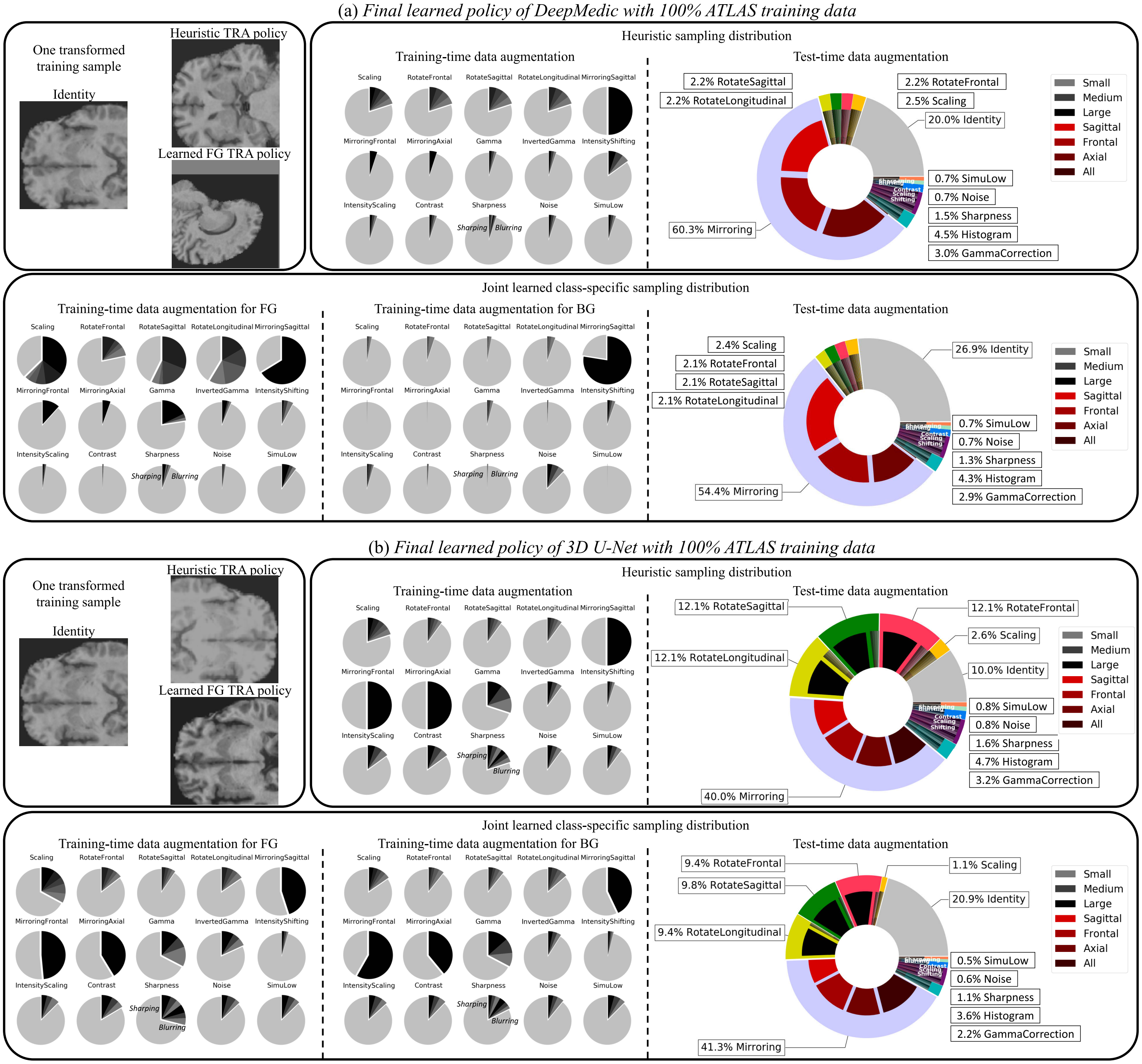
1) The Design of Predefined Transformations: Following the design of data augmentation in many medical image segmentation frameworks [16], [18], we design the transformation set with L=15 cascaded operations including rotation, mirroring, gamma correction, histogram transformations, blurring, sharpening, adding noise, and simulating low resolution. The operation magnitudes are decided by uniformly sampling from predefined ranges. We summarize the detailed information about the operations in supplementary material. Specifically, the probability distribution and transformations of TRA is extended as $\boldsymbol{p}_T = (\boldsymbol{p}^1, \ldots, \boldsymbol{p}^L)$ and $\hat{\mathcal{T}}_i = \{\mathcal{T}^1{}_i, \ldots, \mathcal{T}^L{}_i\}$.

We ensure that our design of TRA is able to accomplish the same functionality with the built-in data augmentation in prevailing frameworks such as DeepMedic [18] and nnU-Net [16], therefore our method can act as a replacement for heuristic TRA. We initialize $\boldsymbol{p}_T$ with heuristic policies provided by these frameworks, as shown in Fig. 3.

2) Class-Specific Data Augmentation: We adopt different TRAs for training samples from different classes. Specifically, we extend the probability distribution to $\hat{\boldsymbol{p}}_T = (\boldsymbol{p}_T{}^1, \ldots, \boldsymbol{p}_T{}^c)$ which contains different probability distributions for c classes. In this way, TRA becomes more flexible and powerful as it gains the ability to draw $\hat{\mathcal{T}}_i$ from different distributions for different classes.

In practice, we determine the class of a training patch with the central pixel of the patch. Note that in this study we only regard the training samples to come from 2 classes consisting of foreground (tumor, lesion, and organs) and background.

3) Policy Optimization With Meta-Gradients: Similar to previous works on learning TRA [7], [25], we aim to learn TRA


(a) *Final learned policy of DeepMedic with 100% ATLAS training data*

(b) *Final learned policy of 3D U-Net with 100% ATLAS training data*

Fig. 3. The heuristic data augmentation policy and the learned probability distributions over augmentations based on different segmentation models for brain stroke lesion segmentation with 100% ATLAS training data. We also visualize an example of the transformed foreground (FG) training sample with different sampling distributions for TRA. Our framework provides application-specific and class-specific data augmentation policies. We find the learned policies would adopt larger transformations to the FG than the background (BG) samples, implicitly alleviating the class imbalance issue.

based on the performance of validation data and formulate the optimization of TRA as a bi-level optimization problem:

$$\min_{\hat{\boldsymbol{p}}_T} \frac{1}{M}\sum_{i=1}^{M}\mathcal{L}_{val}(f_{\theta^*}(\tilde{\boldsymbol{x}}_i),\tilde{\boldsymbol{y}}_i) \tag{3}$$

$$s.t.\ \theta^* = \operatorname*{argmin}_{\theta} \frac{1}{N}\sum_{i=1}^{N} w_i\mathcal{L}_{train}(f_\theta(\hat{\mathcal{T}}_i(\boldsymbol{x}_i)),\boldsymbol{y}_i). \tag{4}$$

We propose to solve this based on gradient descent following [11] and [32]. We train the model with a training batch containing n samples and a validation batch consisting of m samples. For simplicity, we shorten $\frac{1}{n}\sum_{i=1}^{n} w_i\mathcal{L}_{train}(f_\theta(\hat{\mathcal{T}}_i(\boldsymbol{x}_i)),\boldsymbol{y}_i))$ as $\mathcal{L}_{train}(\theta,\hat{\boldsymbol{p}}_T)$ and $\frac{1}{m}\sum_{i=1}^{m}\mathcal{L}_{val}(f_{\theta^*}(\tilde{\boldsymbol{x}}_i),\tilde{\boldsymbol{y}}_i))$ as $\mathcal{L}_{val}(\theta^*)$ in the following paragraphs. Based on the chain rule, the gradient of validation loss w.r.t. $\hat{\boldsymbol{p}}_T$ is derived as:

$$\nabla_{\hat{\boldsymbol{p}}_T}\mathcal{L}_{val}(\theta^*) = (\frac{\partial\theta^*}{\partial\hat{\boldsymbol{p}}_T})^\intercal\nabla_\theta\mathcal{L}_{val}(\theta^*), \tag{5}$$

where $\nabla_{\hat{\boldsymbol{p}}_T} = (\frac{\partial}{\partial\hat{\boldsymbol{p}}_T})^\intercal$ and $\nabla_\theta = (\frac{\partial}{\partial\theta})^\intercal$. The calculation of $\frac{\partial\theta^*}{\partial\hat{\boldsymbol{p}}_T}$ can be derived based on implicit function theorem [1].

