CPU-Efficient U-Net-Transformer with Quantized Soft Labels for Abdominal Organ Segmentation

Kwan Leong Chit Jeff[1][0009−0002−0645−150X] and Albert C.S. Chung[2][0000−0003−4400−9261]

[1] Hong Kong University of Science and Technology lcjkwanaa@connect.ust.hk
[2] Hong Kong University of Science and Technology achung@ust.hk

Abstract. This paper introduces a U-Net-Transformer hybrid model designed for the MICCAI FLARE 2025 Abdominal CT Organ Segmentation on Laptop Challenge. The model achieves both efficiency and performance through the integration of depthwise separable convolutions and transformer layers within the bottleneck. Our method incorporates quantized soft labels for improved boundary accuracy, aggressive multi-category data augmentation for enhanced robustness, class-weighted loss function and class-specific post-processing for precise segmentation of organs. On the online validation set, the pseudo-labeling model achieves mean Dice 0.9110 and NSD 0.9575, while the CPU-inference model achieves 0.8912 and 0.9518, respectively. Average inference time over 50 public validation cases is 19.3s per volume. These findings highlight the potential for practical deployment of the model in clinical environments with limited computational resources.

Keywords: Medical image segmentation · Abdomen · Computed tomography · Semi-supervised learning · U-Net · Transformer

1 Introduction

Accurate segmentation of abdominal organs in medical imaging is important for various downstream clinical applications. While deep learning has substantially advanced segmentation accuracy, most state-of-the-art models rely on GPU resources for inference, limiting their practicality in clinical environments. Thus, there is a pressing need for methods optimized for low-resource settings, particularly for deployment on laptops.

The FLARE 2025 Abdominal CT Organ Segmentation on Laptop Challenge addresses this issue by investigating whether state-of-the-art abdominal segmentation models can be adapted for CPU-only environments without sacrificing accuracy. Unlike previous years, this challenge prohibits cascaded approaches to encourage single-model designs that are both computationally efficient and accurate. Participants are tasked with developing a model to segment 13 organs and the background, using 50 labeled training images, 2,000 unlabeled images, and 50 publicly labeled validation images. Pseudo-labels generated by the winning models of FLARE 2022 [13] are available for the 2000 images, including

contributions from team aladdin5 [9], awarded for best segmentation accuracy, and team blackbean [25], recognized for overall performance.

Advances in abdominal organ segmentation have been driven by deep convolutional networks, transformer-based architectures, and hybrid models that capture both local texture and global context, with U-Net variants and 3D extensions serving as robust baselines. These methods remain computationally demanding, raising challenges for deployment in CPU-only clinical workflows. For example in 2022, while the aladdin5 team's [9] model achieved the highest Dice score using nnU-Net [10], their approach proved inefficient for the speed and low-resource constraints even for the FLARE 2022 challenge, which allowed the use of 2GB GPU memory. While the blackbean team's [25] model was efficient, they employed a localization-to-segmentation 2-stage framework which is not allowed this year, and also required a GPU.

Motivated by the need for designing a one-shot, accurate, and lightweight segmentation model, this paper proposes an efficient model architecture tailored for CPU deployment, alongside a carefully designed data processing pipeline. Our approach aims to strike a balance between computational efficiency and segmentation accuracy, making it viable for real-world use cases such as laptop-based inference in clinical environments. The key methodological contributions of our work are as follows:

1. Employing quantized soft labels to achieve sub-voxel boundary accuracy, and allowing the combination of multiple model predictions for pseudo-labeling.
2. Applying aggressive data augmentation strategies across spatial, intensity, and coarse block categories to improve model robustness and generalization.
3. Applying a class-weighted loss function and a class-specific small object filtering in post-processing tailored to each organ.
4. Model design integrating the performance advantages of the U-Net and transformer while balancing with inference speed on CPU.

2 Method

2.1 Pre-processing

All volumes are first reoriented into the "RAS" format. For training the larger pseudo-label annotator model, volumes are resampled to (0.8, 0.8, 2.5) mm voxel spacing to preserve resolution. For the smaller inference model, volumes are resampled to (1.6, 1.6, 2.5) mm voxel spacing, as well as center-cropped on the right-left and anterior-posterior axes into (256, 256) voxels to improve computational speed. As we constrain the size of the first two dimensions, we only need to apply a 1-D sliding window over the inferior-superior axis, thus computation time is now linear to volume size instead of cubic.

CT images CT images are preprocessed by reorienting and resampling to the desired voxel spacing using trilinear interpolation. Image intensities are clipped

to the foreground intensity range of [-974.0, 295.0], derived from the joint 0.05 and 99.5 percentiles of foreground voxels across 50 ground-truth images and 2000 pseudo-labels generated by aladdin5. Intensities are then z-score-normalized using a mean of 95.958 and a standard deviation of 139.964. Due to potential inaccuracies in pseudo-labels, z-score statistics for normalization are computed solely from ground-truth foreground voxels.

Quantized soft labels Our method introduces quantization to soft labels, allowing practical use of soft labels on volumetric data, which would have been too bulky to process otherwise.

