Automated segmentation of organs and tumors from partially labeled 3D CT in MICCAI FLARE 2023 Challenge

Andriy Myronenko[0000−0001−8713−7031], Dong Yang[0000−0002−5031−4337], Yufan He[0000−0003−4095−9104], and Daguang Xu[0000−0002−4621−881X]

NVIDIA
amyronenko@nvidia.com

Abstract. Automated organ and tumor segmentation from 3D CT is one of the key tasks in medical image analysis. In this work, we describe our solution to the FLARE 2023[1] challenge (team NVAUTO). We use an automated segmentation method Auto3DSeg[2] available in MONAI[3]. Our method achieves a 93% average organs class Dice score, and a 43% tumor class Dice score based on the 5-fold cross validation.

Keywords: Auto3DSeg · MONAI · Segmentation.

1 Introduction

Three dimensional computer tomography (CT) is one of the key medical imaging modalities, which gives insights into the human body anatomy and has many applications in disease detection and monitoring. Automated 3D segmentation of organs and tumors from 3D CT is a valuable tool for treatment planning and disease analysis. Deep learning methods are able to learn from image examples, and can be run fast in clinical practice. The accuracy of such methods depends on the ground truth data available for training. The amount of public 3D CT labeled data remains low, since it is challenging and time consuming to create ground truth 3D labels, and requires a trained physician or radiologist expertise.

Fast, Low-resource, and Accurate oRgan and Pan-cancer sEgmentation in Abdomen CT (FLARE23) challenge aims to promote the development of universal organ and tumor segmentation in 3D CT scans [14,15]. This year, FLARE23 combines several publicly available datasets to achieve one of the largest 3D CT datasets, albeit with only partial labels. FLARE23 includes a large subset of cases with tumor only labels in various locations (without focusing on a specific area). Finally a large unlabeled CT subset is also provided to facilitate potentially unsupervised learning. In total, FLARE23 includes 4000 CT cases: 2200 partially labeled, and 1800 unlabeled (see Sec. 2.3 for more details). Such

1 https://codalab.lisn.upsaclay.fr/competitions/12239
2 https://monai.io/apps/auto3dseg
3 https://github.com/Project-MONAI/MONAI

a large CT data collection provides diverse variability of data examples, which come from a variety of medical institutions and populations. At the same time it poses a big algorithmic challenge, since ground truth labels are only partial, contoured by different people with different protocols. FLARE23 also adds another layer of complexity with a requirement for low GPU memory and short time processing, with an ideal setting of segmenting image under 15 seconds and under 4GB peak GPU memory.

Our solution is based on the automated supervised training of Auto3DSeg from MONAI [2]. We re-use supervised semantic segmentation where possible, and convert the data to a form acceptable for supervised training. Specifically we re-label the missing labels in each image using our own pseudo-labels, and retrain on the full dataset as it were a supervised problem. In a nutshell, our submission is an ensemble of 5 models (SegResNet [18] from MONAI[4]) processed at $1 \times 1 \times 1\text{mm}^3$ CT resolution. We made several optimizations to the inference pipeline to be able run end-to-end inference at ~3.6GB peak GPU memory and ~25 seconds per CT image on average. We describe the method and the optimizations in Sec. 2.

2 Method

We implemented our approach with MONAI [2] using Auto3DSeg open-source project. Auto3DSeg is an automated solution for 3D medical image segmentation, utilizing open source components in MONAI, offering both beginner and advanced researchers the means to effectively develop and deploy high-performing segmentation algorithms.

The labeled portion of the FLARE23 dataset are only partially labeled, with many cases including only a single labeled class. Since Auto3DSeg is a fully supervised segmentation solution, we split training into two stages: a) training on the fully labeled small subset (we found 250 cases out of 2200 to include all labels) b) pseudo-labeling the missing classes in the rest of the cases, and retraining a second round (second supervised training). Pseudo-labeling is common practice for semi-supervised learning, which was also used in the previous year FLARE22 champion solutions [10,22]. The unlabeled portion of the FLARE23 dataset was not used (see Sec. 2.3 for more details).

The fully supervised segmentation training with Auto3DSeg is simple:

```bash
#!/bin/bash
python -m monai.apps.auto3dseg AutoRunner run \
    --input="./input.yaml --algos=segresnet"
```

where a user provided input configuration (input.yaml) including only a few lines:

```yaml
# This is the YAML file "input.yaml"
modality: CT
```

[4] https://docs.monai.io/en/latest/networks.html#segresnetds

```
datalist: "./dataset.json"
dataroot: "/data/flare23"
```

When running this command, Auto3DSeg will analyze the dataset, generate hyperparameter configurations for several supported algorithms, train them, and produce inference and ensemble. The system will automatically scale to all available GPUs and also supports multi-node training. The 3 minimum user options (in input.yaml) are data modality (CT in this case), location of the dataset (dataroot), and the list of input filenames with an associated fold number (dataset.json). We generate the 5-fold equal split assignments randomly (one based on the fully labeled 250 cases, and the second one based on the pseudo-labeled 2200 cases)

