Real-time MRI tumor tracking with SAM2, mostly automated finetuning, and nnU-Net postprocessing

Mike Zhu

Abstract. Magnetic resonance imaging (MRI)-guided radiotherapy is an emerging modality that can provide more accurate and personalized cancer treatment. Tracking moving objects and adjusting the X-ray beam in real-time allows the delivery of higher radiation doses to target sites while sparing healthy tissue. This project applies deep learning methods to real-time MRI tumor tracking. Our approach employed Segment Anything Model 2 (SAM2), a foundation vision model, and nnU-Net for postprocessing. For each cine-MR sequence, SAM2 uses the labeled first frame to predict tumor segmentation in subsequent frames. Mask refinement was performed using nnU-Net, encoding the images and predicted masks (2 images) for the current and previous four frames (five timepoints) as a 10-channel input to predict the final mask. Additionally, we leveraged SAM2 to label data. We developed software for users to draw bounding boxes around objects on the first frame of a sequence and used SAM2 used to infer segmentations across the sequence; this was applied to 120 unlabeled sequences to finetune SAM2. The nnU-Net was trained on 50 labeled sequences. We used SAM2's default image input size and training configuration including loss, data augmentation, optimizer, and learning rates. We also used the nnU-Net defaults. We added elastic deformation to those default augmentations to reflect soft tissue deformation. We did not use field strength, frame rate, or location in our model. We selected the model with the highest percentage improvement in the competition metrics relative to zero-shot SAM2.

1 Introduction

The use of magnetic resonance imaging (MRI) to guide radiotherapy has become increasingly prevalent. Hybrid machines that combine MRI with a linear accelerator, known as MR-Linacs, allow visualization of soft tissues during treatment delivery at frame rates ranging from 1 to 8 Hz [1]. Accurate, real-time tracking of tumors that move and deform throughout motion—including regular (e.g. respiratory or cardiac cycles), planned (voluntary breath holds), and irregular (peristalsis or patient shifts) motions—enables precise, targeted delivery of higher radiation doses in various disease sites [2].

Existing clinical software solutions that rely on image registration or template-matching approaches are limited by large, non-rigid motion. Therefore, the current standard practice is to use a gating procedure, turning off the X-ray beam entirely when large motion is detected. This reduces the duty cycle of the beam and extends treatment times [1]. Recently, the field of computer vision has seen rapid development of deep learning models for real-time video object tracking. Many groups have published large

off-the-shelf models for medical imaging. Such models may enhance the accuracy of real-time tumor tracking on cine-MR sequences.

Segment Anything Model 2 (SAM2) is a foundation model for object segmentation in images and videos [3]. For a given image, SAM2 predicts an object mask based on an input prompt, such as a bounding box. Prompts can also take the form of object masks from a previous frame, generalizing the model from image to video segmentation. Given a video sequence, and object masks for the first frame, SAM2 can infer masks for all subsequent frames. SAM2 was trained on a large corpus of images and videos and shows strong predictive accuracy over different domains.

Frameworks such as nnU-Net further allow for domain-specific fine-tuning. nnU-Net is an auto-configuring U-Net [4] known for its robust performance across multiple medical imaging applications [5]. nnU-Net automatically configures a segmentation pipeline based on dataset properties, including preprocessing steps, data augmentation, network architecture, and training hyperparameters. Models such as SAM2 and nnU-Net are attractive because they can be quickly applied to new tasks without the need for extensive hyperparameter tuning and domain expertise.

TrackRAD2025 is an open medical imaging competition for developing real-time MRI tumor tracking algorithms [1]. The competition provides a large dataset of sagittal 2D MR sequences from clinical MR-linacs used in multiple medical institutions across the world, and a platform to evaluate algorithm performance. Algorithms are evaluated based on mask prediction accuracy and runtime.

2 Methods

Our algorithm uses a fine-tuned SAM2 as the primary mask predictor, followed by a post-processing step that uses nnU-Net for mask refinement. We also developed a web-based labeling application to generate training data to fine-tune SAM2.

2.1 Data

Two training datasets were provided by the TrackRAD2025 challenge: one unlabeled and one labeled. The unlabeled dataset consists of over 2000 cine-MR (time-series) sequences from 477 patients, totaling over 2.8 million frames. The labeled dataset contains 50 sequences, one from each of 50 patients, totaling 5027 frames.