For the 50 ground-truth labels, we first reoriented and resampled them to the target voxel spacing. Integer labels are then converted to one-hot encoded representations, enabling the use of trilinear interpolation in terms of probabilities, which more accurately preserves sub-voxel boundaries compared to nearest-neighbor interpolation.

Pseudo-labels from aladdin5 and blackbean models are first filtered to ensure the labels only span from 0 to 14, replacing out of range integers with the background 0 class. Then, we take the mean ensemble of labels in one-hot form, after which their probabilities are renormalized. In contrast, due to computational constraints and inherent uncertainty in pseudo-label accuracy, nearest-neighbor interpolation is used for these 2000 labels.

However, the resulting expansion from single-channel uint8 integer labels to 14-channel float32 soft labels significantly increased computational demand during training. To alleviate this, we quantized the soft labels to uint8 precision by multiplying probabilities by 255 and truncating the results. We redistribute residual mass by adding 1 to the largest-residual classes until the per-voxel channel sum is 255. Prior to loss computation, these quantized labels are converted back to float32 format on the GPU.

2.2 Proposed method

To accommodate for the FLARE 2025 CPU inference constraint, we developed a custom architecture that balances accuracy and computational efficiency. After exploring several U-Net-based designs incorporating residual convolution blocks, transformer modules, and various normalization schemes, we settled on the following configuration as shown in Fig. 1.

Drawing inspiration from Inception v3 [22], our architecture integrates multi-scale representations using factorized convolutions. In the annotation model, we begin with a $3 \times 3 \times 3$ convolution to a number of hidden channels, followed by group normalization [28] with each channel as a group, effectively instance normalization, and a GELU [8] activation. Hidden layer activations are subsequently processed in parallel by both standard and dilated $3 \times 3 \times 3$ convolutions, effectively creating factorized receptive fields equivalent to $5 \times 5 \times 5$ and $7 \times 7 \times 7$ kernels, followed by normalization and activation, each to half the number of hidden channels. Channel-wise normalization complements channel-based

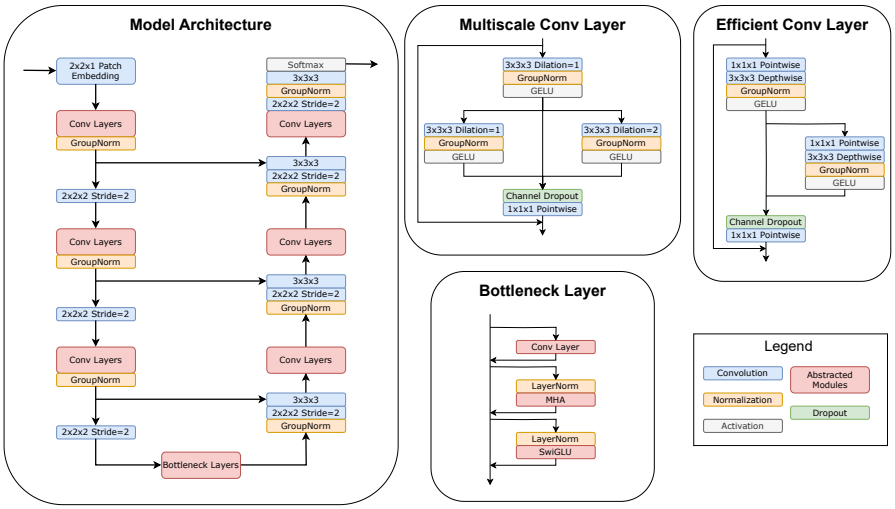

Fig. 1. U-Net based architecture with residual connections, transformer layers at the bottleneck bridge and multiscale convolution feature concatenation.

dropout [23], promoting channel independence. We set the hidden channels to be smaller than the number of channels in the residual stream to reduce computation per convolution, but the concatenation of multiscale feature maps gives a total of 2× hidden channels to mix with the final point-wise convolution, thus also achieving the representational capacity of channel expanding blocks.

At the encoder-decoder bottleneck, convolutional blocks alternate with transformer [24] blocks. Transformer blocks reshape the volume into a linear sequence and apply multi-head attention followed by SwiGLU [20] blocks instead of conventional MLP layers. Activations are pre-normalized via layer normalization, and dropout is applied conventionally in i.i.d. form. The bottleneck stage offers a global field of view via the transformer layers with the semantically rich latents extracted from previous convolution layers.

We perform upsampling and downsampling using convolutions or transposed convolutions with $2 \times 2 \times 2$ kernels and stride 2. To further stabilize training, group normalization [28] is applied to residual streams with groups of 8 channels, chosen to balance computational efficiency and representation capability before downsampling or upsampling. Initial patch embeddings of size $2 \times 2 \times 1$ reduce input dimensionality, while transposed convolutions near the output restore resolution to the original size with 16 channels. A final $3 \times 3 \times 3$ convolution removes checkerboard artifacts to generate predictions for the 14-class segmentation task. Similarly, a $3 \times 3 \times 3$ convolution is used when merging the U-Net skip connections across the same resolution with the upsampled activations.