Currently, the default Auto3DSeg setting trains three 3D segmentation algorithms: SegResNet [18], DiNTS [7] and SwinUNETR [6] with their unique training recipes. SegResNet and DiNTS are convolutional neural network (CNN) based architectures, whereas SwinUNETR is based on transformers [21]. Here we used only SegResNet for simplicity and describe its training procedure in this paper to be self-inclusive. At inference, we ensemble 5 best model checkpoints of SegResNet (5-folds). Since FLARE23 specifically required a fast and low GPU memory inference, we made several trade offs between accuracy and compute time, which we describe in Sec. 2.5

2.1 Preprocessing

- we resample data to $1 \times 1 \times 1\text{mm}^3$ isotropic resolution using tri-linear interpolation for CT images, and nearest neighbor interpolation for label images.
- we normalize images to $[0, 1]$ intensity interval from a $[-250, 250]$ input CT interval.
- for the first training stage only the 250 fully labeled images are used (based on the label analysis in Sec. 2.3)

Image resampling and normalization are done on the fly during training, and are a part of the training processing (in contrast to an off-line preprocessing). Auto3DSeg also caches in RAM the resampled data during the first training epoch, to speed up training automatically. If the RAM size is not sufficient, only a fraction of the data is cached, and the rest is recomputed at each epoch. This allowed us to avoid an off-line resaving step of the resampled data (which is quite large for FLARE23 dataset), and to quickly experiment with different re-sampling strategies (e.g. to try different image resolutions).

2.2 Proposed Method

The underlying network architecture is SegResNet [18] from MONAI[5]. It is an asymmetric encode-decoder based semantic segmentation network. It is a U-net alike convolutional neural network with deep supervision (see Figure 1).

[5] https://docs.monai.io/en/latest/networks.html#segresnetds

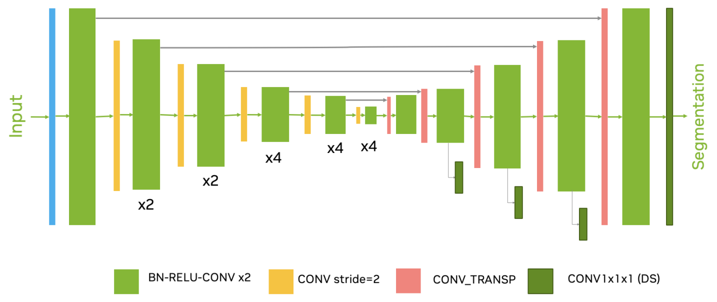

Fig. 1. SegResNet network configuration. The network uses repeated ResNet blocks with batch normalization and deep supervision

The encoder part uses residual network blocks, and includes 5 stages of 1, 2, 2, 4, 4 blocks respectively. It follows a common CNN approach to downsize image dimensions by 2 progressively and simultaneously increase feature size by 2. All convolutions are $3 \times 3 \times 3$ with an initial number of filters equal to 32. The decoder structure is similar to the encoder one, but with a single block per each spatial level. Each decoder level begins with upsizing with transposed convolution: reducing the number of features by a factor of 2 and doubling the spatial dimension, followed by the addition of encoder output of the equivalent spatial level.

We use a combined Dice-Focal loss[17,12][6], and sum it over all deep-supervision sublevels[7]:

$$Loss = \sum_{i=0}^{4} \frac{1}{2^i} Loss(pred, target^{\downarrow}) \tag{1}$$

where the weight $\frac{1}{2^i}$ is smaller for each sublevel (smaller image size) i. The target labels are downsized (if necessary) to match the corresponding output size using nearest neighbor interpolation.

We use the AdamW optimizer with an initial learning rate of $2e^{-4}$ and decrease it to zero at the end of the final epoch using the Cosine annealing scheduler[8] with 3 warmup epochs. We use batch size of 1 (per GPU), random crop of 224×224×224, weight decay of $1e^{-5}$, and optimize for 300 epochs. We use several augmentations including random rotation and scale (in axial plane only), random flips, random histogram shift and random contrast adjustment.

The same exact network architecture and schedule was used both in the first stage training on the fully labeled data (250 cases) and then in the second stage on the pseudo-labeled data (2200 cases).

[6] https://docs.monai.io/en/stable/losses.html#dicefocalloss
[7] https://github.com/Project-MONAI/MONAI/blob/dev/monai/losses/ds_loss.py
[8] https://docs.monai.io/en/latest/optimizers.html#warmupcosineschedule

2.3 Partial labels

We follow a common practice of pseudo-labeling to tackle the partially labeled data. Auto3DSeg initial step uses DataAnalyzer() from MONAI to create a compact data description of the available data statistics (see data_stats_by_case.yaml file in the working directory). Among other things it will summarize available labeled classes per case, which is a convenient way to explore the data. Table 1 shows various data subsets found in the overall FLARE23 data.