However, the calculation would introduce a Hessian which is not practical to calculate with the parameters of deep nerual network as the number of parameters is too large. There are many methods to approximate the gradient without Hessian calculation [11], [29], [32], in this study we choose to approximate θ^* by using a single training step [10], [29]. Specifically, we approximate the optimal θ^* via a standard training step with:

$$\theta^* \approx \theta - \alpha \nabla_\theta \mathcal{L}_{train}(\theta, \hat{\boldsymbol{p}}_T). \tag{6}$$

Here, α is the step length which we set equal to the learning rate of the task model. Eq. 6 defines the approximated optimal θ^* when trained using the training data with sampled data augmentation $\hat{\mathcal{T}}_i$. In this manner, we can evaluate the effectiveness of the data augmentation policy $\hat{\boldsymbol{p}}_T$ based on the performance of the updated model f_{θ^*} on a held-out validation dataset. We differentiate this equation w.r.t. $\hat{\boldsymbol{p}}_T$ from both sides and yield:

$$\frac{\partial \theta^*}{\partial \hat{\boldsymbol{p}}_T} = -\alpha \nabla^2_{\theta, \hat{\boldsymbol{p}}_T} \mathcal{L}_{train}(\theta^*, \hat{\boldsymbol{p}}_T), \tag{7}$$

where $\nabla^2_{\theta, \hat{\boldsymbol{p}}_T} = \frac{\partial \nabla_\theta}{\partial \hat{\boldsymbol{p}}_T}$. By substituting Eq. 7 into Eq. 5, now we can update $\hat{\boldsymbol{p}}_T$ with:

$$\begin{aligned} \hat{\boldsymbol{p}}_T^{t+1} &= \hat{\boldsymbol{p}}_T^t - \beta \nabla_{\hat{\boldsymbol{p}}_T^t} \mathcal{L}_{val}(\theta^*) \\ &= \hat{\boldsymbol{p}}_T^t + \alpha \beta \nabla^2_{\hat{\boldsymbol{p}}_T^t, \theta} \mathcal{L}_{train}(\theta^*, \hat{\boldsymbol{p}}_T^t) \nabla_\theta \mathcal{L}_{val}(\theta^*), \end{aligned} \tag{8}$$

which can be interpreted as a gradient of the gradient from the task-driven training. In the above, β is the learning rate for determining the probability distribution. In this way, we can optimize the distribution $\hat{\boldsymbol{p}}_T$ explicitly with the aim to improve generalization of the segmentation model using the validation data. Eq. 8 includes a second-order gradient. As the distribution $\hat{\boldsymbol{p}}_T$ is represented with only a few parameters (K, which is in the order of 10-100), we find the complexity of the gradient computation to be $O(|\hat{\boldsymbol{p}}_T||\theta|)$ which is feasible and can be handled by prevailing toolboxes such as PyTorch and Tensorflow.

After updating $\hat{\boldsymbol{p}}_T$, we update f_θ to f_{θ^*} to fit the updated TRA policy for the next iteration. We optimize $\hat{\boldsymbol{p}}_T$ along with the training of the task model, and at the end of the training we may have higher performance than any model learned with a random or manually configured augmentation policy.

4) Gradient Normalization: We normalize the gradient from different classes of training samples, as we notice that the contributions of training samples from different classes to the reduction of the validation loss varied a lot. For example, the foreground samples are more effective for reducing the validation loss, resulting in increased probabilities of the policies associated with the foreground samples. Specifically, if we rewrite the gradient in Eq. 5 by the chain rule as:

$$\nabla_{\hat{\boldsymbol{p}}_T} \mathcal{L}_{val}(\theta^*) = \sum_{i=1}^{n} \nabla_{w_i} \mathcal{L}_{val}(\theta^*) (\frac{\partial w_i}{\partial \hat{\boldsymbol{p}}_T})^\intercal, \tag{9}$$

we would find the magnitude of $\nabla_{w_i} \mathcal{L}_{val}(\theta^*)$ is significantly larger for the foreground samples than the background samples. To resolve this, we rewrite Eq. 9 as $\nabla_{\hat{\boldsymbol{p}}_T} \mathcal{L}_{val}(\theta^*) = \sum_{i=1}^{n} h_i (\frac{\partial w_i}{\partial \hat{\boldsymbol{p}}_T})^\intercal$ with the normalized gradient h_i:

$$h_i = \nabla_{w_i} \mathcal{L}_{val}(\theta^*) - \frac{\sum_{j=1}^{n} \mathbb{1}_{[y_{jc} = y_{ic}]} \nabla_{w_j} \mathcal{L}_{val}(\theta^*)}{\sum_{j=1}^{n} \mathbb{1}_{[y_{jc} = y_{ic}]}} \tag{10}$$

where y_{ic} is the central pixel label of the segmentation label map $\boldsymbol{y}_i$ and $\mathbb{1}_{y_{jc} = y_{ic}} \in \{0, 1\}$ is an indicator function which is equal to 1 if and only if $y_{jc} = y_{ic}$. Thus, the gradients are normalized for different classes. Another benefit of gradient normalization is that we can guarantee that the probability of transformation which is not sampled in one iteration would remain unchanged.

5) Sampling Normalization: We notice that the optimization process could also be biased towards transformations with high probability. Because the more frequently one transformation is sampled, its probability would be increased more as long as it is more effective than the majority of the transformations in the same batch. As a consequence, it would be likely to be trapped in a local minimum and the probability of any preferable transformation which is frequently sampled by chance could be increased a lot. Therefore, we also normalize h_i with the sampling frequency and obtain $\hat{h}_i$ as:

$$\hat{h}_i = \frac{h_i}{\sum_{v=1}^{n} \mathbb{1}_{[\mathrm{argmax}_j(s_{vj}) = \mathrm{argmax}_j(s_{ij})]}}. \tag{11}$$

E. Learning of Test-Time Data Augmentation

1) The Design of Predefined Transformations: We design the transformation set for TEA with K=84 kinds of deterministic operations including identity along with 41 spatial transformations, 30 intensity transformations, and 12 noise transformations. We summarize the detailed information about those transformations in supplementary material. We initialize $\boldsymbol{p}_V$ referring to the heuristic policies used in nnU-Net [16] which comprises mirroring and 180° rotation in three directions, as shown in Fig. 3.

2) Policy Optimization Based on Reverted Predictions: The optimization of TEA is straightforward. We aim to optimize the function with normal gradient descend:

$$\min_{\boldsymbol{p}_V} \frac{1}{Z} \sum_{k=1}^{Z} \tilde{w}_k \tilde{\mathcal{L}}_{val}(\mathcal{V}_k^{-1}(f_\theta(\mathcal{V}_i(\tilde{\boldsymbol{x}}_i))), \tilde{\boldsymbol{y}}_i), \tag{12}$$

where Z is the number of samples TEA transformations in a batch. We update $\boldsymbol{p}_V$ by choosing the transformations which have the lowest validation loss with the same validation sample:

$$\boldsymbol{p}_V^{t+1} = \boldsymbol{p}_V^t - \gamma \nabla_{\boldsymbol{p}_V} \frac{1}{Z} \sum_{k=1}^{Z} \tilde{w}_k \tilde{\mathcal{L}}_{val}(\mathcal{V}_k^{-1}(f_\theta(\mathcal{V}_k(\tilde{\boldsymbol{x}}_i))), \tilde{\boldsymbol{y}}_i), \tag{13}$$

where γ is the learning rate to update the probability and $\tilde{\mathcal{L}}_{val}$ is the validation loss function for TEA optimization, which can differ from $\mathcal{L}_{val}$.