Table 1 show the specifications of the small inference model while Table 2 show the specifications of the large annotation model. The resolution column

Table 1. Small inference model, 8.38M parameters

Resolution	Layers	Channels	Hidden Channels	Dropout
256, 256, 64	-	1 | 14	-	0
128, 128, 64	4×2	32	24×2	0.0125
64, 64, 32	6×2	64	48×2	0.025
32, 32, 16	6×2	128	96×2	0.0375
16, 16, 8	24	256	192×2	0.05

Table 2. Large annotation model, 37.98M parameters

Resolution	Layers	Channels	Hidden Channels	Dropout
224, 224, 112	-	1 | 14	-	0
112, 112, 112	6×2	48	32×2	0.05
56, 56, 56	6×2	96	64×2	0.1
28, 28, 28	6×2	192	128×2	0.15
14, 14, 14	18	384	256×2	0.2

show the spatial size of the activations on each level of the encoder and decoder. The encoder and decoder are symmetric so the number of layers are denoted with ×2. There is 1 input channel and 14 output channels for each class. Inside the convolution block, the convolutions would apply a channel reduction towards the hidden channel dimension, but the concatenation allows the total hidden channels to be twice as large. Finally, dropout is applied increasingly in a linear fashion in deeper levels with more channels.

For our small inference model, we note that the largest computational load comes from the 3D convolutions. We replace full convolutions with a point-wise convolution for channel mixing, followed by depthwise convolutions for spatial mixing, to create the efficient convolution layers. Channel-wise group normalization and GELU activation are applied in the same manner. Because depthwise convolutions are also channel-independent, this change is consistent with our architecture design. We eliminate dilated convolutions and replace artifact-smoothing layers with conventional point-wise convolutions to reduce cost. Transformer layers are retained in the bottleneck as the sequence length is only 2048 after downsampling, and short enough so that global attention is not a significant computational load.

2.3 Post-processing

By visualizing the model predictions against the ground truth of the public validation labels, we observe the following common mistakes made by our large annotation model. Firstly, there are boundary discrepancies around the organs, confusion with the background or with other organs. However, it contributes only to a small part of the errors. More significantly, if the patient had surgery to remove certain organs, any false positive prediction of the organ would render a 0 Dice score for the organ. Moreover, due to anatomical variations and potentially tumors, the model sometimes cannot segment organs with peculiar shapes

accurately, instead breaking into several convex components in the prediction, thus choosing to keep the largest connected component only could be detrimental to overall performance. As two-stage cascaded models are not allowed in this year's competition, we cannot do localization followed by segmentation, which causes false positives to occur in far away regions in full-body validation scans.

Based on the empirical observations of the raw predictions, we choose to take a two-pronged approach to tackle the errors. We reduce the weight of the loss of the background class to 0.05, thus increasing the proportion of false positives, and remove small objects to clean the predictions. As organs differ greatly in size, we use a per-class threshold for the small object filtering. Table 3 shows the other class-specific loss weights, as well as the small-object voxel threshold applied for each model on their corresponding spacing resolution. The loss weights are computed by the log-scale of the prevalence of organ class voxels, with the largest organ, the liver, fixed to weight 1.0, and the smallest organ, the right adrenal gland fixed to weight 2.0.

Table 3. Class specific training loss weights, and voxel thresholds for small object removal post processing.

Class	Loss Weight	Small Threshold	Large Threshold
Background	0.05	-	-
Liver	1.00	10000	10000
Right Kidney	1.35	1000	1000
Spleen	1.33	1000	1000
Pancreas	1.48	1000	1000
Aorta	1.48	1000	1000
Inferior Vena Cava	1.50	1000	1000
Right Adrenal Gland	2.00	50	100
Left Adrenal Gland	1.97	100	100
Gallbladder	1.66	300	500
Esophagus	1.78	100	100
Stomach	1.27	1000	1000
Duodenum	1.52	500	1000
Left Kidney	1.35	500	1000

2.4 Pseudo-label update

After training our teacher annotation model, we update the pseudo-labels by making predictions with (0.8, 0.8, 2.5) spacing and aforementioned postprocessing on the 2000 unlabeled data, with 0.75 overlap of sliding windows to ensure multiple views on regions of interest. To avoid false positives in full-body scans, we only update the region within a 2 cm margin of the foreground bounding box of the aladdin5 and blackbean labels. We take the mean ensemble of the 3

sources of pseudo-labels. We use trilinear interpolation to (1.6, 1.6, 2.5) spacing to take advantage of the higher resolution teacher annotations. Finally, we normalize and quantize the labels to uint8.

3 Experiments

3.1 Dataset and evaluation measures

The dataset is curated from more than 40 medical centers under the license permission, including TCIA [2], LiTS [1], MSD [21], KiTS [6,7], autoPET [5,4], AMOS [11], AbdomenCT-1K [19], TotalSegmentator [27], and past FLARE challenges [16,17,18]. The training set includes 2050 abdomen CT scans where 50 CT scans with complete labels and 2000 CT scans without labels. The validation and testing sets include 250 and 300 CT scans, respectively. The annotation process used ITK-SNAP [30], nnU-Net [10], MedSAM [14,15], and Slicer Plugins [3,15]. In additional to use all training cases for model development, we also added a coreset track where participants can select 50 cases from the training set for model development in an automatic way.

