RECIST-Anchored Lightweight 3D Lesion Segmentation for CPU-only Inference

Zhou Jiang[1[0000−0001−8146−2028]] and Lisheng Wang[1[0000−0003−3234−7511]]

School of Automation and Intelligent Sensing, Shanghai Jiao Tong University, Shanghai 200240, China
Corresponding author: Lisheng Wang, lswang@sjtu.edu.cn

Abstract. RECIST-to-3D lesion segmentation aims to generate a volumetric lesion mask from a CT volume and a sparse 2D RECIST diameter marker. In Subtask 2 of the Pan-cancer Segmentation challenge, this task must be performed under strict CPU-only inference constraints, requiring both accurate prompt utilization and efficient implementation. Based on the public Lite ENSAM implementation and checkpoint, we fine-tune the lightweight model on the released training data and select the best validation checkpoint. We further introduce a RECIST-anchored inference strategy that treats the diameter marker as a lesion-specific geometric prior. Multi-label RECIST annotations are parsed into per-lesion prompts, localized prediction is performed around the marker, and RECIST-aware post-processing is applied to refine the final mask. The post-processing step merges the RECIST marker and its 3D radius-1 neighborhood as foreground seeds, retains connected components intersecting the corresponding marker, and performs hole filling within a local bounding box to avoid expensive full-volume operations. On 49 public validation cases, the proposed method achieved 0.7615 DSC and 0.8326 NSD, outperforming our same-machine rerun of the public Lite ENSAM checkpoint by 0.0623 DSC and 0.0855 NSD. The mean CPU inference time was 3.29 s per case, with no case exceeding the 60 s failure threshold. These results show that RECIST-anchored geometric constraints can improve prompt-to-3D lesion segmentation while maintaining efficient CPU-only deployment. Code is available at https://github.com/zhoujiang18/recist-anchor.

Keywords: RECIST-guided segmentation · 3D lesion segmentation · Pan-cancer CT segmentation · Lightweight medical image segmentation · Localized post-processing · CPU-only inference

1 Introduction

1.1 Background and challenge

Accurate lesion segmentation in volumetric CT images is important for tumor burden assessment, treatment planning, and longitudinal response evaluation. In routine clinical oncology, however, lesion size is often summarized using RECIST measurements [4], where a radiologist annotates a 2D diameter on the

axial slice that best represents the lesion. Such measurements are easy to acquire and remain widely used, but they are only a sparse description of a 3D object. Large-scale lesion resources such as DeepLesion have also shown that clinically recorded lesion diameters are valuable weak spatial annotations for learning lesion localization [20]. Nevertheless, a diameter line cannot directly capture irregular boundaries, through-plane extent, partial-volume effects, or multiple nearby components.

Subtask 2 of the Pan-cancer Segmentation challenge focuses on this practical gap: given a 3D CT image and a 2D RECIST diameter marker, a method must produce a 3D lesion mask. The task differs from fully automatic whole-body tumor segmentation because the RECIST marker provides a strong spatial prompt. It also differs from interactive segmentation because the inference must run on a laptop-like CPU-only environment. The challenge therefore requires a careful balance between segmentation accuracy, prompt utilization, memory footprint, and runtime. A method that predicts good masks but uses slow full-volume operations may fail individual cases once the per-case runtime exceeds 60 s.

1.2 Related work

General medical image segmentation has been strongly influenced by U-Net-style encoder-decoder architectures [17], volumetric extensions such as 3D U-Net [2], Dice-based 3D networks such as V-Net [14], and self-configuring pipelines such as nnU-Net [5]. In abdominal and pan-cancer CT segmentation, FLARE challenges have highlighted the importance of computational efficiency, pseudo labeling, and robust post-processing [11,12,13]. Foundation and promptable segmentation models, from the original Segment Anything Model [6] to medical adaptations such as MedSAM and MedSAM2 [9,10], further motivate the use of sparse prompts to localize targets.

For the present task, Lite ENSAM [1] provides a particularly relevant starting point because it is lightweight and already supports prompt-like bounding-box information derived from lesion annotations. It was designed for efficient cancer segmentation in 3D CT and reports strong performance under limited resources. Nevertheless, directly applying a public checkpoint is not sufficient for this task: the public validation data, the containerized evaluation protocol, multi-lesion RECIST labels, and strict CPU runtime constraint expose failure modes that require additional checkpoint selection, prompt-consistent post-processing, and efficient implementation.

1.3 Motivation and contributions

Our main motivation is to exploit the fact that the RECIST marker is not merely an auxiliary annotation; it is a reliable geometric anchor for the target lesion. A plausible 3D lesion mask should contain the marker, remain spatially close to it, and suppress components that are disconnected from the prompted lesion. These requirements suggest a different design principle from generic lesion

segmentation: instead of treating post-processing as a class-agnostic clean-up step, the RECIST marker can be used as a lesion-specific geometric constraint.