3) Sampling Normalization: Similar to TRA, we notice that the optimization of $\boldsymbol{p}_V$ would be biased due to the sampling results. We simplify $\frac{1}{Z}\sum_{k=1}^{Z}\tilde{w}_k\tilde{\mathcal{L}}_{val}(\mathcal{V}_k{}^{-1}(f_\theta(\mathcal{V}_k(\tilde{\boldsymbol{x}}_i))), \tilde{\boldsymbol{y}}_i)$ to $\tilde{\mathcal{L}}_{val}(\theta)$, and derive the gradient based on the chain rule:

$$\nabla_{\boldsymbol{p}_V}\tilde{\mathcal{L}}_{val}(\theta) = \sum_{k=1}^{Z}\tilde{h}_k(\frac{\partial \tilde{w}_k}{\partial \boldsymbol{p}_V})^{\mathsf{T}}. \tag{14}$$

Similarly, we normalize the gradients based on sampling frequency and calculate $\tilde{h}_k$ as:

$$\tilde{h}_k = \frac{\nabla_{\tilde{w}_k}\tilde{\mathcal{L}}_{val}(\theta)}{\sum_{v=1}^{Z}\mathbb{1}_{[\text{argmax}_j(s_{vj})=\text{argmax}_j(s_{kj})]}}. \tag{15}$$

F. Inference of Test-Time Data Augmentation

Given unseen test data, we adopt the learned TEA policy to transform the image. Specifically, in order to simplify the inference process, we do TEA at test time with the weighted sum of operations that have the highest z probability, where z is a hyper-parameter indicating the number of operations to be selected for aggregation. We choose z=8 for 3D U-Net while z=4 for DeepMedic. The weight of operation $\mathcal{O}^j$ is set as the corresponding sampling probability which is calculated as $e^{p_j}/\sum_{v=1}^{K}e^{p_v}$.

G. Joint Learning of Training- and Test-Time Data Augmentation

We propose to jointly optimize TRA and TEA, and specifically we optimize $\hat{\boldsymbol{p}}_T$ with the transformed validation data based on $\boldsymbol{p}_V^t$ and rewrite Eq. 3 and 4 as:

$$\min_{\hat{\boldsymbol{p}}_T}\ \frac{1}{M}\sum_{i=1}^{M}\mathcal{L}_{val}(f_{\theta^*}(\mathcal{V}_i(\tilde{\boldsymbol{x}}_i)), \tilde{\boldsymbol{y}}_i) \tag{16}$$

$$s.t.\ \theta^* = \underset{\theta}{\text{argmin}}\ \frac{1}{N}\sum_{i=1}^{N}w_i\mathcal{L}_{train}(f_\theta(\hat{\mathcal{T}}_i(\boldsymbol{x}_i)), \boldsymbol{y}_i). \tag{17}$$

Note that we optimize both $\hat{\boldsymbol{p}}_T$ and $\boldsymbol{p}_V$ in one training iteration. Bridging the optimization process of $\hat{\boldsymbol{p}}_T$ and $\boldsymbol{p}_V$ has two advantages: First, we can reduce the risk of overfitting on the validation data as it is extended with augmented samples. Second, the model can generalize well to the transformations we adopt at test time. The full procedure is summarized in Algorithm 1.

Additional implementation details are provided in the supplementary material. We find that the policies do not need to be updated in every iteration, making training more efficient. In practice, we observe the training time would only increase by about 20% compared to standard training. Typically, when a model takes 4 days to train with an NVIDIA 1080TI GPU for the segmentation task, our method costs 20 hours to find the optimal data augmentation strategies. This is computational efficient than AutoAugment [7] which could takes thousands of hours.

Algorithm 1 Joint Optimization of Class-Specific Training- and Test-Time Data Augmentation in Segmentation

```
Require:
  $\mathcal{D}_T = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{N}$: training data, $\mathcal{D}_V = \{(\tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{y}}_i)\}_{i=1}^{M}$: validation data; $f_\theta(\cdot)$: the segmentation model, $\hat{\mathcal{T}}_i(\cdot)$: TRA which is determined by drawing from class-specific probability $\hat{\boldsymbol{p}}_T$, $\mathcal{V}_i(\cdot)$: TEA which is determined by drawing from probability $\boldsymbol{p}_V$.
  $\alpha$, $\beta$, $\gamma$: learning rate to update $\theta$, $\hat{\boldsymbol{p}}_T$ and $\boldsymbol{p}_V$.
Initialize $\hat{\boldsymbol{p}}_T$, $\boldsymbol{p}_V$ with heuristic policies referring to the ones in DeepMedic [18] or nnU-Net [16].
for each iteration do
    Sample a batch of training data $\mathcal{B}_T = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{n}$ from $\mathcal{D}_T$ and a batch of validation data $\mathcal{B}_V = \{(\tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{y}}_i)\}_{i=1}^{m}$ from $\mathcal{D}_V$.
    for a number of steps do ▷ Note: One step is sufficient in our experiments.
        Sample a set of $\{\hat{\mathcal{T}}_i(\cdot)\}_{i=1}^{n}$ with Gumbel-Softmax distribution parameterized by $\boldsymbol{p}_T{}^j$ based on sample class.
        Sample $\{\mathcal{V}_i(\cdot)\}_{i=1}^{m}$ and $\{\mathcal{V}_k(\cdot)\}_{k=1}^{Z}$ with Gumbel-Softmax distribution parameterized by $\boldsymbol{p}_V$.
        Calculate $\theta^*$ with an optimization step via Eq. 6.
        Optimize $\hat{\boldsymbol{p}}_T$ based on normalized meta-gradients: $\hat{\boldsymbol{p}}_T^{t+1} = \hat{\boldsymbol{p}}_T^{t} - \beta\nabla_{\hat{\boldsymbol{p}}_T^t}\frac{1}{m}\sum_{i=1}^{m}\mathcal{L}_{val}(f_{\theta^*}(\mathcal{V}_i(\tilde{\boldsymbol{x}}_i)), \tilde{\boldsymbol{y}}_i)$. ▷ Learning of TRA.
        Optimize $\boldsymbol{p}_V$ based on normalized gradient via Eq. 13. ▷ Learning of TEA.
    end for
    Update $\theta$ to $\theta^*$. ▷ Training the segmentation model.
end for
```

IV. Experiments, Results, and Discussion

A. Experimental Setup

1) Data Pre-Processing: We normalize all datasets using the pipeline of nnU-Net. Specifically, we adopt case-wise Z-score normalization for magnetic resonance (MR) images, and we normalize computed tomography (CT) images with dataset-wise Z-score normalization based on foreground samples after clipping the Hounsfield units (HU) values from 0.5% to 99.5%.

2) Network Configurations: Our experiments are performed with DeepMedic [18] and a well configured 3D U-Net [16]. We choose cross-entropy (CE) as $\mathcal{L}_{train}$ for DeepMedic and an equal combination of CE and soft Dice similarity coefficient (DSC) for 3D U-Net. We find that the optimal choice of $\mathcal{L}_{val}$ varies for different datasets and summarize the information in supplementary material. We adopt soft DSC for $\tilde{\mathcal{L}}_{val}$ for all the experiments. We choose the batch sizes n and m to be 10. We set the primary patch size as 37×37×37 for all the experiments with DeepMedic and a patch size of 64×64×64 for all the applications with 3D U-Net except prostate segmentation. We choose a patch size of 64×64×32 for prostate segmentation with 3D U-Net because images in this dataset have fewer slices. We train the networks for 1,000 epochs except for kidney and kidney tumor segmentation where we train for 2,000 epochs, as we observed that the networks need more iterations to converge on this task. All the reported results are the average of two runs with different random seeds.

TABLE I
AVERAGE DSC RESULTS OF BOTH DEEPMEDIC AND 3D U-NET FOR DIFFERENT SEGMENTATION TASKS UNDER VARIED SETTINGS USING DIFFERENT DATA AUGMENTATION METHODS. ORGAN$_r$ IS THE AVERAGE PERFORMANCE OF ALL RARE ORGAN CLASSES

Dataset	Heuristic TRA w/o TEA	**Learned Class-Specific** w/o TEA	Heuristic TRA w/ Heuristic TEA	**Joint Learned Class-Specific**
ATLAS [27]	58.5	**60.9** (+2.4)**	61.1	**62.4** (+1.3)*
KiTS [15]	70.0	**74.9** (+4.9)**	75.5	**76.8** (+1.3)*
Organ [22]	79.6	**80.7** (+1.1)*	80.3	**81.1** (+0.8)*
Organ$_r$ [22]	72.8	**74.1** (+1.3)*	73.5	**74.6** (+1.1)*
Prostate [2], [23]	71.6	**71.7** (+0.1)~	74.0	**76.4** (+2.4)*

$^*p$-value < 0.05; $^{**}p$-value < 0.01; $^{\sim}p$-value ≥ 0.05 (compared to Heuristic TRA w/o TEA or Heuristic TRA w/ Heuristic TEA)

3) Brain Stroke Lesion Segmentation: We firstly evaluate the proposed method with binary brain stroke lesion segmentation using the dataset of Anatomical Tracings of Lesions After Stroke (ATLAS) [27]. The images have a voxel spacing of 1.0×1.0×1.0 mm. With a total of 220 T1-weighted MR images, we randomly select 73 (50%) or 145 (100%) for training, 31 for validation, and 44 for test.