We labeled a subset of the unlabeled dataset to finetune SAM2. Given limited time and resources, we employed a semi-automated approach that leveraged SAM2 itself to generate most labels (see **Fig. 1**). We developed a web application using Gradio [6], a Python library for building user interfaces for machine learning applications, to allow users to label objects on one image from a cine-MR sequence. The application was designed to optimize labeling throughput by minimizing the number of user interactions. Upon page load, an MR sequence is randomly sampled and loaded from the

dataset. The application displayed an MR sequence and allowed users to draw one or more bounding boxes around objects of interest in the first frame. SAM2 was used to predict a segmentation mask from each bounding box and then propagate the predicted masks to all subsequent frames of the sequence. Upon label submission, the page automatically reloads with a new sequence. We deployed the application on HuggingFace Spaces [7], a cloud hosting service for machine learning applications. We used HuggingFace Datasets to store both the input MR images and the output labels.

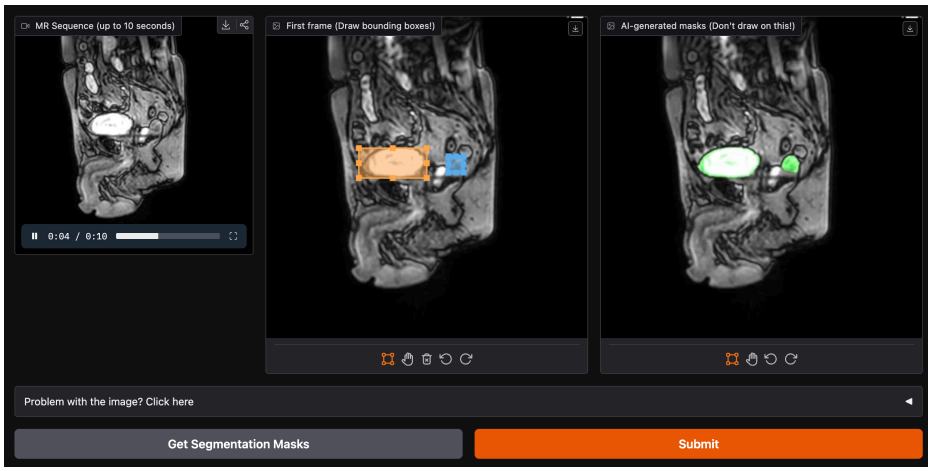

Fig. 1. Web application to visualize and label MR sequences. This image represents the user interface with (*left*) the cine-MR sequence as an interactive video, (*center*) two user-defined bounding boxes on two unique regions of interest, and (*right*) masks predicted by SAM2.

A small team, composed of medical students, a software engineer, and a medical physicist, contributed to labeling images. For the most part, the team did not have advanced training in identifying tumors in MR images. Therefore, we opted to label any objects of interest, and up to five per image, believing that this would improve the model's ability to segment generic objects in medical images. We labeled the first image from 200 cine-MR sequences from 140 unique patients, totaling 94,441 frames after masks were propagated by SAM2. Given time constraints, we did not perform validation of the SAM2-predicted object masks. We then used the 50 labeled sequences as the validation set to evaluate the fine-tuned SAM2 checkpoints.

We used the labeled dataset as ground truth to train the mask refinement U-Net with five-fold cross-validation to assess performance.

2.2 Model

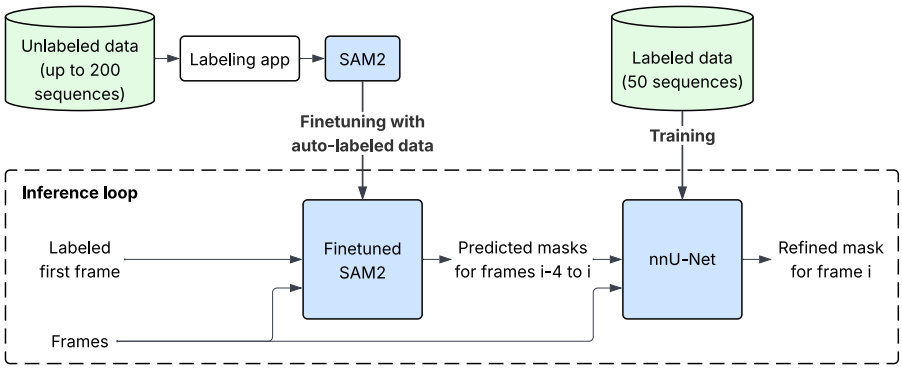

Fig. 1. Algorithm overview**.** A finetuned SAM2 makes an initial mask prediction, which nnU-Net then refines.