Table 1. FLARE23 analysis of data subsets.

Number of cases	Number labeled classes
250	14 labeled classes: liver, right kidney, spleen, pancreas, aorta inferior vena cava, right adrenal gland, left adrenal gland, gallbladder, esophagus, stomach, duodenum, left kidney, tumor.
1062	6 labeled classes: liver, right kidney, spleen, pancreas, left kidney, tumor.
888	1 labeled class: tumor.
1800	unlabeled (unused in our method)

The inhomogeneity of the labeled cases comes from the inhomogeneity of various datasets used to assemble the overall FLARE23 data.

We train a fully supervised segmentation model (5-folds) on the fully labeled subset of 250 cases. Then, we run inference ensemble of 5 models on the remaining partially labeled cases and assign the pseudo-labels to the missing classes (only the missing labels were replaced, and the original partial label subsets were preserved). One exception is the tumor class, we did not add it as a pseudo-label (and maintain the original tumor labels for all data cases). This allowed us to have all 2200 cases fully labeled, and used them for a second round of supervised segmentation training. We did not use unlabeled cases in our method

We did experiment initially with bypassing pseudo-labeling and training on the input 2200 partially labeled cases as is. Those experiments were futile, as the network (surprisingly accurately) learned to predict only partial labels (on the partially labeled training data), and full labels (on the fully labeled data), after all, those are the examples it learned from.

2.4 Unlabeled images.

Unlabeled images were not used. The provided pseudo labels generated by the previous year FLARE22 top algorithms [10,22] were not used.

2.5 Inference

Our inference is an ensemble of 5 SegResNet model checkpoints, each inferred with a sliding window strategy (ROI size 224×224×224) over a resampled image to $1 \times 1 \times 1\text{mm}^3$ resolution. Initially we attempted to ensemble more model checkpoints, but even for the 5 models, the inference time were prohibitively long. Below we list some of the main techniques we used to speed up the overall inference.

Global abdominal region cropping. Some of the input CT images are full body CT, as large as 512×512×2000 voxels, with abdominal organs occupying only a portion of the image. To reduce image size, we trained a separate simple binary segmentation network to segment 1 class (a union of all abdominal organs) at $4 \times 4 \times 4\text{mm}^3$ low resolution. We used SegResNet with 1,2,2,4 blocks and only 16 initial filters, and trained it on 250 fully labeled cases with 128×128×128 ROI. At $4 \times 4 \times 4\text{mm}^3$ resolution, even large CT images can be inferred fast. The segmentation accuracy of the network was not essential since we wanted only to detect an approximate abdominal region. A dedicated bounding box detector would have been faster, but here we re-used the existing Auto3DSeg functionality. After running this network on the input image, we crop it to the abdominal region to reduce the image size for the subsequent main network inference/ensemble. We crop the input image only in axial (x,y) planes, but keep the full inferior-superior length of the CT. This is because the tumor class is potentially present anywhere in the body (including head and neck region or legs) and is not conforming to the abdominal region alone in FLARE23 data. Nevertheless with this technique we significantly reduced the input image size.

First model cropping. To further reduce the computational time, we further crop the input CT based on the result of the very first model (first out of 5 in the ensemble). If the first model did not detect tumors outside of the abdominal region, the inferior-superior region is cropped to further reduce the input image size (which is especially effective for full body 3D CT). This technique, however, is a trade off, that compromises accuracy in cases when the first model prediction had errors.

GPU memory. We reduced the peak GPU memory consumption in the inference pipeline with various simple code optimizations, including releasing all intermediate GPU memory variables right away and re-using GPU memory (e.g. softmax can be computed in-place). We switched to non-overlapping sliding window inference and copied to GPU memory only the current window patch, and transferred the results to CPU immediately, instead of keeping full input and outputs on GPU. This inevitably led to a compromise between compute time and peak GPU memory. Ultimately, we decided to prioritize peak GPU memory, and keep it under 4GB, and allow computations to take longer. It was easier to control, unlike the overall compute time which depends on several factors (including the input image size).

Left-right merging One small optimization we attempted was to merge labels with left-right symmetry into a single class, specifically merging both kidneys and both adrenal glands classes. The output of the network becomes 13 (a background, 11 organs and 1 tumor class), instead of 15. This saves a bit of GPU memory, as the network output is smaller. Another reason for merging, was our hypothesis that it will be easier for the model to learn the organ segmentation without attempting to differentiate left-right symmetry. At inference, a simple heuristic was used to divide merged labels into the left and right counterparts, based on the aorta center-line and two largest components of the merged kidney classes.

1D connected component analysis For post-processing, we used a simplified 1D connected component analysis (see Sec. 2.6)

Unused optimizations . Several optimizations we tried, but ultimately not used. Compiling the model with torch.compile() did lead to a faster inference, but at a price of several seconds required for compilation. Since the overall timing is measured on each case independently after a cold start, we decided against compilation. We also considered training models at a lower resolution of $1.3 \times 1.3 \times 1.3\text{mm}^3$, which reduced image size and increased the overall inference time, at a price of segmentation accuracy. And finally, it was possible to forgo ensembling of 5 models completely, and simply use 1 model, which drastically improved the computational time, but again at the price of the segmentation accuracy.