Building on this principle, we adapt the public Lite ENSAM checkpoint through supervised fine-tuning and validation-based checkpoint selection, rather than using the released model unchanged. The methodological novelty lies in the RECIST-anchored inference formulation: multi-label diameter markers are parsed as lesion-specific prompts, predicted components are selected according to their spatial relationship with the corresponding RECIST marker, and the RECIST line together with its 3D radius-1 neighborhood is introduced as a foreground seed before hole filling. This design improves prompt-neighborhood recall while retaining the irregular shape predicted by the network. To make this geometry practical for large CT scans, the computationally critical connected-component and hole-filling operations are restricted to a local bounding box around the predicted lesion and RECIST marker. This local formulation preserves the segmentation mask, avoids unnecessary full-volume operations, and keeps inference within the CPU-only runtime constraint.

2 Method

Fig. 1 summarizes the proposed CPU-only inference framework. The method follows a single-model design: one fine-tuned lightweight network generates lesion logits, and all subsequent operations are deterministic functions of the RECIST marker and the predicted mask. This separation keeps the learned component compact while allowing the sparse clinical measurement to guide the final 3D topology.

2.1 Pre-processing

The challenge organizers provide preprocessed NPZ files, so we avoid expensive image preprocessing during inference. Each input contains a CT volume in the key `imgs`, RECIST annotations in `recist`, and spatial metadata. The image intensities have already been mapped to the range $[0, 255]$, and the array layout is (Z, Y, X). During containerized inference, only keys required for prediction are loaded: image, RECIST marker, spacing, direction, origin, and optional prompt-related keys. The ground-truth mask key is deliberately ignored during inference.

RECIST markers may contain multiple lesion labels. We therefore parse nonzero RECIST voxels label by label. For each label, the marker voxels are obtained with `np.argwhere` and grouped by their RECIST label. The implementation uses the lexicographically first and last nonzero voxels of each straight single-slice marker as the two line extremes, and these endpoints are used as the geometric prompt for the model. If a case contains no RECIST marker, the method returns an empty mask. This also provides a simple false-positive control for healthy or no-prompt cases.

No additional resampling is performed in the submitted configuration. This choice avoids interpolation cost and potential shape mismatches. Surface-distance

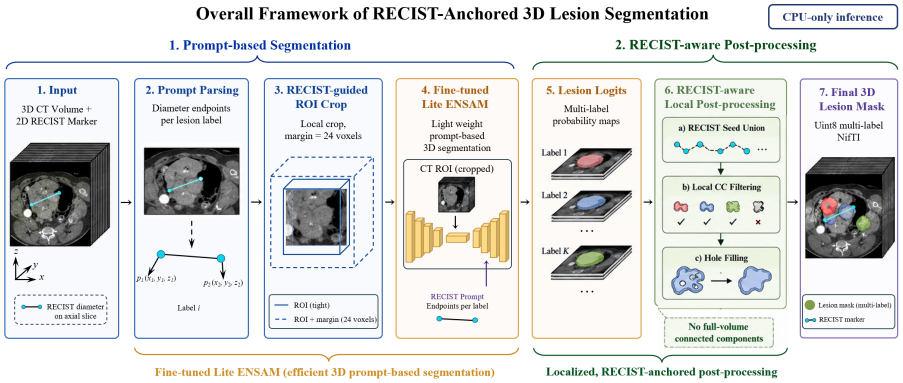

Fig. 1. Overall framework of the proposed RECIST-anchored CPU-only inference method. The input CT volume and RECIST marker are loaded from NPZ files. RECIST markers are converted into per-lesion diameter prompts and used to define a local ROI. A fine-tuned Lite ENSAM model predicts lesion logits, after which RECIST seed union, local connected-component filtering, and hole filling produce the final uint8 multi-label NIfTI mask.

evaluation still uses the spacing stored in the NPZ file. The method therefore operates in the provided voxel grid and relies on RECIST-localized inference to limit the region of interest.

2.2 Network and prompt-based prediction

The segmentation backbone is based on Lite ENSAM [1], a compact 3D CT lesion segmentation model with a SegResNet-style encoder [15] and prompt-conditioned decoder. In our submitted model, the instantiated network contains approximately 1.32 million trainable parameters. We keep the model lightweight because Subtask 2 evaluates inference on CPU without GPU acceleration.