The evaluation metrics encompass two accuracy measures—Dice Similarity Coefficient (DSC) and Normalized Surface Dice (NSD)—alongside one efficiency measures—runtime. These metrics collectively contribute to the ranking computation. During inference, GPU is not available where the algorithm can only rely on CPU.

3.2 Implementation details

Environment settings The development environments and requirements are presented in Table 4.

Table 4. Development environments and requirements.

System	Ubuntu 22.04 LTS
CPU	2× AMD EPYC 9B14 96-Core Processor
RAM	512.8 GB
Programming language	Python 3.12
Deep learning framework	PyTorch 2.7, MONAI 1.5.0, torchvision 0.22.0
Specific dependencies	numpy 1.26.4, nibabel 5.3.2, scikit-image 0.24.0, scipy 1.14.1, matplotlib 3.10.1, huggingface-hub 0.24.6, pandas 2.2.2, tqdm 4.67.1
Code	https://github.com/LCJKwan/FLARE25-Task2-Harmonics

Data augmentations During training, random spatial cropping is performed with maximum dimensions of $(224, 224, 112)$ for the pseudo-labeler model and $(256, 256, 64)$ for the small inference model. Crops are subsequently symmetrically padded to dimensions divisible by 16. Extensive augmentation techniques are then applied to enhance dataset variability and model generalization. Augmentations are categorized into spatial, intensity-based, and coarse transformations, with one randomly selected augmentation per category applied to each training sample.

Spatial augmentations are randomly selected according to predefined probabilities. With probability $\frac{2}{5}$ no augmentation is applied. With probability $\frac{2}{5}$, random affine transformations are employed, comprising rotations within $\pm 20°$ and scaling within $\pm 10\%$. With the remaining probability $\frac{1}{5}$, a random 3D elastic deformation is applied in conjunction to the affine transformation. Elastic transformations create a deformation field by Gaussian smoothing, with kernel with σ randomly sampled between 1.5 and 2.0, and of voxel displacements ranging from 8 to 16. Translation augmentations are added in the right-left and anterior-posterior axes for 20 voxels while training the small inference model, as the large 256 voxel coverage is larger than most volumes, thus random cropping does not act as translation augmentation here. Global shearing, flipping, and large rotations are excluded due to anatomical implausibility.

Intensity augmentations are selected to improve robustness to variations in intensity profiles and scanner differences. With probability $\frac{4}{10}$, no intensity augmentation is performed. With probability $\frac{1}{10}$ each, one of six possible augmentations of the following are applied: Gaussian smoothing, Gaussian sharpening, Gaussian noise addition, bias field addition, contrast adjustment, or histogram shifting. Default settings provided in `MONAI 1.5.0` are used.

Coarse augmentations encourage model resilience against local information loss and promote reliance on contextual neighborhood information. With a probability of $\frac{3}{5}$, no coarse augmentation is applied. Otherwise, with probability $\frac{1}{5}$ each, either coarse dropout or coarse shuffle is randomly applied. Coarse dropout randomly replaces intensities in 1 to 4 blocks, each of size between $(16, 16, 16)$ and $(32, 32, 32)$ voxels, with random values between the maximum and minimum intensity of the volume. Coarse shuffle randomly shuffles intensities within each of 8 to 16 blocks, each sized between $(6, 6, 6)$ and $(12, 12, 12)$ voxels.

Training protocols Training protocols are detailed in Tables 5 and 6. We use the AdamW optimizer with learning rate 0.0001 and weight decay of 0.002. We apply a cosine learning rate scheduler as well. We use the Dice loss adapted for soft labels [26] along with focal loss [12] with a 1:2 ratio.

We allow models to train for 2400 epochs on the ground truth images in total along with the pseudo-labels, as ground truth labels are of higher quality. Concretely, we repeat ground truth labels by $12\times$ per epoch when training the

Table 5. Training protocols for the inference model.

Network initialization	PyTorch 2.7.0 defaults
Batch size	4
Patch size	256×256×64
Total epochs	600 pseudo + 2400 ground truth
Optimizer	AdamW
Initial learning rate (lr)	0.0001
Lr decay schedule	Cosine annealing
Weight decay	0.002
Autocast precision	float32
Training time	38.6 hours
Loss function	1×Soft Dice [26] + 2×Focal [12]
Number of model parameters	8.38M
Number of flops	147.33G[3]

large model with 200 epochs on the pseudo-labels, and 4× per epoch when training on the small inference model with 600 epochs on the pseudo-labels. We use a batch size of 4 for both the annotation and inference model.

We employ mixed precision training with `PyTorch` autocasting for the large annotation model to bfloat16, while we train the small inference model natively in float32, because bfloat16 does not offer speed gains on CPU inference.

We used `fvcore` for the flops estimation of our models, but we also wrote our rough estimates for the elementwise add, multiply, sum, divide, SiLU, GELU, unflatten, and scaled dot product attention operations as these are not supported in the `fvcore` package.

4 Results and discussion

4.1 Quantitative results on validation set

Tables 7 and 8 provides detailed segmentation performance results for each organ class, reporting both the Dice Similarity Coefficient (DSC) and Normalized Surface Dice (NSD) scores. We use a sliding window overlap of 0.75 for the annotation model, and 0.2 for the smaller inference model, according to the overlap in the use case of pseudo-label annotation or CPU inference.