2.6 Post-processing

We used several simplified post-processing techniques, since the overall computational time was one the FLARE23 metrics. We remove the tumor class predictions of less than a 100 voxels total. We also keep the largest connected component for a subset of organs (liver, kidneys, spleen, pancreas, aorta). Furthermore, for each of these organs, we ran a connected component analysis in 1D instead of a 3D analysis (the image is converted into a vector with ones for slices with at least one foreground voxel). This simplified 1D approach is able to remove only some disconnected inf/sup outliers (but not axial in-plane outliers).

3 Experiments

3.1 Dataset and evaluation measures

The FLARE 2023 challenge is an extension of the FLARE 2021-2022 [14][15], aiming to promote the development of foundation models in abdominal disease analysis. The segmentation targets cover 13 organs and various abdominal lesions. The training dataset is curated from more than 30 medical centers under the license, including TCIA [3], LiTS [1], MSD [20], KiTS [8,9], autoPET [5,4], TotalSegmentator [23], and AbdomenCT-1K [16]. The training set includes 4000 abdomen CT scans where 2200 CT scans with partial labels and 1800 CT scans without labels. The validation and testing sets include 100 and 400 CT scans, respectively, which cover various abdominal cancer types, such as liver cancer, kidney cancer, pancreas cancer, colon cancer, gastric cancer, and so on. The organ annotation process used ITK-SNAP [24], nnU-Net [11], and MedSAM [13].

The evaluation metrics encompass two accuracy measures—Dice Similarity Coefficient (DSC) and Normalized Surface Dice (NSD)—alongside two efficiency measures—running time and area under the GPU memory-time curve. These metrics collectively contribute to the ranking computation. Furthermore, the running time and GPU memory consumption are considered within tolerances of 15 seconds and 4GB, respectively.

3.2 Implementation details

Environment settings The development environments used for training is presented in Table 2, and was done inside of a docker "nvidia/pytorch:23.06-py3", which comes with PyTorch 2.1 and many libraries preinstalled.

Table 2. Development environments and requirements.

Docker	nvcr.io/nvidia/pytorch:23.06-py3
System	Ubuntu 22.04.2 LTS
CPU	Intel(R) Xeon(R) Platinum 8362 CPU
RAM	950G
GPU (number and type)	8x NVIDIA A40 48G
CUDA version	12.1
Programming language	Python 3.10
Deep learning framework	MONAI 1.2, PyTorch 2.1

Training protocols 1. The training was done in 2 stages (see 2.3). For each stage we trained 5 models (based on a random 5-fold split), and kept the best checkpoint based on the corresponding validation Dice value. The first stage checkpoints were used only to append pseudo-labels to the cases with missing labels (partial labels). After that 5 more models were trained on the full set of 2200 cases.

2. We use several augmentations including random rotation and scale (in axial plane only), random flips, random histogram shift and random contrast adjustment from MONAI transforms.

3. We crop 1 random patch per image of size 224×224×224 voxels (which is equivalent to a batch size of 8 on 8 GPU machine). The crop is sampled with equal probability from the one of the 15 regions (background, tumor and 13 organs)

4. We use only the Dice value (on the validation fold subset) to select the best checkpoint.

Table 3. Training protocols.

Network initialization	Random
Batch size	8
Patch size	224×224×224
Total epochs	300
Optimizer	AdamW
Initial learning rate (lr)	2e-4
Lr decay schedule	Cosine
Training time	24 hours (per 1 model)
Loss function	dice-focal loss
Number of model parameters	87M

4 Results and discussion

We trained 5 models using 5-fold cross-validation on the 2200 cases (with added pseudo labels). Based on our random 5-fold split, the average Dice scores per fold per class are shown in Table 4. When computing the Dice score, for each class, only the original ground truth labels were used and the missing labels were skipped. We obtained a good organ segmentation accuracy, with many organs having average Dice scores above 95%, and a low tumor Dice score of 44.71% on average.

The low Dice score of the tumor class can be attributed to several factors. A large sub-set of the FLARE23 dataset included only the tumor class labels (888 cases out of 2200, see 2.3). Many of these cases come from the Autopet [5,4] challenge dataset, whose goal is a whole body tumor segmentation from paired 3D PET/CT images. The ground truth Autopet labels were primarily contoured on the PET modality using the hyper-intensity indicators. If we understand it correctly, the FLARE23 dataset adopted only the 3D CT portion of these images. So, there may not be enough information in CT images alone to identify tumors. Secondly, we hypothesize, that since many FLARE23 data subsets come from *organ* focused challenges, they may have missing tumor labels (e.g., a labeled liver class, without labeling tumors within the liver). This would create a conflicting ground truth for the algorithm to differentiate tumors and organs.