The network receives a local CT crop and RECIST-derived prompts. The prompt is represented through diameter endpoints and propagated through the decoder as bounding-box-like spatial information. For network inference, the RECIST-derived prompt box is cropped with a 24-voxel margin in each dimension before downsampling and padding. This crop margin is used only to define the model input region and is distinct from the one-voxel local bounding-box expansion used later for post-processing. This is important because the task is prompt-to-mask rather than unconstrained whole-body segmentation. The model predicts a logit map for each lesion label in the crop. The crop is then mapped back to the original CT volume.

We considered adding the RECIST marker as an explicit second image channel. However, this would require modifying the first convolutional layer, adapting pretrained weights, retraining the image encoder, and revalidating the containerized inference path. Since the public Lite ENSAM architecture already

uses RECIST-derived box prompts, we retained the original prompt path and focused on robust training selection and inference-time RECIST geometry.

2.3 Training and checkpoint selection

We initialized from the public Lite ENSAM checkpoint and performed supervised fine-tuning on the provided training data. The training objective follows the public implementation and uses a Dice plus cross-entropy compound loss, implemented as DiceCE loss [8]. The optimizer is the AdamMuon variant used in the Lite ENSAM training code, with AdamW-style updates for non-matrix parameters [7]. We searched over random seeds, learning rates, and epoch sizes. The best checkpoint was selected by validation DSC/NSD. The public validation set was therefore used as a development set for checkpoint and module selection; the reported validation results document development-stage performance before hidden-test evaluation.

The final selected checkpoint used seed 2032, learning rate 1×10^{-4}, batch size 1, and 12,000 fine-tuning iterations. This was implemented as two fixed-length pseudo-epochs with 6000 iterations per epoch. Additional low-learning-rate fine-tuning from the selected checkpoint, decoder/prompt-only fine-tuning, boundary-loss fine-tuning, and intensity-augmentation fine-tuning were also tested, but none improved the final validation score. We therefore used the best validation checkpoint rather than the latest checkpoint.

Unlabeled images and partially labeled data were not used in the final method. Pseudo labels generated by previous FLARE winning algorithms were not used. Algorithm distillation was not used. These choices simplify the proposed method and avoid introducing uncertain labels close to the submission deadline.

2.4 RECIST-aware post-processing

The final post-processing is designed around the RECIST marker rather than around global foreground probability alone. For each nonzero RECIST label, the corresponding predicted mask is processed independently. The RECIST marker itself and its 3D radius-1 neighborhood are first merged into the predicted binary mask for that label. This *RECIST seed-union* rule encodes the assumption that the annotated diameter should lie inside the target lesion, improves recall around the most reliable foreground evidence, and does not require ground truth or an additional model.

The procedure can be written compactly as follows. Let $P_l \subset \Omega$ denote the predicted foreground voxels for lesion label l, and let $R_l \subset \Omega$ denote the corresponding RECIST marker in the image domain Ω. We construct a RECIST-augmented seed mask and a local processing domain as

$$S_l = P_l \cup \delta_1(R_l), \tag{1}$$

$$\Omega_l = \mathrm{bbox}\,(S_l \cup R_l) \oplus 1, \tag{2}$$

where $\delta_1(\cdot)$ is a 3D binary dilation with a radius of one voxel, $\mathrm{bbox}(\cdot)$ denotes the tight bounding box, and $\oplus 1$ expands it by one voxel in each direction.

Connected components are then computed on the union mask, following the classical connected-component labeling principle in digital picture processing [18]. Components that intersect the corresponding RECIST marker are retained, while components disconnected from the marker are discarded as likely false positives. If no component intersects the marker, the largest component is retained as a conservative fallback. Binary hole filling is subsequently applied to the retained local mask to remove small internal gaps. The result is written back using the original lesion label, so multi-lesion cases keep labels such as 1, 2, 3, and 4 instead of being collapsed into a binary mask.

Formally, if $\{C_{l,k}\}_{k=1}^{N_l}$ are the connected components of S_l within Ω_l, the retained component index set and final mask are

$$\mathcal{I}_l = \begin{cases} \{k \mid C_{l,k} \cap R_l \neq \emptyset\}, & \text{if such components exist,} \\ \{\arg\max_k |C_{l,k}|\}, & \text{otherwise,} \end{cases} \tag{3}$$

$$\hat{Y}_l = \mathrm{Fill}\left(\bigcup_{k \in \mathcal{I}_l} C_{l,k}\right). \tag{4}$$

where $\mathrm{Fill}(\cdot)$ denotes binary hole filling. The complete multi-label output is obtained by assigning label l to voxels in $\hat{Y}_l$ for each RECIST label. Labels are processed in ascending numerical order; if masks from different RECIST labels overlap after post-processing, the later label overwrites the earlier one, which provides a deterministic conflict-resolution rule matching the implementation.