The large annotator model achieves a higher mean DSC of 0.9077 and mean NSD of 0.9506 on the public validation set, and 0.9110 DSC and 0.9575 NSD on the online validation set. The smaller inference model obtains a mean DSC of 0.8884 and mean NSD of 0.9452 on the public validation set, and 0.8912 DSC and 0.9518 NSD on the online validation set. The large model consistently outperforms the smaller one across all organ classes, which indicates that the larger model's additional capacity and training strategies effectively capture more complex anatomical details.

Table 6. Training protocols for the annotation model.

Network initialization	PyTorch 2.7.0 defaults
Batch size	4
Patch size	224×224×112
Total epochs	200 pseudo + 2400 ground truth
Optimizer	AdamW
Initial learning rate (lr)	0.0001
Lr decay schedule	Cosine annealing
Weight decay	0.001
Autocast precision	bfloat16
Training time	22.5 hours
Loss function	1×Soft Dice [26] + 2×Focal [12]
Number of model parameters	37.98M
Number of flops	2545.86G[4]

Specifically, the organs that achieve the highest DSC scores include the liver, spleen and right kidney. Similarly, the NSD scores reflect strong surface segmentation performance, especially for the liver, spleen, pancreas and the aorta. On the contrary, small and anatomically complex organs such as the duodenum, the gallbladder and the adrenal glands yield comparatively lower DSC and NSD scores, with higher standard deviation as well.

4.2 Qualitative results on validation set

Fig. 2 visualizes the qualitative results for our pseudo-labeler and inference models. Case 48, 44, 15, 45, 21 are selected to represent the 0th, 25th, 50th, 75th, and 100th percentiles of mean DSC respectively from the public validation set, as segmented by the inference model. The slice number is chosen by a maximizing combination score over the number of different organs, total organ area, and area of inference model incorrect predictions.

For cases #48 and #44, we observe the existence of false positive predictions in both the pseudo-labeler and inference model, which suggests that the inference model inherited some of the false positive errors from the pseudo-labels. For case #15, we observe a false negative of a small organ label in the same inherited error fashion. For case #45 and #21, there are no significant errors but the borders of the label predictions may not match the ground truth exactly. Overall, we observe that our methods struggle relatively more for small organs, sometimes hallucinating or missing small parts for predictions with low DSC scores.

4.3 Ablation studies

Firstly, we compare using 0.2 overlap and 0.75 overlap for the inference model. It achieves 0.8883 mean DSC and 0.9450 mean NSD for the public validation set

Table 7. Quantitative evaluation results of the small inference model.

Target	Public Validation		Online Validation		Testing	
	DSC(%)	NSD(%)	DSC(%)	NSD(%)	DSC(%)	NSD (%)
Liver	97.06 ± 1.50	98.12 ± 3.43	96.52	98.16		
Right Kidney	94.68 ± 11.50	96.11 ± 11.34	95.37	97.09		
Spleen	95.97 ± 5.08	97.20 ± 8.41	96.40	98.09		
Pancreas	88.94 ± 5.00	97.46 ± 3.65	86.07	95.83		
Aorta	94.84 ± 2.00	98.77 ± 3.54	95.05	98.99		
Inferior vena cava	89.88 ± 7.07	92.01 ± 8.84	89.98	92.66		
Right adrenal gland	78.48 ± 15.99	92.72 ± 16.88	82.82	96.11		
Left adrenal gland	80.75 ± 16.49	92.88 ± 19.33	82.47	95.57		
Gallbladder	84.87 ± 23.98	87.06 ± 24.87	86.73	89.01		
Esophagus	85.52 ± 8.82	94.61 ± 9.20	82.19	92.96		
Stomach	91.70 ± 11.92	95.33 ± 10.62	93.07	96.25		
Duodenum	80.57 ± 13.59	93.02 ± 8.65	79.07	91.64		
Left kidney	91.72 ± 15.73	93.44 ± 16.09	92.78	94.97		
Average	88.84 ± 13.78	94.52 ± 13.14	89.12	95.18		

Table 8. Quantitative evaluation results of the large annotation model.

Target	Public Validation		Online Validation	
	DSC(%)	NSD(%)	DSC(%)	NSD(%)
Liver	97.45 ± 1.50	98.11 ± 3.35	97.66	98.84
Right Kidney	95.60 ± 9.17	96.18 ± 10.92	96.86	97.91
Spleen	97.01 ± 2.79	97.87 ± 6.16	95.27	96.50
Pancreas	90.00 ± 5.65	97.64 ± 4.87	87.94	96.90
Aorta	96.67 ± 1.33	99.27 ± 2.12	96.95	99.38
Inferior vena cava	91.19 ± 8.51	91.71 ± 9.61	92.23	93.30
Right adrenal gland	85.08 ± 12.67	91.71 ± 9.61	88.38	97.99
Left adrenal gland	84.62 ± 17.38	95.14 ± 14.61	87.52	97.22
Gallbladder	85.18 ± 26.40	93.96 ± 19.40	86.05	87.81
Esophagus	89.54 ± 4.22	87.15 ± 26.99	84.57	93.40
Stomach	92.45 ± 13.11	97.05 ± 3.77	94.10	96.66
Duodenum	83.68 ± 11.42	93.58 ± 15.01	82.09	92.69
Left kidney	91.60 ± 18.23	93.30 ± 8.26	94.68	96.13
Average	90.77 ± 13.33	95.06 ± 13.40	91.10	95.75