FLARE23 evaluation metrics assign a high value to tumor: equal value to the tumor accuracy vs all the organs combined. In effort to improve the tumor Dice score, we tried to retrain with a re-weighted loss. Instead of equal weighted averaging of a loss value per class, we assign a high weight (e.g. 10) to the tumor loss component. We were able to improve the tumor class average Dice score to

$\sim$50% (up from 44.71%), at the price of reducing average organs Dice score by almost $\sim$10%. Visually, these results looked worse for organ segmentation, and the tumor segmentation improvements were difficult to judge. Since the ground truth tumor labels were the least consistent/reliable in the training dataset (compared to the organs labels), we ultimately decided not to use prioritize tumor loss, and not to use this strategy.

Table 4. Dice accuracy per class using our 5-fold training data random split. Each fold corresponds to the best checkpoint model trained during 5-fold cross-validation.

	fold 0	fold 1	fold 2	fold 3	fold 4	Average
Liver	97.08	97.18	97.19	97.29	97.39	97.22±0.11
Right Kidney	95.73	94.77	94.98	95.21	95.38	95.21±0.33
Spleen	96.59	96.95	97.04	96.94	96.87	96.87±0.15
Pancreas	85.41	85.93	86.31	86.96	86.36	86.19±0.51
Aorta	96.57	96.39	96.44	96.76	96.70	96.57±0.14
Inferior vena cava	95.18	94.73	95.71	95.00	94.74	95.07±0.36
Right adrenal gland	88.12	89.90	90.09	89.71	87.54	89.07±1.03
Left adrenal gland	87.80	89.23	90.16	89.83	87.39	88.88±1.09
Gallbladder	94.79	90.31	93.38	94.71	93.11	93.26±1.62
Esophagus	90.61	90.81	91.94	91.26	90.63	91.05±0.50
Stomach	96.75	96.77	97.33	97.37	96.65	96.97±0.31
Duodenum	91.48	91.24	92.93	93.06	89.75	91.69±1.21
Left kidney	95.77	95.11	94.76	95.10	95.59	95.26±0.37
Tumor	43.51	41.54	43.69	48.30	46.49	44.71±2.39
Average						89.55±12.96

The compute performance of our submission docker is shown in Table 4. We were able to achieve a peak GPU memory allocation of 3.5GB (well below the recommended 4GB minimum). Furthermore, the peak GPU memory allocation is independent of the input size. This allows our method to run even on low GPU memory GPU, or on a shared environment where GPU resources are shared between several applications. The average compute time of $\sim$25 seconds was above the recommended 15 seconds minimum. It was possible to reduce it by e.g. using only a single model (instead of a 5 model ensemble), but we decided to keep the ensemble to maintain better segmentation quality. Furthermore, about a half of the run-time was used to start the docker and load the libraries and the model checkpoints for each input image from scratch. In practice these steps can be done just once, and the overall inference time per case becomes much smaller.

4.1 Qualitative results on validation set

A visualization of the ground truth labels and the predicted results of one of the validation cases is shown in Fig. 2. Organs are accurately segmented inline with Tab. 4 results, with a slight undersegmentation of the pancreas class. The tumor

Table 5. Quantitative evaluation of segmentation efficiency in terms of the running them and GPU memory consumption. Total GPU denotes the area under GPU Memory-Time curve. Evaluation GPU platform: NVIDIA QUADRO RTX5000 (16G) CUDA 11.8. The numbers are provided by the organizers after running our docker.

Case ID	Image Size	Running Time (s)	Max GPU (MB)	Total GPU (MB)
0001	(512, 512, 55)	23.5	3562MB	470995
0051	(512, 512, 100)	26.2	3562MB	692182
0017	(512, 512, 150)	20.2	3562MB	392637
0019	(512, 512, 215)	25.5	3562MB	619403
0099	(512, 512, 334)	29.7	3562MB	682317
0063	(512, 512, 448)	28.7	3562MB	647214
0048	(512, 512, 499)	22.68	3562MB	598048
0029	(512, 512, 554)	21.61	3562MB	532494

segmentation includes a cut-off undersegmentation artifact, most likely caused by the sliding window inference with no overlap (we removed the default overlap of 0.625 as a trade off to speed up the inference).

4.2 Limitation and future work

Some limitations of the current work include not using the unlabeled data, and using a simple pseudo-labeling method. A dedicated semi-suprvised or unsupervised learning method, could provide better accuracy. Future work can include a better technique to handle tumor class segmentation, taking into account its diverse variability in various data subsets.

5 Conclusion

In this work, we describe our method submission to the FLARE23 challenge, using Auto3DSeg from MONAI. Our method is a supervised semantic segmentation, which uses pseudo-labels on the missing classes, to be able to work with partially labeled data. Our method achieves Dice scores of 93% for organs classes, and 43% for the tumor class.