A direct full-volume implementation of this RECIST-aware post-processing can be slow for thick CT scans. For example, a validation case of shape $(554, 512, 512)$ caused a severe runtime bottleneck when connected-component analysis was applied to the full volume. For local post-processing, we compute the tight bounding box of the union between the seeded prediction and RECIST marker, expand it by one voxel as in Eq. (2), and restrict connected-component analysis and hole filling to that local box. This one-voxel expansion is separate from the 24-voxel margin used for network-input cropping. The local-bounding-box implementation produced zero voxel differences compared with the full-volume version while reducing the runtime of the largest case from hundreds of seconds on a slower CPU to approximately 10 s. The acceleration is therefore algorithmically lossless for the tested validation cases.

Operationally, the localized post-processing has five deterministic steps. First, the raw multi-label prediction is split according to the nonzero RECIST labels. Second, for each lesion label, the RECIST diameter voxels and their 3D radius-1 neighborhood are inserted as foreground seeds. Third, a local bounding box is computed from the union of the seeded prediction and the RECIST marker, so all expensive connected-component operations are restricted to a small ROI. Fourth, only connected components that intersect the corresponding RECIST marker are retained; disconnected components are removed as remote false pos-

itives. Fifth, hole filling is applied inside the local ROI, and the retained mask is written back to the full image with its original lesion label.

3 Experiments

3.1 Dataset and evaluation measures

The challenge organizers curate CT scans from multiple public and licensed sources and release them in a unified NPZ format for Subtask 2. Each case provides a 3D CT image, a 2D RECIST diameter marker, and spatial metadata as input, and the expected output is a 3D lesion segmentation mask. In this work, we use only the official files released for Subtask 2 and do not introduce additional private data.

We evaluated on the 49 public validation NPZ cases available locally. The two accuracy metrics are Dice Similarity Coefficient (DSC) [3] and Normalized Surface Dice (NSD), a spacing-aware surface-overlap metric related to the surface Dice formulation used for clinically oriented segmentation assessment [16]. DSC measures volumetric overlap:

$$\mathrm{DSC}(G, S) = \frac{2|G \cap S|}{|G| + |S|}, \tag{5}$$

where G is the ground truth and S is the predicted segmentation. NSD measures the fraction of surface elements that are within a tolerance distance from the opposite surface. We used the challenge-style spacing-aware NSD implementation with a 2 mm tolerance. For multi-lesion cases, metrics were computed per nonzero ground-truth label and averaged.

Efficiency is also part of the challenge ranking, consistent with the broader view that biomedical challenge rankings should consider multiple task-relevant metrics rather than a single accuracy number [19]. The Task 2 Docker is evaluated without GPU and with an 8 GB memory limit. A case exceeding 60 s is considered failed. Running times below 45 s are within the official tolerance.

3.2 Implementation details

Table 1 summarizes the development environment used for local experiments. Training and checkpoint search were performed on GPUs, while all validation results for the submitted method used CPU-only inference. The container entry point explicitly sets `CUDA_VISIBLE_DEVICES=""`.

Table 2 lists the main training protocol for the selected checkpoint. We report only settings used by the final selected training run. Other exploratory runs are discussed in Sec. 4.2.

Table 1. Development environment and final inference setting.

Operating system	Ubuntu Linux, kernel 5.4
CPU for local validation	Intel Core i9-10920X, 12 cores / 24 threads
RAM	128 GB local memory; container tested with 8 GB limit
GPU for training/search	2 NVIDIA GeForce RTX 3090 GPUs, 24 GB each
Inference GPU	None for submitted inference
Programming language	Python 3.8
Framework and libraries	PyTorch 1.12, NumPy 1.23, SciPy 1.8, NiBabel 5.1
Output format	uint8 NIfTI files with multi-label lesion masks

Table 2. Training protocol for the selected checkpoint.

Network initialization	Public Lite ENSAM checkpoint
Training strategy	Supervised fine-tuning / checkpoint selection
Batch size	1
Total epochs	2 fixed-length pseudo-epochs
Total fine-tuning iterations	12,000
Epoch size	6000 iterations
Optimizer	AdamMuon / AdamW-style optimizer from Lite ENSAM code
Learning rate	1×10^{-4}
Weight decay	1×10^{-4}
Loss function	DiceCE loss
Model selection	Best validation Dice checkpoint
Number of parameters	1.32M
Unlabeled data	Not used
Pseudo labels	Not used

4 Results and Discussion

4.1 Quantitative validation results

Table 3 compares the original Lite ENSAM reference, our same-machine rerun of the public Lite ENSAM checkpoint, and the main stages of the proposed method. The reported Lite ENSAM score is included only as an external reference from the public release, because differences in validation files, RECIST handling, runtime environment, and metric implementation may affect direct comparability. The same-machine rerun is used as the primary fair baseline, because it uses the same 49 public validation NPZ files, the same workstation, the provided `recist` marker key as input, the same container setting, and the same metric implementation as our submitted method.