4) Kidney and Kidney Tumor Segmentation: Secondly, we evaluate the proposed method with kidney and kidney tumor segmentation using the training dataset of Kidney Tumor Segmentation Challenge (KiTS) [15] which contains 210 CT images. We resample all images to voxel spacing of 1.6×1.6×3.2 mm. We randomly select 70 (50%) or 140 (100%) for training, 28 for validation, and 42 for test. We omit the segmentation results of kidney as find most methods perform well (DSC > 95.0) on the task of kidney segmentation.

5) Abdominal Organ Segmentation: We also evaluate the proposed method with the task of abdominal organ segmentation [22] which contains 14 classes including spleen (SP), right kidney (RK), left kidney (LK), gallbladder (GB), esophagus (E), liver (LIV), stomach (STO), aorta (AO), inferior vena cava (IVC), portal vein and splenic vein (V), pancreas (PA), right adrenal gland (RA) and left adrenal gland (LA). We resample all images to a voxel spacing of 1.6×1.6×3.2 mm. We randomly select 20 for training, 4 for validation, and 6 for test.

6) Cross-Site Prostate Segmentation: Additionally, we utilize our method to align training data and validation data of prostate segmentation from different domains [30]. Specifically, we utilize 30 T2-weighted MR images from site A [2] which were collected with 1.5T Philips MRI machine with endorectal coil and 19 T2-weighted MR images from site B [23] which were collected with 3T Siemens MRI machines without endorectal coil. We resample all the images to a voxel spacing of 0.8×0.8×1.5 mm. We investigate the scenario where the target domain (site B) has limited labeled data. We select 20 cases from site A for training and 6 cases for test. We select 1 case from site B for validation and use 18 cases for testing. Note that for cross-site prostate segmentation, we report results with models trained with both training data and validation data as this serves as a fairer baseline compared to using training data only.

B. Compared Methods

1) Heuristic: We compare with the heuristic TRA and TEA which are set as the default configurations in DeepMedic [18] and nnU-Net [16]. We also report a few results based on models trained using both the training and validation data with heuristic TRA.

2) Learned TRA: We compare with methods that adopt the data augmentation policies based on the validation performance without considering class dependency [7], [25], [28].

3) TRA With Different Transformation Magnitudes: We also compare with RandAugment [8], which only changes the transformation magnitudes based on grid searching. Specifically, we keep the data augmentation probability and replace the operations of the same type with different magnitudes, yielding RandAugment-S, RandAugment-M and RandAugment-L. We summarize the results with RandAugment in supplementary material.

4) Learned TEA: We compare with methods which optimize TEA based on a pretrained segmentation model [20], [33]. Specifically, after training the model with proposed TRA, we refine TEA as described in Section III-E.

C. Quantitative Results

Taking the manual segmentation as the ground truth, we calculate evaluation metrics including DSC, sensitivity (SEN), precision (PRC), 95% Hausdorff distance (HD) (mm). We calculate the mean DSC results of different models under different settings for different datasets in Table I. We summarize more detailed results in Table II, III, IV and V, separately. In order to assess the overall segmentation performance of different methods, we rank the methods according to different metrics under the same experiment setting and report the average rank (AVG rank) of the four metrics. The learned probability distributions over augmentations for brain lesion segmentation based on different models with 100% ATLAS training data are summarized in Fig. 3. As shown for TRA policies, the darkness of different pie chart segments stands for the magnitudes of the operations. For example, the lightest grey segments refer to the operation without any transformations and the darkest (black) segments represent the operations with large transformations. We summarize all the learned policies under different settings in supplementary material.

1) The Effectiveness of Class-Specific TRA: Heuristic TRAs, which were tuned based on varied segmentation tasks [16], [18], significantly help improve the segmentation performance in all cases compared with models trained without TRA. This indicates that TRA is vital for medical image segmentation as limited training data and class imbalance can easily lead to model overfitting [26].

Learned TRA, which is optimized with validation data, can provide application-specific policies and is more effective than heuristic TRA in most cases. We find that the models trained with learned TRA can even outperform the ones trained with heuristic TRA that use both training and validation data, as shown in Table II. This may indicate that it will

TABLE II

EVALUATION OF BRAIN STROKE LESION SEGMENTATION ON ATLAS BASED ON DIFFERENT NETWORK ARCHITECTURES WITH DIFFERENT AMOUNTS OF TRAINING DATA USING DIFFERENT DATA AUGMENTATION METHODS. BEST AND SECOND BEST RESULTS ARE IN **BOLD**, WITH BEST ALSO $\underline{\textbf{UNDERLINED}}$

Model	Training-time augmentation	Test-time augmentation	50% training data				100% training data				AVG Rank
			DSC ↑	SEN ↑	PRC ↑	HD ↓	DSC ↑	SEN ↑	PRC ↑	HD ↓	↓
DeepMedic [18]	None	None	51.7	50.6	65.0	$\underline{\mathbf{20.4}}$	55.2	57.9	63.1	**24.3**	4.0
	Heuristic [18]	None	58.2	62.3	65.3	26.8	58.9	**65.5**	64.9	32.6	3.6
	Heuristic† [18]	None	**59.1**	$\underline{\mathbf{64.7}}$	65.0	30.3	**59.5**	63.0	$\underline{\mathbf{67.5}}$	29.3	2.9
	Learned [7], [25], [28]	None	$\underline{\mathbf{59.5}}$	62.7	$\underline{\mathbf{68.5}}$	**25.1**	58.4	62.9	67.0	26.2	**2.6**
	Learned Class-Specific	None	$\underline{\mathbf{59.5}}$ (+1.3)$^{\sim}$	**62.9**	**68.1**	26.1	$\underline{\mathbf{60.2}}$ (+1.3)*	$\underline{\mathbf{66.1}}$	**67.3**	$\underline{\mathbf{22.7}}$	$\underline{\mathbf{1.6}}$
	Heuristic [18]	Heuristic [16]	60.1	63.7	68.7	$\underline{\mathbf{23.5}}$	60.6	65.7	67.7	27.6	3.5
	Learned Class-Specific	Heuristic [16]	**61.1**	$\underline{\mathbf{64.4}}$	70.2	25.6	**61.6**	$\underline{\mathbf{66.2}}$	68.6	**24.4**	2.4
	Learned Class-Specific	Learned [20], [33]	$\underline{\mathbf{61.3}}$	64.2	**70.8**	25.3	**61.6**	**66.0**	**69.7**	$\underline{\mathbf{23.5}}$	**2.0**
	Joint Learned Class-Specific		$\underline{\mathbf{61.3}}$ (+1.2)$^{\sim}$	**64.3**	$\underline{\mathbf{71.2}}$	**24.5**	$\underline{\mathbf{61.9}}$ (+1.3)$^{\sim}$	64.5	$\underline{\mathbf{71.7}}$	25.0	$\underline{\mathbf{1.9}}$
3D U-Net [5]	None	None	54.6	56.3	$\underline{\mathbf{67.2}}$	$\underline{\mathbf{32.6}}$	56.7	58.8	$\underline{\mathbf{69.8}}$	$\underline{\mathbf{23.0}}$	**3.0**
	Heuristic [16]	None	58.4	66.9	61.4	39.0	58.9	67.9	60.6	44.1	3.4
	Heuristic† [16]	None	58.8	**67.8**	59.8	52.2	58.3	$\underline{\mathbf{69.5}}$	56.2	58.6	3.8
	Learned [7], [25], [28]	None	**59.3**	66.6	61.1	40.9	**59.5**	**69.2**	61.1	48.2	3.1
	Learned Class-Specific	None	$\underline{\mathbf{62.0}}$ (+3.6)**	$\underline{\mathbf{68.8}}$	**66.2**	**37.8**	$\underline{\mathbf{62.1}}$ (+3.2)**	68.9	**65.8**	**34.8**	$\underline{\mathbf{1.8}}$
	Heuristic [16]	Heuristic [16]	61.7	**67.0**	69.6	22.0	62.3	$\underline{\mathbf{68.6}}$	68.4	31.4	3.1
	Learned Class-Specific	Heuristic [16]	61.8	66.4	$\underline{\mathbf{70.2}}$	$\underline{\mathbf{20.3}}$	**63.9**	68.4	$\underline{\mathbf{72.2}}$	$\underline{\mathbf{23.5}}$	**2.1**
	Learned Class-Specific	Learned [20], [33]	**62.2**	66.9	**69.9**	**20.5**	**63.9**	**68.5**	72.0	**24.5**	**2.3**
	Joint Learned Class-Specific		$\underline{\mathbf{62.3}}$ (+0.6)$^{\sim}$	$\underline{\mathbf{67.4}}$	69.2	28.9	$\underline{\mathbf{64.0}}$ (+1.7)$^{\sim}$	**68.5**	**72.1**	**24.5**	$\underline{\mathbf{2.1}}$