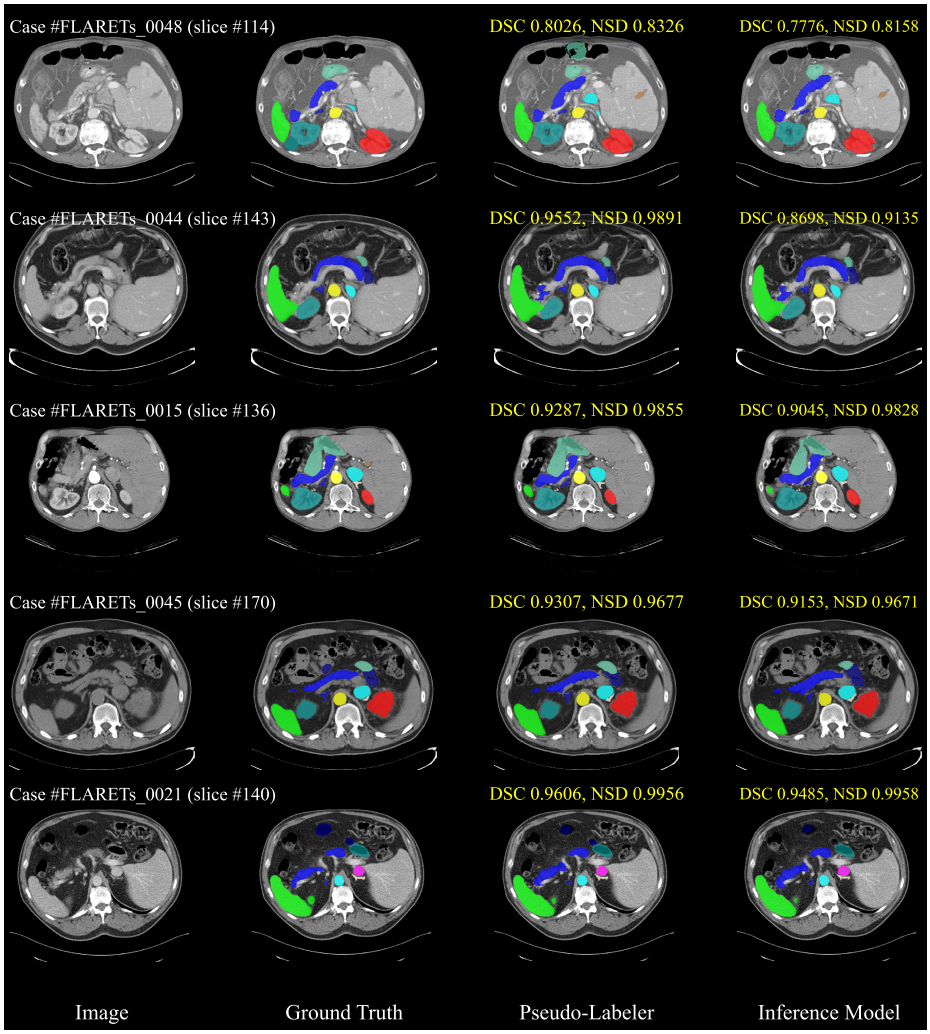

Fig. 2. Qualitative results of the pseudo-labeler and the inference model chosen by the 0^{th}, 25^{th}, 50^{th}, 75^{th}, 100^{th} percentiles of the inference model DSC scores.

with 0.75 overlap, which is not higher than 0.8884 mean DSC and 0.9542 mean NSD with 0.2 overlap. This shows that our inference model is able to create good segmentation predictions in a one-shot manner for most of the volume, thus saving computation. Next, we ablate the post-processing of class-specific small object filtering, which reduces the performance to 0.8850 mean DSC and 0.9412 mean NSD, demonstrating the effectiveness of the strategy.

Further ablation studies are done by training the model for 200 epochs on pseudo-labels and 2400 epochs on ground truth, as well as autocasting to bfloat16, to save training time. The ablation models are all trained on the (1.6, 1.6, 2.5) mm resolution and run inference with the same settings as the inference model. We report the mean DSC and mean NSD on the public validation set.

The model benefits from longer training times and full precision training as the inference model trained with 200 instead of 600 pseudo-label epochs achieves 0.8813 mean DSC and 0.9378 mean NSD, lower than the one trained with 600 pseudo-label epochs. Pseudo-label update by our labeler model also contribute to performance gains, as training with only aladdin and blackbean soft labels result in 0.8778 mean DSC and 0.9352 mean NSD. We find that using transformer layers at the bottleneck allow a small benefit as replacing them with convolution layers result in 0.8797 mean DSC and 0.9360 mean NSD.