The final method achieved 0.7615 DSC and 0.8326 NSD, compared with 0.6992 DSC and 0.7471 NSD for the same-machine Lite ENSAM rerun. This corresponds to absolute improvements of 0.0623 DSC and 0.0855 NSD, or relative improvements of 8.9% and 11.4%, respectively. The improvement over the fine-tuned model alone is also consistent: RECIST-aware post-processing increases DSC from 0.7424 to 0.7615 and NSD from 0.8005 to 0.8326, corresponding to relative gains of 2.6% and 4.0%. We therefore use the controlled same-machine rerun and the fine-tuned-only checkpoint as the main evidence for the proposed adaptation; the public Lite ENSAM report is retained in the table as contextual information rather than as the primary comparison protocol.

Table 3. Quantitative results on the public validation set. Metrics are averaged over 49 validation cases. The public Lite ENSAM report is included for external reference; the same-machine rerun is the primary controlled baseline.

Method	Evaluation setting	DSC	NSD	Mean CPU time
Lite ENSAM public report [1]	Reported by authors	0.7606	0.7899	–
Lite ENSAM public checkpoint	Same-machine rerun, actual RECIST	0.6992	0.7471	4.03 s
Fine-tuned selected model only	Same-machine containerized evaluation	0.7424	0.8005	3.98 s
RECIST CC + hole filling, network ROI margin 24	Same-machine containerized evaluation	0.7558	0.8185	8.83 s
Final method with local post-processing	Same-machine containerized evaluation	**0.7615**	**0.8326**	**3.29 s**

Fig. 2 visualizes the same comparison. The performance progression shows that the gain is not produced by a single implementation detail: supervised fine-tuning improves the baseline model, while RECIST-aware component selection and seed union further refine the spatial support of the lesion. The larger relative improvement in NSD than DSC suggests that the marker-anchored post-processing reduces boundary errors and disconnected false positives, both of which are strongly reflected by surface-based evaluation.

4.2 Ablation study

Table 4 summarizes the module-level ablation of the final method. Starting from the selected fine-tuned model output, RECIST-aware connected-component selection and hole filling improved DSC from 0.7424 to 0.7558 and NSD from

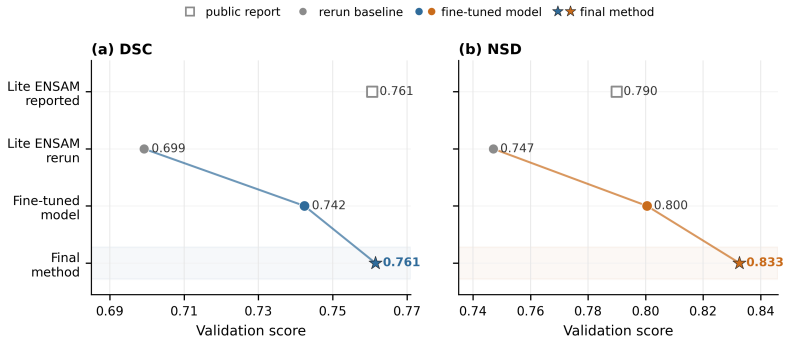

Fig. 2. Performance progression on the public validation set. Hollow squares denote the public Lite ENSAM report, gray circles denote the rerun baseline, colored circles denote the fine-tuned model, and stars denote the final method. The proposed method improves both DSC and NSD compared with the rerun baseline and the fine-tuned checkpoint alone.

0.8005 to 0.8185, giving absolute gains of 0.0134 DSC and 0.0180 NSD. Adding the 3D radius-1 RECIST seed union further improved DSC to 0.7615 and NSD to 0.8326, showing that explicitly enforcing the RECIST neighborhood as foreground recovers useful lesion voxels near the prompt. The final local-bounding-box implementation did not change the output masks; it produced zero voxel differences compared with the full-volume version. However, it reduced mean CPU time from 8.94 s to 3.29 s, a 63.2% reduction, and reduced the maximum official-like runtime from 27.17 s to 8.86 s, a 67.4% reduction.

Table 4. Module-level ablation on the public validation set. Metrics are averaged over 49 validation cases.