$^*p$-value < 0.05; $^{**}p$-value < 0.01; $^{\sim}p$-value ≥ 0.05 (compared to Heuristic TRA w/o TEA or Heuristic TRA w/ Heuristic TEA)
†We train these models with both training and validation data.

TABLE III

EVALUATION OF KIDNEY TUMOR SEGMENTATION BASED ON DIFFERENT NETWORK ARCHITECTURES WITH DIFFERENT AMOUNTS OF TRAINING DATA USING DIFFERENT DATA AUGMENTATION METHODS. BEST AND SECOND BEST RESULTS ARE IN **BOLD**, WITH BEST ALSO $\underline{\textbf{UNDERLINED}}$

Model	Training-time data augmentation	Test-time data augmentation	50%				100%				AVG Rank
			DSC ↑	SEN ↑	PRC ↑	HD ↓	DSC ↑	SEN ↑	PRC ↑	HD ↓	↓
DeepMedic [18]	None	None	40.1	35.4	56.2	93.0	51.1	50.0	62.0	$\underline{\mathbf{72.8}}$	3.6
	Heuristic [18]	None	66.6	69.3	72.4	76.3	**69.5**	77.2	**69.8**	**76.3**	2.6
	Learned [7], [25], [28]	None	**69.1**	**71.3**	**75.1**	$\underline{\mathbf{61.8}}$	**69.5**	$\underline{\mathbf{79.3}}$	67.6	89.4	**2.1**
	Learned Class-Specific	None	$\underline{\mathbf{71.6}}$ (+5.0)**	$\underline{\mathbf{72.8}}$	$\underline{\mathbf{76.8}}$	**66.5**	$\underline{\mathbf{71.2}}$ (+1.7)$^{\sim}$	**78.1**	$\underline{\mathbf{70.4}}$	88.3	$\underline{\mathbf{1.5}}$
	Heuristic [18]	Heuristic [16]	70.5	70.5	**78.5**	58.7	72.9	77.9	$\underline{\mathbf{74.1}}$	62.5	3.3
	Learned Class-Specific	Heuristic [16]	72.5	73.4	78.3	**48.1**	73.1	**79.5**	73.2	$\underline{\mathbf{57.2}}$	2.8
	Learned Class-Specific	Learned [20], [33]	**72.8**	$\underline{\mathbf{73.6}}$	**78.5**	$\underline{\mathbf{47.9}}$	**73.3**	79.3	73.6	**60.5**	**2.0**
	Joint Learned Class-Specific		$\underline{\mathbf{73.3}}$ (+2.8)**	**73.5**	$\underline{\mathbf{79.7}}$	48.4	$\underline{\mathbf{74.1}}$ (+1.2)$^{\sim}$	$\underline{\mathbf{79.6}}$	**74.0**	71.7	$\underline{\mathbf{1.9}}$
3D U-Net [5]	None	None	43.5	39.5	60.9	104.6	60.3	57.1	71.6	78.4	4.0
	Heuristic [16]	None	76.6	80.2	**77.4**	$\underline{\mathbf{40.6}}$	**77.6**	82.1	**77.6**	59.5	2.5
	Learned [7], [25], [28]	None	**76.7**	**82.0**	76.1	55.7	$\underline{\mathbf{78.5}}$	$\underline{\mathbf{84.2}}$	77.4	**50.6**	**2.1**
	Learned Class-Specific	None	$\underline{\mathbf{78.4}}$ (+1.8)$^{\sim}$	$\underline{\mathbf{82.2}}$	$\underline{\mathbf{78.0}}$	**47.2**	$\underline{\mathbf{78.5}}$ (+0.9)$^{\sim}$	**83.2**	$\underline{\mathbf{78.0}}$	$\underline{\mathbf{48.8}}$	$\underline{\mathbf{1.3}}$
	Heuristic [16]	Heuristic [16]	**78.8**	**82.1**	79.4	$\underline{\mathbf{37.2}}$	79.7	83.3	79.5	45.5	3.1
	Learned Class-Specific	Heuristic [16]	78.7	81.7	**79.6**	**42.0**	**80.5**	**84.0**	80.2	42.3	2.6
	Learned Class-Specific	Learned [20], [33]	**78.8**	81.7	**79.6**	**42.0**	$\underline{\mathbf{80.6}}$	$\underline{\mathbf{84.1}}$	**80.4**	$\underline{\mathbf{38.3}}$	$\underline{\mathbf{1.8}}$
	Joint Learned Class-Specific		$\underline{\mathbf{79.3}}$ (+0.5)$^{\sim}$	$\underline{\mathbf{82.2}}$	$\underline{\mathbf{79.7}}$	45.4	80.4 (+0.7)$^{\sim}$	83.5	$\underline{\mathbf{80.6}}$	**38.6**	**2.0**

$^*p$-value < 0.05; $^{**}p$-value < 0.01; $^{\sim}p$-value ≥ 0.05 (compared to Heuristic TRA w/o TEA or Heuristic TRA w/ Heuristic TEA)

be more effective to increase the heterogeneity within the training data by adopting application-specific TRA than adding a small amount of training data. We find RandAugment with specific magnitude could be more effective than the learned one in some cases. Specifically, RandAugment-L is better than the learned ones for kidney tumor segmentation under specific setting, as shown in Table III. This might indicate that learned TRA is prone to overfitting the validation data and the optimized policies are not guaranteed to be optimal for unseen test data, as also found in [8].

In contrast, class-specific TRA can better model the heterogeneity of the real data by taking class imbalance into account, and thus overfit less and perform better on unseen test data than alternative methods. We argue that class-specific TRA is important as it concerns the imbalanced nature of the segmentation datasets and directly regularizes the training data in an implicit way. As shown in Fig. 3, compared with heuristic TRA, the learned policies tend to generate larger transformations for foreground samples while adopting smaller transformations to background samples. In segmentation, foreground classes are typically underrepresented and a learned baseline model would be biased towards the majority class. As a result, the model would map the foreground samples near the decision boundary and cause false negatives, as shown in [26]. Class-specific TRA can mitigate the class imbalance problem by inducing larger variance within the foreground samples, making the model learn a better decision boundary, consistently leading to better segmentation results with higher sensitivity. Particularly, we find class-specific TRA would improve the segmentation performance of rare classes more significantly (c.f. Table IV) as it can enhance the rare class representation by increasing the heterogeneity of foreground sample variation. We also find that the probabilities of spatial transformations change more significantly compared to

TABLE IV
EVALUATION OF ABDOMINAL ORGAN SEGMENTATION BASED ON DIFFERENT NETWORK ARCHITECTURES USING RANDOM DATA AUGMENTATION METHODS. BEST AND SECOND BEST RESULTS ARE IN **BOLD**, WITH THE BEST ALSO **UNDERLINED**. AVG$_r$ IS THE AVERAGE PERFORMANCE OF ALL RARE CLASSES INCLUDING GB, E, AO, IVC, V, PA, RA AND LA