Acknowledgements The authors of this paper declare that the segmentation method they implemented for participation in the FLARE 2023 challenge has not used any pre-trained models nor additional datasets other than those provided by the organizers. The proposed solution is fully automatic without any manual intervention. We thank all the data owners for making the CT scans publicly available and CodaLab [19] for hosting the challenge platform.

We also thank FLARE23 organizers for assembling such a large combined dataset, and for creating this valuable platform for researchers to compare and evaluate their methods.

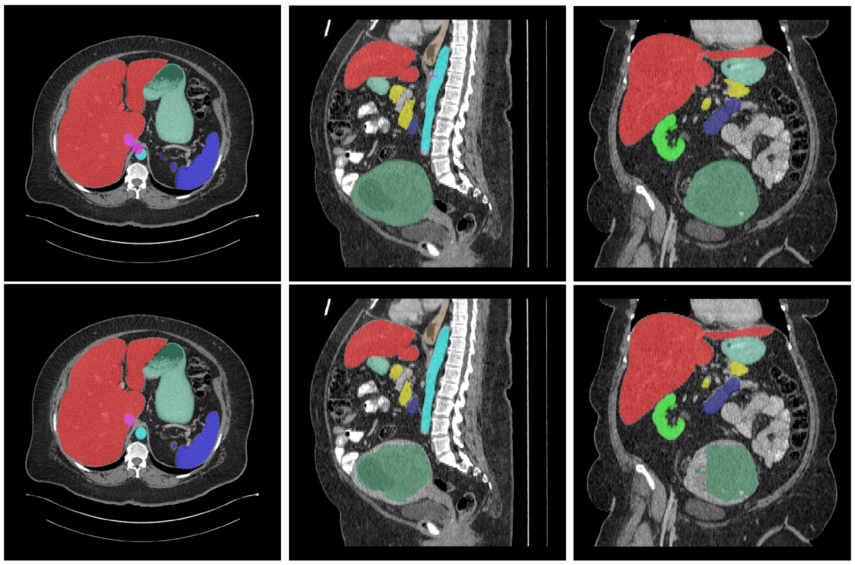

Fig. 2. A visualization of the FLARE23Ts_0017 case (axial slice 117): top row - ground truth, bottom row - predicted result. Visually organ segmentation matches the groundtruth well, with our predicted result slightly under-segmenting Pancreas (in yellow). Interestingly, the ground truth label incorrectly includes 2 inferior vena cavas (in pink/magenta) on this slice, with one of them even overlapping with aorta (in cyan). A large tumor in the lower abdomen was accurately detected but substantially under-segmented in our prediction result (in dark green). Straight edges of the tumor cut-off indicate that one of the reasons for under-segmentation could be the sliding window inference without overlap (which produces this boundary artifact).

References

1. Bilic, P., Christ, P., Li, H.B., Vorontsov, E., Ben-Cohen, A., Kaissis, G., Szeskin, A., Jacobs, C., Mamani, G.E.H., Chartrand, G., Lohöfer, F., Holch, J.W., Sommer, W., Hofmann, F., Hostettler, A., Lev-Cohain, N., Drozdzal, M., Amitai, M.M., Vivanti, R., Sosna, J., Ezhov, I., Sekuboyina, A., Navarro, F., Kofler, F., Paetzold, J.C., Shit, S., Hu, X., Lipková, J., Rempfler, M., Piraud, M., Kirschke, J., Wiestler, B., Zhang, Z., Hülsemeyer, C., Beetz, M., Ettlinger, F., Antonelli, M., Bae, W., Bellver, M., Bi, L., Chen, H., Chlebus, G., Dam, E.B., Dou, Q., Fu, C.W., Georgescu, B., i Nieto, X.G., Gruen, F., Han, X., Heng, P.A., Hesser, J., Moltz, J.H., Igel, C., Isensee, F., Jäger, P., Jia, F., Kaluva, K.C., Khened, M., Kim, I., Kim, J.H., Kim, S., Kohl, S., Konopczynski, T., Kori, A., Krishnamurthi, G., Li, F., Li, H., Li, J., Li, X., Lowengrub, J., Ma, J., Maier-Hein, K., Maninis, K.K., Meine, H., Merhof, D., Pai, A., Perslev, M., Petersen, J., Pont-Tuset, J., Qi, J., Qi, X., Rippel, O., Roth, K., Sarasua, I., Schenk, A., Shen, Z., Torres, J., Wachinger, C., Wang, C., Weninger, L., Wu, J., Xu, D., Yang, X., Yu, S.C.H., Yuan, Y., Yue, M., Zhang, L., Cardoso, J., Bakas, S., Braren, R., Heinemann, V., Pal, C., Tang, A., Kadoury, S., Soler, L., van Ginneken, B., Greenspan, H., Joskowicz, L., Menze, B.: The liver

tumor segmentation benchmark (lits). Medical Image Analysis **84**, 102680 (2023) 8