Configuration	DSC	NSD	Mean CPU time	Main effect
Fine-tuned selected model only	0.7424	0.8005	3.98 s	Network prediction baseline
+ RECIST CC + hole filling, network ROI margin 24	0.7558	0.8185	8.83 s	Suppress remote false positives
+ 3D radius-1 RECIST seed union	**0.7615**	**0.8326**	8.94 s	Improve prompt-neighborhood recall
+ Local-bounding-box post-processing	**0.7615**	**0.8326**	**3.29 s**	Preserve masks while accelerating CPU inference

Fig. 3 shows the same module ablation visually. The accuracy gain comes mainly from RECIST-anchored component selection and seed union, whereas the local-bounding-box implementation is a mask-preserving efficiency improvement. This separation is useful for interpreting the method: the RECIST geometry improves segmentation quality, while the localized implementation makes the same geometry feasible under strict CPU inference constraints.

We also tested several variants that were not retained. A 3× test-time augmentation run obtained 0.7417 DSC and 0.7974 NSD, a two-model ensemble obtained 0.7383 DSC and 0.7953 NSD, and a weighted ensemble with RECIST filling obtained 0.7469 DSC and 0.8089 NSD. These results suggest that sim-

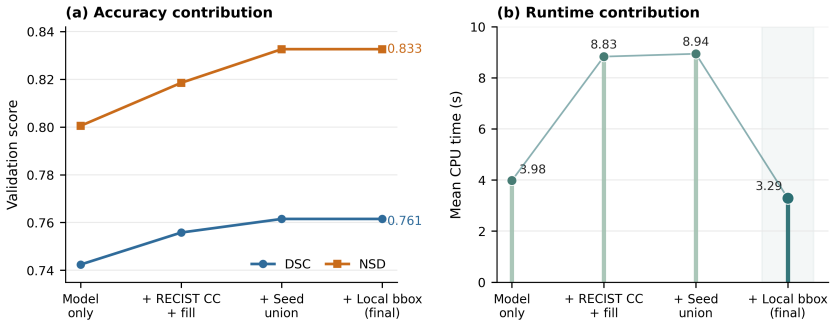

Fig. 3. Module ablation on the public validation set. Panel (a) shows the DSC/NSD contribution of each module, and panel (b) shows the corresponding mean CPU runtime. RECIST-aware post-processing improves segmentation accuracy, while local-bounding-box post-processing preserves the final mask and substantially reduces CPU runtime.

ply averaging predictions does not address the dominant error mode in this task, which is spatial inconsistency with the RECIST marker. Boundary-loss fine-tuning reached 0.7352 DSC and 0.8000 NSD, while decoder/prompt-only low-learning-rate fine-tuning reached 0.7233 DSC and 0.7880 NSD. Ellipsoid constraints derived from the RECIST diameter, volume guards, slice-wise propagation, adaptive crop margins, and morphological smoothing also did not consistently improve both DSC and NSD. In practice, hard geometric priors can over-constrain irregular lesions. The softer strategy of retaining components anchored at the RECIST marker while preserving the network-predicted shape was more robust.

4.3 Qualitative validation results

Fig. 4 shows two high-scoring cases and two difficult cases. Cases 0097 and 0027 illustrate compact lesions whose RECIST marker lies near the central slice and whose through-plane extent is well captured by the ROI crop. Cases 0069 and 0048 illustrate more difficult situations with elongated structures, nearby high-contrast anatomy, and boundary disagreement between prediction and ground truth. Even in these difficult cases, the RECIST marker helps keep the prediction spatially aligned with the target lesion and suppresses remote false positives.

4.4 Segmentation efficiency

Table 5 reports representative CPU-only runtimes for cases of different depths. Since Task 2 forbids GPU use during inference, GPU memory is zero by design. The final evaluation was performed with an 8 GB container memory limit. All representative cases are well below the 60 s failure threshold.

Fig. 5 shows the full 49-case runtime distribution from the validation sanity run. The maximum case runtime in the full sanity video was 9.87 s, and the

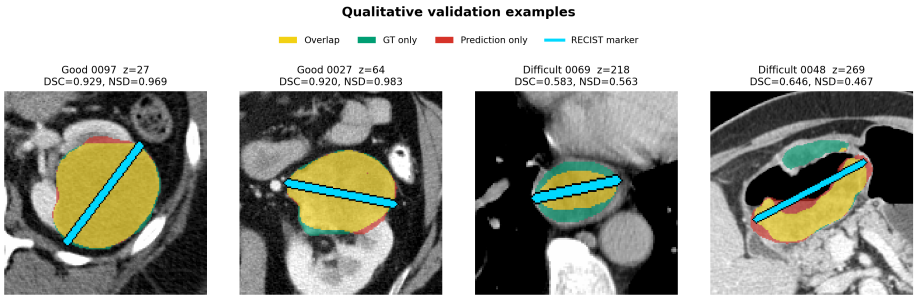

Fig. 4. Qualitative examples on public validation cases. Yellow denotes overlap between prediction and ground truth, green denotes ground truth only, red denotes prediction only, and cyan denotes the RECIST marker, drawn with a black outline for visibility. The first two cases are high-scoring examples, while the last two illustrate typical difficult cases with boundary disagreement or partial under-segmentation.