SAM2 consists of multiple transformer-based modules: the image encoder, prompt encoder, memory attention module, and mask decoder. The image encoder uses a hierarchical vision transformer [8] to extract multiscale features from images. The prompt encoder converts bounding boxes, points, and masks on the current frame to embeddings. The memory attention module uses transformer blocks with self-attention and cross-attention mechanisms and a streaming memory bank to condition current frame features on previous frames and mask predictions. This module allows SAM2 to learn information about short-term motion. Finally, the mask decoder combines features from the image embedding, the prompt encoder, and the memory attention module to generate mask predictions. Output features are stored in the streaming memory bank to condition future frames.

We used the updated SAM2.1 checkpoints, which were made publicly available in October 2024. Of the four checkpoints (Tiny, Small, Base-Plus, Large), we selected the Small variant, as it provided a good balance of accuracy and runtime on the labeled training dataset (see **Table 1.**). Sam2.1-Small has 46M total parameters [3].

For the mask refinement U-Net, we employed a sliding window approach to incorporate motion information. We encoded the image and predicted mask for the current (i) and previous four frames (i-1: i-4) in separate channels to create a 10-channel 2D input. The task was to predict the ground truth mask for the current frame. For the initial four frames, we repeated the first image and mask to fill all channels.

nnU-Net determines model architecture automatically based on dataset characteristics [5]. We used a 2D U-Net with a 384x384 pixel sliding patch. The U-Net had 7 stages, with 32, 64, 128, 256, 512, 512, and 512 features respectively, and a 3x3 kernel at each stage. Two convolutions were applied between each stage on both the encoding and decoding paths. Z-score normalization was applied to input at each stage.

The network has 92M trainable parameters. At inference time, we used a tile step size of 0.05 instead of the default 0.5 because this gave small improvements in accuracy at the cost of runtime. As the U-Net would occasionally generate outputs with holes, so we implemented a final binary hole-filling step using the scipy library [9].

SAM2 and nnU-Net are both developed on the PyTorch library [10].

2.3 Training

SAM2 and nnU-Net were trained separately. We used SAM2's default training configuration [3]. Data augmentations include random horizontal flip, affine shift, resizing, color jitter, and grayscale conversion. For training on the mask prediction task, we combined focal loss and Dice loss weighted 20:1 and used the AdamW optimizer. We fine-tuned the full model using two learning rates: 5.0e-6 for the image encoder and 3.0e-6 for other components. We trained the model for 40 epochs on 2 Nvidia A100 GPUs with 80GB VRAM each for 2 hours total. Over the course of the competition, we performed fine-tuning three times as the team labeled more data. The loss curves did not converge during the training runs, though we did see empirical improvement when running the fine-tuned models on the validation set (see **Table 2**.). We selected the checkpoint that gave the greatest improvement on the accuracy metrics relative to zero-shot SAM2.1-Small.

Once we selected a checkpoint, we ran SAM2 to predict masks for the 50-sequence labeled training set. We used these predictions, along with the raw frames and ground truth labels, to train the mask refinement U-Net. By default, nnU-Net employs extensive data augmentation, including rotation, resizing, noise, blur, brightness, gamma, and contrast transforms [5]. We added elastic deformation with p=0.2 to reflect soft tissue deformation in the MR sequences. The p=0.2 value was selected based on similar default values for other data augmentations in the nnU-Net codebase. Otherwise, we used the default training configuration, using a combined Dice-CE loss weighted 1:1, SGD optimizer with a Nesterov momentum of 0.99, an initial learning rate of 1e−2, and a polyLR scheduler. The final nnU-Net model was trained for 1000 epochs, taking 22 hours on 6 A100 GPUs. We performed five-fold cross validation, using one GPU to train each fold and a sixth for the entire training set. We selected the model that gave the greatest improvement on the competition metrics, averaged across all folds, relative to the finetuned SAM2 described above.

2.4 Evaluation

We evaluated model performance by comparing predicted target segmentations against ground truth labels, using the four geometric plus one radiotherapy dose metrics specified by the challenge [1]. The geometric metrics are Dice similarity coefficient (Dice), the 95th percentile (P95) Hausdorff distance, the mean average surface distance, and the Euclidean center distance.

The dose metric (Dose) is an estimation of clinical radiotherapy dose delivery accuracy that accounts for X-ray scatter and tissue-specific dose fall-off [1]. To compute dose, the ground truth label of the first frame was converted to an approximated radiation therapy dose by applying a 3mm expansion of the gross tumor volume (GTV) indicated by the label to the clinical target volume (CTV). Then, this expansion was smoothed by a Gaussian of 6 mm standard deviation for targets in the lung (simulates a dose fall-off like those observed in clinical dose distributions for lung patients) and of 4 mm for all other targets (dose fall-off in higher density tissue). This dose distribution was then shifted by the distance between the ground truth centroid position of the tracking target and the centroid position obtained by the predicted segmentation for each frame. These shifted distributions were averaged to get a centroid-error shifted dose. The relative difference between the GTV (or tracking target) D98% (from the cumulative dose volume histogram) for the ground truth dose distribution and the final shifted dose distribution were calculated for each patient.