Model	Training-time data augmentation	Test-time data augmentation	DSC														
			SP	RK	LK	GB	E	LIV	STO	AO	IVC	V	PA	RA	LA	AVG ↑	AVG$_r$ ↑
DeepMedic [18]	None	None	87.7	89.5	91.6	48.7	71.9	94.8	76.7	83.8	82.5	62.0	51.9	49.4	54.3	72.7	63.1
	Heuristic [18]	None	90.9	90.5	90.8	59.8	75.1	93.0	77.9	84.3	84.9	69.8	65.1	65.4	66.4	**78.0**	71.3
	Learned [7], [25], [28]	None	93.4	90.8	93.1	63.6	76.0	94.1	81.7	86.2	84.2	71.5	66.7	68.2	64.6	**79.5**	**72.6**
	Learned Class-Specific	None	91.8	89.7	91.0	65.7	76.6	93.9	79.6	85.2	85.2	69.9	68.9	69.7	66.8	**79.5** (+1.5)**	**73.5** (+2.2)**
	Heuristic [18]	Heuristic† [16]	91.4	90.6	92.2	60.3	78.6	93.1	78.1	84.4	85.3	70.5	66.4	65.4	69.2	78.9	72.5
	Learned Class-Specific	Heuristic† [16]	92.1	90.2	92.1	65.3	77.3	94.0	80.8	86.6	86.1	69.2	68.5	66.7	64.9	79.5	**73.1**
	Learned Class-Specific	Learned [20], [33]	92.1	89.9	91.5	65.6	77.5	94.0	80.4	86.3	86.0	69.7	68.8	68.7	66.6	**79.8**	**73.7**
	Joint Learned Class-Specific		92.9	91.5	93.3	63.7	77.6	93.4	80.0	86.1	85.1	69.9	68.5	66.3	66.8	**79.6** (+0.7)$^{\sim}$	73.0 (+0.5)$^{\sim}$
3D U-Net [5]	None	None	89.2	91.6	92.5	29.9	69.8	95.2	86.9	88.1	86.1	69.3	60.0	54.8	61.7	75.0	65.0
	Heuristic [16]	None	94.9	92.7	92.6	55.4	75.9	96.0	89.3	91.6	88.0	72.9	75.4	67.7	67.8	**81.5**	**74.3**
	Learned [7], [25], [28]	None	94.6	92.9	92.8	59.7	76.6	96.1	90.0	91.8	88.2	73.1	73.3	64.4	66.0	**81.5**	74.1
	Learned Class-Specific	None	95.0	93.4	92.9	59.1	76.0	96.2	88.9	91.9	88.0	73.8	74.3	61.9	72.6	**81.8** (+0.3)$^{\sim}$	**74.7** (+0.4)$^{\sim}$
	Heuristic [16]	Heuristic [16]	95.3	93.6	92.8	58.4	73.2	96.2	90.4	91.8	88.6	74.6	75.2	64.5	69.0	81.8	74.4
	Learned Class-Specific	Heuristic [16]	95.3	93.4	93.0	52.7	77.5	96.1	89.9	92.0	88.2	76.2	74.7	64.8	72.8	82.0	74.9
	Learned Class-Specific	Learned [20], [33]	95.3	93.3	92.9	56.1	77.6	96.2	89.9	92.0	88.3	75.7	74.7	64.8	72.6	**82.2**	**75.2**
	Joint Learned Class-Specific		94.4	93.5	93.1	61.8	80.8	96.0	89.3	90.9	87.7	72.7	77.6	66.7	71.4	**82.8** (+1.0)$^{\sim}$	**76.2** (+1.8)$^{\sim}$

$^{*}p$-value < 0.05; $^{**}p$-value < 0.01; $^{\sim}p$-value ≥ 0.05 (compared to Heuristic TRA w/o TEA or Heuristic TRA w/ Heuristic TEA)
† We adopt a heuristic TEA policy with larger probability of identity transformation here because typical ones would decrease the performance.

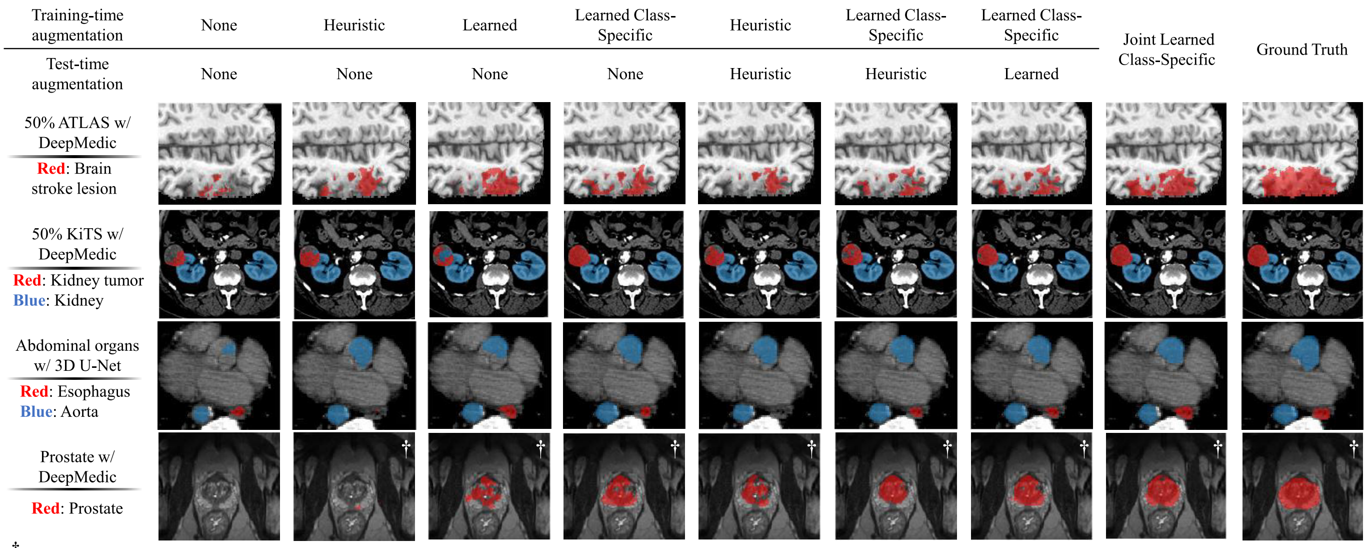
TABLE V
EVALUATION OF CROSS-SITE PROSTATE SEGMENTATION BASED ON DIFFERENT NETWORK ARCHITECTURES USING DIFFERENT DATA AUGMENTATION METHODS. BEST AND SECOND BEST RESULTS ARE IN **BOLD**, WITH THE BEST ALSO **UNDERLINED**

Model	Site A Training-time data augmentation	Site A/B Test-time data augmentation	Site B				Site A				AVG
			DSC ↑	SEN ↑	PRC ↑	HD ↓	DSC ↑	SEN ↑	PRC ↑	HD ↓	Rank ↓
DeepMedic [18]	None	None	14.9	11.6	45.3	42.6	82.4	77.1	90.7	6.7	6.0
	Heuristic [18]	None	46.4	43.2	59.4	26.9	88.0	**85.5**	91.4	4.8	5.0
	Heuristic Fine-Tuning† [18]	None	56.7	46.4	**77.5**	**9.4**	27.6	20.0	81.4	18.9	**2.5**
	Heuristic‡ [18]	None	**69.3**	**67.2**	73.5	**15.1**	**88.1**	84.8	**92.3**	**4.5**	**2.5**
	Learned‡ [7], [25], [28]	None	65.8	62.8	75.1	21.9	87.5	83.4	**92.6**	**4.6**	3.0
	Learned Class-Specific‡	None	**70.0** (+0.7)$^{\sim}$	**68.0**	**75.9**	18.7	**88.2**	**85.6**	91.5	4.7	**1.8**
	Heuristic‡ [18]	Heuristic [16]	69.4	66.3	76.5	**8.0**	88.2	84.6	**92.6**	4.6	3.3
	Learned Class-Specific‡	Heuristic [16]	69.9	66.3	**80.0**	**8.0**	**88.8**	**86.1**	**92.4**	**4.4**	**2.3**
	Learned Class-Specific‡	Learned [20], [33]	**70.2**	**67.7**	**77.6**	15.3	**88.5**	**86.0**	91.7	4.6	2.5
	Joint Learned Class-Specific‡		**72.8** (+3.4)**	**71.0**	76.6	**7.9**	88.2	85.4	91.8	**4.5**	**1.5**
3D U-Net [5]	None	None	57.2	52.8	**69.7**	**13.7**	87.1	84.2	**91.1**	**5.3**	4.0
	Heuristic [16]	None	63.3	89.3	50.9	64.0	**89.4**	88.4	**90.8**	16.0	5.0
	Heuristic Fine-Tuning† [16]	None	68.7	63.9	**84.1**	**9.8**	55.6	48.6	77.2	25.1	**2.8**
	Heuristic‡ [16]	None	**73.9**	88.4	65.6	60.1	**89.4**	88.9	90.0	18.7	3.8
	Learned‡ [7], [25], [28]	None	**76.5**	**89.8**	67.9	44.2	87.0	**89.2**	85.3	26.9	**2.5**
	Learned Class-Specific‡	None	73.2 (-0.7)$^{\sim}$	**90.1**	63.4	41.5	**87.7**	**89.6**	86.6	**15.4**	3.0
	Heuristic‡ [16]	Heuristic [16]	78.7	**90.5**	71.4	28.0	**89.7**	89.1	**90.6**	**12.1**	2.8
	Learned‡ [7], [25], [28]	Heuristic [16]	78.5	89.2	71.6	31.1	88.9	89.2	89.0	12.8	3.5
	Learned‡ [7], [25], [28]	Learned [20], [33]	**79.6**	**89.9**	**72.7**	**23.6**	89.0	**89.3**	89.1	12.3	**2.0**
	Joint Learned‡		**80.0** (+1.3)$^{\sim}$	88.2	**74.1**	**18.9**	**90.0**	**89.3**	**91.0**	**11.8**	**1.8**