Table 5. Representative CPU-only efficiency results on public validation cases. Image size is reported as (Z, Y, X).

Case ID	Image size	Runtime (s)	GPU memory
0001	$(55, 512, 512)$	0.76	0 MB
0051	$(100, 512, 512)$	3.13	0 MB
0017	$(151, 512, 512)$	3.84	0 MB
0019	$(215, 512, 512)$	4.11	0 MB
0099	$(334, 512, 512)$	7.26	0 MB
0063	$(448, 512, 512)$	8.03	0 MB
0048	$(499, 512, 512)$	8.11	0 MB
0029	$(554, 512, 512)$	8.86	0 MB

maximum runtime in the official-like metric run was 8.86 s. The difference comes from repeated Docker runs and normal CPU variation. In both settings, every case remained far below 45 s and 60 s.

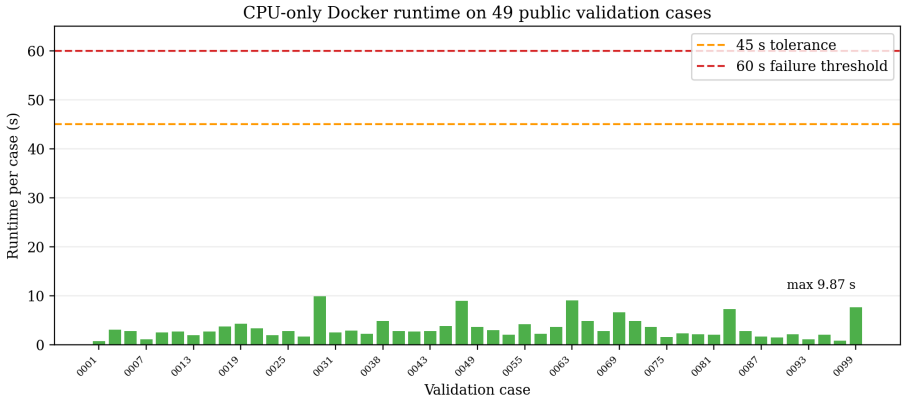

Fig. 5. CPU-only runtime on the public validation set. Dashed lines mark the 45 s tolerance and 60 s failure threshold. No case exceeded either threshold.

4.5 Limitations and future work

The method remains dependent on the quality and location of the RECIST marker. If the marker is incomplete, mislabeled, or lies far from the true lesion center, the ROI and component-selection logic may bias the final mask. The deterministic seed-union rule can also slightly enlarge the lesion around the RECIST line, which is usually helpful but may be suboptimal for very thin or ambiguous lesions. Another limitation is that unlabeled and partially labeled scans were not exploited; semi-supervised learning could improve generalization if implemented carefully. Finally, we did not modify the network to accept RECIST as a second image channel, because this would require a more invasive retraining and validation process.

Future work will investigate prompt-aware image encoders, uncertainty-based pseudo labels, stronger lesion-size-aware augmentations, and adaptive post-processing rules that account for the physical RECIST diameter and anisotropic slice spacing without over-constraining irregular lesions.

5 Conclusion

We presented a lightweight RECIST-to-3D lesion segmentation method for CPU-only inference. The final approach combines a fine-tuned Lite ENSAM checkpoint

with RECIST-aware ROI inference, seed-union post-processing, local connected-component filtering, and hole filling. The central observation is that the RECIST marker can serve as a lesion-specific geometric prior: it identifies where the target component should be, supplies reliable foreground seed voxels, and enables efficient local post-processing. On 49 public validation cases, the submitted method improved over the same-machine Lite ENSAM rerun by 0.0623 DSC and 0.0855 NSD, corresponding to relative gains of 8.9% and 11.4%. The method kept the mean CPU runtime at 3.29 s and the maximum official-like runtime below 9 s. These results indicate that RECIST-anchored geometry can improve prompt-to-3D lesion segmentation accuracy without increasing model size or violating CPU-only deployment constraints.

Acknowledgements The authors thank the challenge organizers and data contributors for providing the Pan-cancer Segmentation benchmark. We used the public Lite ENSAM implementation and checkpoint as initialization and integrated it through supervised fine-tuning, checkpoint selection, and task-specific inference-time post-processing. No private external dataset, private pretrained model, pseudo label from previous FLARE winning algorithms, or algorithm distillation was used.

Disclosure of Interests

The authors declare no competing interests.