DSC and dose are dimensionless and range from 0 to 1, and the other metrics are measured in pixels. We also measured each model's runtime. Evaluation was performed on a single Nvidia A100 GPU with 80GB memory.

3 Results

We evaluated zero-shot and finetuned SAM2 on the 50-sequence labeled dataset provided by the challenge. All metrics other than dose and runtime are computed by first averaging over all the frames for a given MRI sequence, then averaging over all sequences. Dose and runtime are averaged over all frames.

Table 1. Zero-shot predictive accuracy of SAM2.1 checkpoints.

SAM2.1 Checkpoint	**Dice**	**P95 Hausdorff distance (pixels)**	**Mean average surface distance (pixels)**	**Center distance (pixels)**	**Dose**	**Runtime per frame (s)**
Tiny	0.9072	3.543	1.302	1.593	0.9444	0.0190
Small	0.9158	3.324	1.198	1.390	0.9577	0.0199
Base plus	0.9150	3.510	1.310	1.500	0.9587	0.0240
Large	0.9163	3.266	1.197	1.356	0.9621	0.0356

Table 2. Predictive accuracy of SAM2.1 after finetuning. Checkpoints are named for the number of MR sequences used for training.

SAM2.1 Checkpoint	Total frames	Dice	P95 Hausdorff Distance (pixels)	Mean average surface distance (pixels)	Center distance (pixels)	Dose
Small	0	0.9158	3.324	1.198	1.390	0.9577
Small-40	9960	0.9192	3.212	1.157	1.325	0.9624
Small-120	**35,531**	**0.9195**	**3.187**	**1.149**	**1.306**	**0.9675**
Small-200	92,441	0.9187	3.231	1.161	1.331	0.9613

Based on these results, we selected the Small-120 checkpoint for further refinement. We ran inference on all 50 sequences of the labeled dataset. We then used the predicted masks to train and evaluate the mask refinement U-Net using five-fold cross-validation. Metrics in the table below are averaged over the five folds.

Table 3. Predictive accuracy and runtime before and after mask refinement.

Model	Dice	P95 Hausdorff Distance (pixels)	Mean average surface distance (pixels)	Center distance (pixels)	Dose	Runtime per frame (s)
Small-120	0.9195	3.187	1.149	1.308	0.9668	0.0199
Small-120 + nnU-Net	0.9197	3.066	1.116	1.258	0.9706	0.164
Percent change	**0.0218%**	**-3.64%**	**-2.75%**	**-3.45%**	**0.324%**	**724%**

The following two tables summarize how the finetuning and postprocessing steps improved accuracy relative to zero-shot SAM2.1-Small.

Table 4. Predictive accuracy and runtime after finetuning and postprocessing

Model	Dice	P95 Hausdorff Distance (pixels)	Mean average surface distance (pixels)	Center distance (pixels)	Dose	Runtime per frame (s)
Small zero-shot	0.9158	3.324	1.198	1.390	0.9577	0.0199
Small-120 finetune	0.9195	3.187	1.149	1.306	0.9675	0.0199
Small-120 finetune + nnU-Net	**0.9197**	**3.066**	**1.116**	**1.258**	**0.9706**	**0.164**

Table 5. Percent change in predictive accuracy and runtime relative to zero-shot SAM2

Model	Dice (%)	P95 Hausdorff distance (%)	Mean average surface distance (%)	Center distance (%)	Dose (%)	Runtime per frame (%)
Small-120 finetune	0.404	-4.12	-4.09	-6.04	1.02	0
Small-120 finetune + nnU-Net	**0.426**	**-7.76**	**-6.84**	**-9.50**	**1.35**	**724**

4 Discussion

4.1 Limitations

Data quality was a major limitation in the performance of our finetuning approach. No team members had advanced training in identifying tumors or labeling MR images. Therefore, we took a "quantity over quality" approach by identifying multiple objects in the frame (not exclusively tumors) and attempting to label as many sequences as possible. The idea was to improve the model's ability to track generic objects in medical images; however, this may have reduced the model's translatability to the tumor tracking task. While the finetuned model showed a modest increase in accuracy, training on high-fidelity labels, curated frame-by-frame by experts, would likely have had a larger effect[11].