$^{*}p$-value < 0.05; $^{**}p$-value < 0.01; $^{\sim}p$-value ≥ 0.05 (compared to Heuristic‡ TRA w/o TEA or Heuristic‡ TRA w/ Heuristic TEA)
† We pretrain these models with training data from site A and fine-tune with validation data from site B.
‡ We train these models with both training data from site A and validation data from site B.

intensity transformations. This might indicate that spatial transformations are more effective in increasing the heterogeneity within training data. We validate our methods for prostate segmentation under domain shifts where the training and test data is collected under different conditions. We find directly fine-tuning the segmentation models with limited target data provides worse results than training with data from both domains. We report the segmentation results of both site B and site A with cross-site prostate segmentation in Table V. Although the learned data augmentation is optimized based on the validation data from site B (target domain), the models can still generalize well on site A. In addition, as we show in supplementary material, we find that our method can help the models generalize better on unseen test domains which are different from either site A or site B. This indicates that our method is robust to domain shifts and can be a safe choice to calibrate the segmentation performance of different domains within multi-domain learning.

2) The Effectiveness of Joint Optimization: We find heuristic TEA can help the pretrained models produce better overall segmentation results with higher precision. This is because the ensemble of multiple predictions can reduce false positives as the models are unlikely to produce the same kind of false positives with all the transformed images. However, when



†We train these models with both training and validation data.

Fig. 4. Visualization of different datasets and segmentation results with different data augmentation methods. The proposed data augmentation framework can help the model produce overall better segmentation results with higher sensitivity. Best viewed in color.

TRA is optimized based on validation data without TEA, heuristic TEA might not work well as the model may overfit to the original data distribution and thus fail to generalize to the transformed data. Specifically, we observe that heuristic TEA would decrease the model performance for 3D U-Net trained with 50% ATLAS training data (-0.3 in terms of DSC, c.f. Table II) and Deepmedic trained for prostate segmentation (-0.1 in terms of DSC, c.f. Table V) using learned class-specific TRA. In comparison, learned TEA can refine the transformations to fit the pretrained models and improve the results for most cases.

However, learned TEA alone does not affect model training and cannot change the results significantly compared to heuristic TEA. In contrast, our method optimizes TRA based on TEA along the training process, jointly aligning the data distributions resulting in larger overlaps. For example, as illustrated in Fig. 3(a), the learned TEA policy would increase the probability of flipping in sagittal planes for DeepMedic trained with 100% ATLAS training data. It might be because the left and right hemispheres of human brains are generally symmetric. Correspondingly, TRA would tune the training data distribution with more samples flipped in the sagittal planes. In this way, the segmentation models not only have lower risks of making the same false positives but also generalize better on varied transformed samples. As a result, we find that the joint optimization further boosts the segmentation performance by achieving higher precision and sensitivity.

We argue that the joint optimization is crucial for data augmentation as it explicitly aligns the training and test-time conditions. Otherwise, the model may get stuck into a local minimum where we cannot find effective test-time transformations to fit the training data distribution. For example, we find that when compared with segmentation without TEA, learned TEA brings limited improvements for 3D U-Net trained with 50% KiTS training data (0.4 in terms of DSC, c.f. Table III) and DeepMedic trained for prostate segmentation (0.2 for site B in terms of DSC, c.f. Table V). This indicates that the predictions on most chosen transformations cannot contribute much to the results on top of the predictions of the original test images. In contrast, the joint optimization leverages the varied test-time transformations and improve the segmentation (0.9 and 2.8 separately in terms of DSC).

We notice that the learned TEA policies would generally prefer the original images (identity). In addition, the transformations which are not included in heuristic TEA are hardly useful. These findings indicate that we may not need to apply large transformations to the test data to improve generalization.

We visualize some segmentation results in Fig. 4. Similar to the findings in a previous study [26], the model trained with imbalanced dataset would be prone to undersegment the foreground samples as a result of overfitting under class imbalance. Our class-specific TRA model can significantly reduce false negatives and improve the sensitivity of segmentation results. We observe heuristic TEA could cause undersegmentation while the joint optimization can further help the model improve segmentation performance by identifying more foreground samples. We further validate our methods with cardiac segmentation in MR images in supplementary material to prove that our methods can work well with anisotropic images under domain shifts.

D. Limitations

Our data augmentation algorithm aims to optimize the sampling distributions for TRA and TEA, and thus, automatically adapt data augmentation policies to given task. However, it might not be very effective when the predefined policies are already nearly optimal. For example, we observe that our methods do not bring much improvements for kidney tumor segmentation based on 3D U-Net when trained with 100% training data. This is possibly because that the predefined policies were already optimized given it is the winning solution for the challenge.

We notice that class-specific TRA could be less effective with 3D U-Net on prostate segmentation. This may be due to the sampled patches always containing foreground, as the image size of this dataset is relatively small, and the structures-of-interest are relatively large. In this case, the optimization could be misled by the class-specific constraints. In practice, this could be alleviated by adopting a smaller patch size, and some investigations can be found in the supplementary material. Moreover, we could consider to restrict the regions of loss calculation to make our algorithms compatible with similar cases where the patch size is large up to the image size. This would need to be explored in future work.

The joint optimization of TRA and TEA will not be effective when TEA decreases the segmentation performance. For example, we find that the joint optimization cannot bring much improvements for DeepMedic with abdominal organ segmentation where most transformations for TEA do not seem to help much and the augmented validation data would improperly influence the TRA optimization. Therefore, we suggest validating the effectiveness of TEA before adopting the joint optimization.

Although we the proposed method can consistently improve the segmentation performance under varied scenarios, we observe that not all the results show statistical significance when compared to heuristic baselines. This might be due to the small size of the test set. We show that our methods show significant improvements when more test data is available (c.f. Table I). We observe that distance based metrics such as HD is unstable for the evaluation of imbalanced regions-of-interest (ROIs) because small false positive predictions could largely increase those metrics. After eliminating the false positive predictions with component-based post-processing, our method can always perform better in terms of both DSC and HD, as we demonstrate in supplementary material.

We present and validate our method in the context of medical image segmentation. We think that it has the potential to be extended to long-tailed image classification tasks where different classes have different properties and TEA is also important for better generalization. We show some initial experiments in supplementary material and will leave the in-depth investigation for future works.

V. Conclusion

We presented a general data augmentation framework for medical image segmentation. Compared with current solutions, our method aims to bridge the gap between training and test data distributions by class-specific TRA and joint optimization of TRA and TEA. We observe promising improvements in various tasks and models, making the proposed framework an attractive alternative to heuristic data augmentation strategies. We believe that the learned policies can provide valuable insights for practitioners to inform dynamic data collection and future designs of image transformations for data augmentation.