References

1. Bjørnstad, A.M., Stenhede, E., Ranjbar, A.: Lite ensam: a lightweight cancer segmentation model for 3d computed tomography. In: MICCAI 2025 FLARE Challenge (2025), https://openreview.net/forum?id=Ev0c6zrp9N 2, 4, 9
2. Çiçek, Ö., Abdulkadir, A., Lienkamp, S.S., Brox, T., Ronneberger, O.: 3d u-net: Learning dense volumetric segmentation from sparse annotation. In: International Conference on Medical Image Computing and Computer-Assisted Intervention. pp. 424–432. Springer (2016) 2
3. Dice, L.R.: Measures of the amount of ecologic association between species. Ecology **26**(3), 297–302 (1945) 7
4. Eisenhauer, E.A., Therasse, P., Bogaerts, J., Schwartz, L.H., Sargent, D., Ford, R., Dancey, J., Arbuck, S., Gwyther, S., Mooney, M., Rubinstein, L., Shankar, L., Dodd, L., Kaplan, R., Lacombe, D., Verweij, J.: New response evaluation criteria in solid tumours: Revised recist guideline (version 1.1). European Journal of Cancer **45**(2), 228–247 (2009) 1
5. 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) 2
6. Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A.C., Lo, W.Y., Dollár, P., Girshick, R.: Segment anything. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 4015–4026 (2023) 2

7. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. In: International Conference on Learning Representations (2019) 5
8. Ma, J., Chen, J., Ng, M., Huang, R., Li, Y., Li, C., Yang, X., Martel, A.L.: Loss odyssey in medical image segmentation. Medical Image Analysis **71**, 102035 (2021) 5
9. Ma, J., He, Y., Li, F., Han, L., You, C., Wang, B.: Segment anything in medical images. Nature Communications **15**, 654 (2024) 2
10. 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) 2
11. 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., Ăătern, 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) 2
12. 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) 2
13. 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) 2
14. Milletari, F., Navab, N., Ahmadi, S.A.: V-net: Fully convolutional neural networks for volumetric medical image segmentation. In: International Conference on 3D Vision. pp. 565–571 (2016) 2
15. Myronenko, A.: 3d mri brain tumor segmentation using autoencoder regularization. In: Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries, pp. 311–320. Springer (2019) 4
16. Nikolov, S., Blackwell, S., Zverovitch, A., Mendes, R., Livne, M., De Fauw, J., Patel, Y., Meyer, C., Askham, H., Romera-Paredes, B., Kelly, C., Karthikesalingam, A., Chu, C., Carnell, D., Boon, C., D'Souza, D., Moinuddin, S.A., Garie, B., McQuinlan, Y., Ireland, S., Hampton, K., Fuller, K., Montgomery, H., Rees, G., Suleyman, M., Back, T., Hughes, C.O., Ledsam, J.R., Ronneberger, O.: Clinically applicable segmentation of head and neck anatomy for radiotherapy: Deep learning algorithm development and validation study. Journal of Medical Internet Research **23**(7), e26151 (2021) 7
17. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical image computing and computer-assisted intervention. pp. 234–241 (2015) 2
18. Rosenfeld, A., Pfaltz, J.L.: Sequential operations in digital picture processing. Journal of the ACM **13**(4), 471–494 (1966) 6
19. Wiesenfarth, M., Reinke, A., Landman, B.A., Eisenmann, M., Aguilera Saiz, L., Cardoso, M.J., Maier-Hein, L., Kopp-Schneider, A.: Methods and open-source toolkit for analyzing and visualizing challenge results. Scientific Reports **11**, 2369 (2021) 7
20. Yan, K., Wang, X., Lu, L., Summers, R.M.: Deeplesion: automated mining of large-scale lesion annotations and universal lesion detection with deep learning. Journal of Medical Imaging **5**(3), 036501–036501 (2018) 2

Table 6. Checklist table.

Requirements	Answer
A meaningful title	Yes
The number of authors ($\leq$6)	2
Author affiliations and ORCID	Yes
Corresponding author email is presented	Yes
Validation scores are presented in the abstract	Yes
Introduction includes background, related work, and motivation	Sec. 1
A pipeline/network figure is provided	Fig. 1
Pre-processing	Sec. 2.1
Strategies to use the partial label	Secs. 2.1, 2.4
Strategies to use the unlabeled images	Not used in the final method
Strategies to improve model inference	Secs. 2.4, 4.4
Post-processing	Sec. 2.4
Dataset and evaluation metric section	Sec. 3.1
Environment setting table	Table 1
Training protocol table	Table 2
Ablation study	Sec. 4.2
Efficiency evaluation results	Table 5, Fig. 5
Visualized segmentation examples	Fig. 4
Limitation and future work are presented	Sec. 4.5
Reference format is consistent	Yes