2. Cardoso, M.J., Li, W., Brown, R., Ma, N., Kerfoot, E., Wang, Y., Murrey, B., Myronenko, A., Zhao, C., Yang, D., Nath, V., He, Y., Xu, Z., Hatamizadeh, A., Myronenko, A., Zhu, W., Liu, Y., Zheng, M., Tang, Y., Yang, I., Zephyr, M., Hashemian, B., Alle, S., Darestani, M.Z., Budd, C., Modat, M., Vercauteren, T., Wang, G., Li, Y., Hu, Y., Fu, Y., Gorman, B., Johnson, H., Genereaux, B., Erdal, B.S., Gupta, V., Diaz-Pinto, A., Dourson, A., Maier-Hein, L., Jaeger, P.F., Baumgartner, M., Kalpathy-Cramer, J., Flores, M., Kirby, J., Cooper, L.A.D., Roth, H.R., Xu, D., Bericat, D., Floca, R., Zhou, S.K., Shuaib, H., Farahani, K., Maier-Hein, K.H., Aylward, S., Dogra, P., Ourselin, S., Feng, A.: MONAI: An open-source framework for deep learning in healthcare (2022) 2
3. Clark, K., Vendt, B., Smith, K., Freymann, J., Kirby, J., Koppel, P., Moore, S., Phillips, S., Maffitt, D., Pringle, M., Tarbox, L., Prior, F.: The cancer imaging archive (tcia): maintaining and operating a public information repository. Journal of Digital Imaging **26**(6), 1045–1057 (2013) 8
4. Gatidis, S., Früh, M., Fabritius, M., Gu, S., Nikolaou, K., La Fougère, C., Ye, J., He, J., Peng, Y., Bi, L., et al.: The autopet challenge: Towards fully automated lesion segmentation in oncologic pet/ct imaging. preprint at Research Square (Nature Portfolio) (2023). `https://doi.org/https://doi.org/10.21203/rs.3.rs-2572595/v1` 8, 9
5. Gatidis, S., Hepp, T., Früh, M., La Fougère, C., Nikolaou, K., Pfannenberg, C., Schölkopf, B., Küstner, T., Cyran, C., Rubin, D.: A whole-body fdg-pet/ct dataset with manually annotated tumor lesions. Scientific Data **9**(1), 601 (2022) 8, 9
6. Hatamizadeh, A., Nath, V., Tang, Y., Yang, D., Roth, H.R., Xu, D.: Swin unetr: Swin transformers for semantic segmentation of brain tumors in mri images. In: International MICCAI Brainlesion Workshop. pp. 272–284. Springer (2021) 3
7. He, Y., Yang, D., Roth, H., Zhao, C., Xu, D.: Dints: Differentiable neural network topology search for 3d medical image segmentation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 5841–5850 (2021) 3
8. Heller, N., Isensee, F., Maier-Hein, K.H., Hou, X., Xie, C., Li, F., Nan, Y., Mu, G., Lin, Z., Han, M., Yao, G., Gao, Y., Zhang, Y., Wang, Y., Hou, F., Yang, J., Xiong, G., Tian, J., Zhong, C., Ma, J., Rickman, J., Dean, J., Stai, B., Tejpaul, R., Oestreich, M., Blake, P., Kaluzniak, H., Raza, S., Rosenberg, J., Moore, K., Walczak, E., Rengel, Z., Edgerton, Z., Vasdev, R., Peterson, M., McSweeney, S., Peterson, S., Kalapara, A., Sathianathen, N., Papanikolopoulos, N., Weight, C.: The state of the art in kidney and kidney tumor segmentation in contrast-enhanced ct imaging: Results of the kits19 challenge. Medical Image Analysis **67**, 101821 (2021) 8
9. Heller, N., McSweeney, S., Peterson, M.T., Peterson, S., Rickman, J., Stai, B., Tejpaul, R., Oestreich, M., Blake, P., Rosenberg, J., et al.: An international challenge to use artificial intelligence to define the state-of-the-art in kidney and kidney tumor segmentation in ct imaging. American Society of Clinical Oncology **38**(6), 626–626 (2020) 8
10. Huang, Z., Wang, H., Ye, J., Niu, J., Tu, C., Yang, Y., Du, S., Deng, Z., Gu, L., He, J.: Revisiting nnu-net for iterative pseudo labeling and efficient sliding window inference. In: MICCAI Challenge on Fast and Low-Resource Semi-supervised Abdominal Organ Segmentation. pp. 178–189. Springer (2022) 2, 5