References

[1] Y. Bengio, “Gradient-based optimization of hyperparameters,” *Neural Comput.*, vol. 12, no. 8, pp. 1889–1900, Aug. 2000.

[2] N. Bloch et al., “NCI-ISBI 2013 challenge: Automated segmentation of prostate structures,” *Cancer Imag. Arch.*, vol. 370, no. 6, pp. 1–5, 2015.

[3] K. Chaitanya et al., “Semi-supervised task-driven data augmentation for medical image segmentation,” *Med. Image Anal.*, vol. 68, Feb. 2021, Art. no. 101934.

[4] C. Chen et al., “Realistic adversarial data augmentation for mr image segmentation,” in *Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Intervent*. Cham, Switzerland: Springer, 2020, pp. 667–677.

[5] O. Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger, “3D U-Net: Learning dense volumetric segmentation from sparse annotation,” in *Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Intervent*. Cham, Switzerland: Springer, 2016, pp. 424–432.

[6] D. C. Cireşan, A. Giusti, L. M. Gambardella, and J. Schmidhuber, “Mitosis detection in breast cancer histology images with deep neural networks,” in *Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Intervent*. Cham, Switzerland: Springer, 2013, pp. 411–418.

[7] E. D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, and Q. V. Le, “AutoAugment: Learning augmentation policies from data,” 2018, *arXiv:1805.09501*.

[8] E. D. Cubuk, B. Zoph, J. Shlens, and Q. V. Le, “RandAugment: Practical automated data augmentation with a reduced search space,” 2019, *arXiv:1909.13719*.

[9] T. DeVries and G. W. Taylor, “Improved regularization of convolutional neural networks with cutout,” 2017, *arXiv:1708.04552*.

[10] C. Finn, P. Abbeel, and S. Levine, “Model-agnostic meta-learning for fast adaptation of deep networks,” in *Proc. 34th Int. Conf. Mach. Learn.*, vol. 70, Aug. 2017, pp. 1126–1135.

[11] L. Franceschi, P. Frasconi, S. Salzo, R. Grazzi, and M. Pontil, “Bilevel programming for hyperparameter optimization and meta-learning,” in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 1568–1577.

[12] I. J. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and harnessing adversarial examples,” 2014, *arXiv:1412.6572*.

[13] A. Gupta, S. Venkatesh, S. Chopra, and C. Ledig, “Generative image translation for data augmentation of bone lesion pathology,” 2019, *arXiv:1902.02248*.

[14] Y. He, A. Carass, L. Zuo, B. E. Dewey, and J. L. Prince, “Autoencoder based self-supervised test-time adaptation for medical image analysis,” *Med. Image Anal.*, vol. 72, Aug. 2021, Art. no. 102136.

[15] N. Heller et al., “The KiTS19 challenge data: 300 kidney tumor cases with clinical context, CT semantic segmentations, and surgical outcomes,” 2019, *arXiv:1904.00445*.

[16] F. Isensee, P. F. Jaeger, S. A. A. Kohl, J. Petersen, and K. H. Maier-Hein, “NnU-Net: A self-configuring method for deep learning-based biomedical image segmentation,” *Nature Methods*, vol. 18, no. 2, pp. 203–211, Feb. 2021.

[17] E. Jang, S. Gu, and B. Poole, “Categorical reparameterization with Gumbel–Softmax,” 2016, *arXiv:1611.01144*.

[18] K. Kamnitsas et al., “Efficient multi-scale 3D CNN with fully connected CRF for accurate brain lesion segmentation,” *Med. Image Anal.*, vol. 36, pp. 61–78, Feb. 2017.

[19] N. Karani, E. Erdil, K. Chaitanya, and E. Konukoglu, “Test-time adaptable neural networks for robust medical image segmentation,” *Med. Image Anal.*, vol. 68, Feb. 2021, Art. no. 101907.

[20] I. Kim, Y. Kim, and S. Kim, “Learning loss for test-time augmentation,” 2020, *arXiv:2010.11422*.

[21] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Adv. neural Inf. Process. Syst.*, pp. 1097–1105, 2012.

[22] B. A. Landman, Z. Xu, J. E. Igelsias, M. Styner, T. R. Langerak, and A. Klein. *2015 MICCAI Multi-Atlas Labeling Beyond the Cranial Vault Workshop and Challenge*. Accessed: Dec. 2020, doi: 10.7303/syn3193805. [Online]. Available: https://www.synapse.org/#!Synapse:syn3193805

[23] G. Lemaître, R. Martí, J. Freixenet, J. C. Vilanova, P. M. Walker, and F. Meriaudeau, "Computer-aided detection and diagnosis for prostate cancer based on mono and multi-parametric MRI: A review," *Comput. Biol. Med.*, vol. 60, pp. 8–31, May 2015.

[24] L. Li, K. Jamieson, G. DeSalvo, A. Rostamizadeh, and A. Talwalkar, "Hyperband: A novel bandit-based approach to hyperparameter optimization," *J. Mach. Learn. Res.*, vol. 18, no. 1, pp. 6765–6816, 2017.

[25] Y. Li, G. Hu, Y. Wang, T. Hospedales, N. M. Robertson, and Y. Yang, "DADA: Differentiable automatic data augmentation," 2020, *arXiv:2003.03780*.

[26] Z. Li, K. Kamnitsas, and B. Glocker, "Analyzing overfitting under class imbalance in neural networks for image segmentation," *IEEE Trans. Med. Imag.*, vol. 40, no. 3, pp. 1065–1077, Mar. 2021.

[27] S.-L. Liew et al., "A large, open source dataset of stroke anatomical brain images and manual lesion segmentations," *Sci. Data*, vol. 5, no. 1, Feb. 2018, Art. no. 180011.

[28] S. Lim, I. Kim, T. Kim, C. Kim, and S. Kim, "Fast AutoAugment," 2019, *arXiv:1905.00397*.

[29] H. Liu, K. Simonyan, and Y. Yang, "DARTS: Differentiable architecture search," in *Proc. Int. Conf. Learn. Represent.*, 2019. [Online]. Available: https://openreview.net/forum?id=S1eYHoC5FX

[30] Q. Liu, Q. Dou, and P. A. Heng, "Shape-aware meta-learning for generalizing prostate MRI segmentation to unseen domains," in *Proc. Int. Conf. Med. Image Comput. Comput. Assist. Intervent (MICCAI)*, 2020, pp. 475–485.

[31] M. Paschali et al., "Data augmentation with manifold exploring geometric transformations for increased performance and robustness," 2019, *arXiv:1901.04420*.

[32] F. Pedregosa, "Hyperparameter optimization with approximate gradient," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 737–746.

[33] D. Shanmugam, D. Blalock, G. Balakrishnan, and J. Guttag, "Better aggregation in test-time augmentation," 2020, *arXiv:2011.11156*.

[34] D. A. van Dyk and X.-L. Meng, "The art of data augmentation," *J. Comput. Graph. Statist.*, vol. 10, no. 1, pp. 1–50, 2001.

[35] G. Wang, W. Li, M. Aertsen, J. Deprest, S. Ourselin, and T. Vercauteren, "Aleatoric uncertainty estimation with test-time augmentation for medical image segmentation with convolutional neural networks," *Neurocomputing*, vol. 338, pp. 34–45, Apr. 2019.

[36] J. Xu, M. Li, and Z. Zhu, "Automatic data augmentation for 3D medical image segmentation," in *Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Intervent*. Cham, Switzerland: Springer, 2020, pp. 378–387.

[37] D. Yang, H. Roth, Z. Xu, F. Milletari, L. Zhang, and D. Xu, "Searching learning strategy with reinforcement learning for 3D medical image segmentation," in *Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Intervent*. Cham, Switzerland: Springer, 2019, pp. 3–11.

[38] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," in *Proc. Int. Conf. Learn. Represent.*, 2018. [Online]. Available: https://openreview.net/forum?id=r1Ddp1-Rb

[39] A. Zhao, G. Balakrishnan, F. Durand, J. V. Guttag, and A. V. Dalca, "Data augmentation using learned transformations for one-shot medical image segmentation," 2019, *arXiv:1902.09383*.

[40] B. Zoph and Q. Le, "Neural architecture search with reinforcement learning," in *Proc. Int. Conf. Learn. Represent.*, 2017. [Online]. Available: https://openreview.net/forum?id=r1Ue8Hcxg