We compare the effect of model architecture and the number of parameters, by training a pseudo-labeler model with the settings of the inference model on (1.6, 1.6, 2.5) mm resolution instead of (0.8, 0.8, 2.5) mm resolution, and (256, 256, 64) patch size instead of (224, 224, 112), for 200 pseudo-label epochs and 2400 ground truth epochs. With 37.98M parameters compared to 8.38M parameters, and more than 17× the flops, it achieves 0.8908 mean DSC and 0.9447 mean NSD, higher than that of our final inference model. Comparison with the small model suggests that size and architecture optimizations may cause only a minor performance loss. In contrast, comparison with the original pseudo-labeler scores indicates that the lower resolution is the main reason for the >1% drop in mean DSC.

4.4 Segmentation efficiency results on validation set

After obtaining the inference model, we tested it on our local laptop with Intel (R) Core (TM) Ultra 9 185H, 2300MHz, 16 Cores, 22 Logical Processors, by running a Docker container allocated with 8GB RAM on CPU. We note that scans 10 and 50 are particularly large and require extensive RAM to compute. After investigation, it is revealed that the internals of the `MONAI Spacingd` class are not memory-optimized compared to the `PyTorch interpolate` function. Thus, we augmented our inference pipeline with a switch between the `MONAI` implementation with normal scans, or to use our manual implementation of data processing and inversion, for a small potential performance degradation. The fail-safe is used when the product of number of voxels of the original volume times the number of expected voxels after interpolation exceeds 1.5×10^{15}. Averaged over the 50 cases, the inference model uses 19.3s per image, with mean (system)

Table 9. Quantitative evaluation of segmentation efficiency in terms of the running time on selected online validation images. Evaluations were run on a local laptop: Intel (R) Core (TM) Ultra 9 185H, 2300MHz, 16 Cores, 22 Logical Processors.

Case ID	Image Size	Preprocessed Size	Sliding Windows	Running Time (s)
0007	(512, 512, 215)	(256, 256, 172)	4	19.88
0027	(512, 512, 169)	(213, 213, 169)	4	15.89
0029	(512, 512, 171)	(204, 204, 171)	4	16.20
0036	(512, 512, 91)	(226, 226, 109)	2	8.35
0058	(512, 512, 56)	(256, 256, 111)	2	8.37
0063	(512, 512, 361)	(241, 241, 181)	4	21.78
0071	(512, 512, 108)	(256, 256, 108)	2	9.84
0164	(512, 512, 114)	(256, 256, 227)	5	20.14
0189	(512, 512, 89)	(199, 199, 177)	2	14.57
0190	(512, 512, 101)	(238, 238, 201)	4	17.08

CPU utilization of 39.07%, median CPU utilization of 49.67%, mean RAM of 4354 MiB and median RAM of 4544 MiB.

4.5 Results on final testing set

This is a placeholder. We will send you the testing results during MICCAI (2025.9.27).

4.6 Limitation and future work

The potential of quantized soft labels has yet to be fully explored. Due to time constraints during development, we were not able to experiment with training multiple pseudo-labelers, pseudo-label uncertainty analysis, and different resolutions. Soft labels are able to aggregate information from multiple sources, preserve sub-voxel information, and function as regularization like label smoothing. There is a higher ceiling of label quality in using soft labels, thus we anticipate that further work in this direction will unlock new levels of model performance and robustness.

Our models struggled with organ shapes exhibiting significant deviations from typical anatomy, sometimes fragmenting predictions. Although morphological post-processing was considered as a potential solution, applying it to 13 organ classes in large volumetric data in practice proved infeasible within the pseudo-label training or CPU-based inference pipelines. Incorporating morphological priors or streamlined morphological processing represents a promising direction for future exploration.

Additionally, our larger models at higher resolution consistently outperformed smaller ones by approximately 1-2% Dice score. Therefore, adapting larger models for efficient inference, possibly through weight and activation quantization, could be advantageous. However, due to limited software support for quantization of 3D convolution modules and quantization-aware training, we leave this strategy for future work.

5 Conclusion

We proposed a computationally efficient abdominal organ segmentation method optimized for CPU-based abdominal CT segmentation. Our approach uniquely integrates quantized soft labels, comprehensive data augmentation, class-specific loss weights and post-processing, all with an efficient UNet-Transformer hybrid architecture. The proposed pseudo-label annotation model demonstrates competitive performance, achieving 0.9110 mean DSC and 0.9575 mean NSD on the online validation dataset. The efficient inference model achieves 0.8912 DSC and 0.9518 NSD as well, indicating robust generalization and practical applicability. Future directions include enhanced pseudo-label refinement, extended applications of soft labels, exploring morphological priors, and model quantization strategies to further improve segmentation accuracy and computational efficiency.

Acknowledgements The authors of this paper declare that the segmentation method they implemented for participation in the FLARE 2025 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 data owners for making the CT scans publicly available and CodaBench [29] for hosting the challenge platform.

Disclosure of Interests

The authors declare no competing interests.