11. Isensee, F., Jaeger, P.F., Kohl, S.A., Petersen, J., Maier-Hein, K.H.: nnu-net: a self-configuring method for deep learning-based biomedical image segmentation. Nature Methods **18**(2), 203–211 (2021) 8
12. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection (2018) 4
13. Ma, J., Wang, B.: Segment anything in medical images. arXiv preprint arXiv:2304.12306 (2023) 8
14. Ma, J., Zhang, Y., Gu, S., An, X., Wang, Z., Ge, C., Wang, C., Zhang, F., Wang, Y., Xu, Y., Gou, S., Thaler, F., Payer, C., Štern, D., Henderson, E.G., McSweeney, D.M., Green, A., Jackson, P., McIntosh, L., Nguyen, Q.C., Qayyum, A., Conze, P.H., Huang, Z., Zhou, Z., Fan, D.P., Xiong, H., Dong, G., Zhu, Q., He, J., Yang, X.: Fast and low-gpu-memory abdomen ct organ segmentation: The flare challenge. Medical Image Analysis **82**, 102616 (2022) 1, 7
15. Ma, J., Zhang, Y., Gu, S., Ge, C., Ma, S., Young, A., Zhu, C., Meng, K., Yang, X., Huang, Z., Zhang, F., Liu, W., Pan, Y., Huang, S., Wang, J., Sun, M., Xu, W., Jia, D., Choi, J.W., Alves, N., de Wilde, B., Koehler, G., Wu, Y., Wiesenfarth, M., Zhu, Q., Dong, G., He, J., the FLARE Challenge Consortium, Wang, B.: Unleashing the strengths of unlabeled data in pan-cancer abdominal organ quantification: the flare22 challenge. arXiv preprint arXiv:2308.05862 (2023) 1, 7
16. Ma, J., Zhang, Y., Gu, S., Zhu, C., Ge, C., Zhang, Y., An, X., Wang, C., Wang, Q., Liu, X., Cao, S., Zhang, Q., Liu, S., Wang, Y., Li, Y., He, J., Yang, X.: Abdomenct-1k: Is abdominal organ segmentation a solved problem? IEEE Transactions on Pattern Analysis and Machine Intelligence **44**(10), 6695–6714 (2022) 8
17. Milletari, F., Navab, N., Ahmadi, S.: V-Net: Fully convolutional neural networks for volumetric medical image segmentation. CoRR (2016), `http://arxiv.org/abs/1606.04797` 4
18. Myronenko, A.: 3D MRI brain tumor segmentation using autoencoder regularization. In: International MICCAI Brainlesion Workshop. pp. 311–320. Springer (2018) 2, 3
19. Pavao, A., Guyon, I., Letournel, A.C., Tran, D.T., Baro, X., Escalante, H.J., Escalera, S., Thomas, T., Xu, Z.: Codalab competitions: An open source platform to organize scientific challenges. Journal of Machine Learning Research **24**(198), 1–6 (2023) 11
20. Simpson, A.L., Antonelli, M., Bakas, S., Bilello, M., Farahani, K., van Ginneken, B., Kopp-Schneider, A., Landman, B.A., Litjens, G., Menze, B., Ronneberger, O., Summers, R.M., Bilic, P., Christ, P.F., Do, R.K.G., Gollub, M., Golia-Pernicka, J., Heckers, S.H., Jarnagin, W.R., McHugo, M.K., Napel, S., Vorontsov, E., Maier-Hein, L., Cardoso, M.J.: A large annotated medical image dataset for the development and evaluation of segmentation algorithms. arXiv preprint arXiv:1902.09063 (2019) 8
21. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. CoRR **abs/1706.03762** (2017), `http://arxiv.org/abs/1706.03762` 3
22. Wang, E., Zhao, Y., Wu, Y.: Cascade dual-decoders network for abdominal organs segmentation. In: MICCAI Challenge on Fast and Low-Resource Semi-supervised Abdominal Organ Segmentation. pp. 202–213. Springer (2022) 2, 5
23. Wasserthal, J., Breit, H.C., Meyer, M.T., Pradella, M., Hinck, D., Sauter, A.W., Heye, T., Boll, D.T., Cyriac, J., Yang, S., Bach, M., Segeroth, M.: Totalsegmentator: Robust segmentation of 104 anatomic structures in ct images. Radiology: Artificial Intelligence **5**(5), e230024 (2023) 8

24. Yushkevich, P.A., Gao, Y., Gerig, G.: Itk-snap: An interactive tool for semi-automatic segmentation of multi-modality biomedical images. In: Annual International Conference of the IEEE Engineering in Medicine and Biology Society. pp. 3342–3345 (2016) 8

Table 6. Checklist Table. Please fill out this checklist table in the answer column.

Requirements	Answer
A meaningful title	Yes
The number of authors ($\leq$6)	4
Author affiliations and ORCID	Yes
Corresponding author email is presented	Yes
Validation scores are presented in the abstract	Yes
Introduction includes at least three parts: background, related work, and motivation	Yes
A pipeline/network figure is provided	Fig. 1
Pre-processing	Sec. 2.1
Strategies to use the partial label	Sec. 2.3
Strategies to use the unlabeled images.	Sec. 2.3
Strategies to improve model inference	Sec. 2.5
Post-processing	Sec. 2.6
Dataset and evaluation metric section is presented	Sec. 3.1
Environment setting table is provided	Tab. 2
Training protocol table is provided	Tab. 3
Efficiency evaluation results are provided	Tab. 4
Limitation and future work are presented	Yes
Reference format is consistent.	Yes