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) 7

2. 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) 7
3. Fedorov, A., Beichel, R., Kalpathy-Cramer, J., Finet, J., Fillion-Robin, J.C., Pujol, S., Bauer, C., Jennings, D., Fennessy, F., Sonka, M., et al.: 3d slicer as an image computing platform for the quantitative imaging network. Magnetic Resonance Imaging **30**(9), 1323–1341 (2012) 7
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. Nature Machine Intelligence (in presss) (2024) 7
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) 7
6. 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) 7
7. 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) 7
8. Hendrycks, D., Gimpel, K.: Gaussian error linear units (gelus) (2023) 3
9. Huang, Z., et al.: Revisiting nnu-net for iterative pseudo labeling and efficient sliding window inference. In: Ma, J., Wang, B. (eds.) Fast and Low-Resource Semi-supervised Abdominal Organ Segmentation, Lecture Notes in Computer Science, vol. 13816. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-23911-3_16 2
10. Isensee, F., Jaeger, P.F., Kohl, S.A.A., Petersen, J., Maier-Hein, K.H.: nnU-Net: a self-configuring method for deep learning-based biomedical image segmentation. Nature Methods **18**, 203–211 (2021). https://doi.org/10.1038/s41592-020-01008-z 2, 7
11. Ji, Y., Bai, H., GE, C., Yang, J., Zhu, Y., Zhang, R., Li, Z., Zhanng, L., Ma, W., Wan, X., Luo, P.: Amos: A large-scale abdominal multi-organ benchmark for versatile medical image segmentation. Advances in Neural Information Processing Systems **35**, 36722–36732 (2022) 7
12. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. In: 2017 IEEE International Conference on Computer Vision (ICCV). pp. 2999–3007 (2017). https://doi.org/10.1109/ICCV.2017.324 8, 9, 10
13. Ma, J., He, J., et al.: Unleashing the strengths of unlabelled data in deep learning-assisted pan-cancer abdominal organ quantification: the flare22 challenge. The Lancet Digital Health **6**(11), e815–e826 (2024). https://doi.org/10.1016/S2589-7500(23)00170-4 1

14. Ma, J., He, Y., Li, F., Han, L., You, C., Wang, B.: Segment anything in medical images. Nature Communications **15**, 654 (2024) 7
15. Ma, J., Yang, Z., Kim, S., Chen, B., Baharoon, M., Fallahpour, A., Asakereh, R., Lyu, H., Wang, B.: Medsam2: Segment anything in 3d medical images and videos. arXiv preprint arXiv:2504.03600 (2025) 7
16. 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) 7
17. 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. Lancet Digital Health (2024) 7
18. Ma, J., Zhang, Y., Gu, S., Ge, C., Wang, E., Zhou, Q., Huang, Z., Lyu, P., He, J., Wang, B.: Automatic organ and pan-cancer segmentation in abdomen ct: the flare 2023 challenge. arXiv preprint arXiv:2408.12534 (2024) 7
19. 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) 7
20. Shazeer, N.: GLU variants improve transformer. CoRR **abs/2002.05202** (2020) 4
21. 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) 7
22. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 2818–2826 (2016). https://doi.org/10.1109/CVPR.2016.308 3
23. Tompson, J., Goroshin, R., Jain, A., LeCun, Y., Bregler, C.: Efficient object localization using convolutional networks. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 648–656 (2015). https://doi.org/10.1109/CVPR.2015.7298664 4
24. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L.u., Polosukhin, I.: Attention is all you need. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 30. Curran Associates, Inc. (2017) 4
25. Wang, E., Zhao, Y., Wu, Y.: Cascade dual-decoders network for abdominal organs segmentation. In: Ma, J., Wang, B. (eds.) Fast and Low-Resource Semi-supervised Abdominal Organ Segmentation, Lecture Notes in Computer Science, vol. 13816. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-23911-3_18 2

26. Wang, Z., Popordanoska, T., Bertels, J., Lemmens, R., Blaschko, M.B.: Dice semimetric losses: Optimizing the dice score with soft labels. In: Greenspan, H., et al. (eds.) Medical Image Computing and Computer Assisted Intervention – MICCAI 2023. Lecture Notes in Computer Science, vol. 14222, pp. 479–489. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-43898-1_46 8, 9, 10
27. 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) 7
28. Wu, Y., He, K.: Group normalization. In: Computer Vision – ECCV 2018: 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XIII. p. 3–19. Springer-Verlag, Berlin, Heidelberg (2018). https://doi.org/10.1007/978-3-030-01261-8_1 3, 4
29. Xu, Z., Escalera, S., Pavão, A., Richard, M., Tu, W.W., Yao, Q., Zhao, H., Guyon, I.: Codabench: Flexible, easy-to-use, and reproducible meta-benchmark platform. Patterns **3**(7), 100543 (2022) 15
30. 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) 7

Table 10. Checklist Table. Please fill out this checklist table in the answer column.

Requirements	Answer
A meaningful title	Yes
The number of authors (≤6)	2
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	Page 2
Strategies to improve model inference	Page 3
Post-processing	Page 5
The dataset and evaluation metric section are presented	Page 7
Environment setting table is provided	Table 4
Training protocol table is provided	Table 5, 6
Ablation study	Page 10
Efficiency evaluation results are provided	Table 9
Visualized segmentation example is provided	Fig. 2
Limitation and future work are presented	Yes
Reference format is consistent	Yes