Given limited time, we relied on SAM2 to generate data from manually-labeled first frames. We did not perform quality control or validation of the SAM2-generated masks before using them for training. The decrease in performance going from 120 to 200 sequences for finetuning is likely due to lower quality images in the

lattermost datasets. For example, the model may have lost track of objects midway through a video, resulting in erroneous segmentation masks. While it would be difficult to manually inspect each frame, we could have employed automated rules-based quality checks. For example, we could identify frames where an object undergoes extreme deformation or disappears entirely. Discarding low-quality frames, or entire sequences, may have resulted in a more accurate model. As a further extension, we could have implemented an active learning approach, where low-quality frames are flagged for manual relabeling. SAM2 supports prompting with multiple labeled frames, including future frames, which would make implementing this workflow feasible.

Although we saw some performance improvements in the finetuned SAM2 instances when evaluated on the labeled dataset, the loss curves never converged during the training process. Therefore, it is possible that we selected a model that happened to perform well on the dataset but does not generalize. Because the team does not have much experience with finetuning foundation models, it was difficult to determine in the moment whether hyperparameter tuning or dataset curation would be more effective.

Early in the project, we selected the SAM2.1-Small variant for finetuning because it balanced predictive accuracy with runtime. Because we later added a post-processing step that was much slower than SAM2, we could have revisited this choice. The Large variant outperformed the Small variant on all competition metrics, at a runtime cost of 0.0157 s/frame. While this is an 80% premium over SAM2.1-Small, it is only a marginal addition to the total algorithm runtime.

4.2 Unsuccessful methods

We experimented with several methods that did not improve model accuracy.

Before the team had labeled a significant number of MR sequences, we tried to finetune SAM2 on 21994 labeled images from Duke Liver, a publicly available dataset of segmented livers from traditional 3D MRI volumes [12]. The idea was to have the model learn domain features from medical imaging and MR specifically that could generalize to the challenge. However, this resulted in dramatically worse performance relative to zero-shot SAM2, with a Dice coefficient of ~0.8 when evaluated on the 50-sequence labeled training set. It is possible that 3D MR volumes are too different from 2D cine-MRI data for the model to generalize to the latter domain.

We also attempted to incorporate metadata about the MR sequence, such as magnetic field strength and scanned anatomical region, into our algorithm. We trialed a “mixture of experts” approach where multiple SAM2 instances were fine-tuned using data only for a given anatomical region or field strength, and a model was selected at inference time based on metadata. This also resulted in dramatically worse performance relative to zero-shot. Given the data quality issues discussed earlier, it is possible that stratifying the datasets only increases the impact of the low-quality segmentations.

For the postprocessing, nnU-Net, we initially used the residual encoder U-Net (ResEnc) architecture that has been recommended by the library authors since late 2024, but this resulted in overfitting. Using the "traditional" U-Net architecture from the earlier nnU-Net implementation showed improved performance in five-fold cross-validation. It is possible that our dataset was too small, and that the ResEnc architecture would perform better given a larger training set.

We also experimented with sliding window lengths of 10 and 15 frames. Compared to the final length of 5, this resulted in no significant performance improvement, but longer training times. This suggests that 5 frame captures sufficient motion information. Given the MR sequences' low frame rates of 1 to 8 Hz, 5 frames make up 0.5 to 2 seconds. In comparison, SAM2 uses a window of 17 frames and was trained on 24 to 60Hz videos [3].

We tried adding extensive MRI-specific data augmentation steps when training both SAM2 and nnU-Net to make the models generalize to different field strengths and scanning protocols, inspired by a similar approach used in an MRI spine segmentation tool [13]. These steps include simulated MRI motion artifacts, intensity variations due to magnetic field inhomogeneity, and Gaussian noise and blur, and were implemented using the torchio library [14]. Augmentations were applied with a high probability of p=0.7. These additions increased the training time of both models by up to 10X but did not result in meaningful accuracy improvements. For SAM2 it may be that applying data augmentation to already low-quality auto-labeled data would not result in more effective fine-tuning. For nnU-Net, we actually had to stop training early, as the augmentations increased total training time to over one week, which was too long for the project timeline. Given that others have demonstrated improved generalization using these extreme data augmentations, it is possible that the fully-trained network would have shown stronger performance [15].

4.3 Future work

Our labeling application, and the methods behind its development, could be worth further exploration as a lightweight and portable alternative to existing interactive medical image analysis software. Most of these tools, such as the widely-used 3D Slicer [16], are desktop-based. In contrast, our application is accessible via a web browser, with compute and data storage handled entirely in the cloud. This simplifies access, as users do not need to install software or powerful GPUs on their local machine. It also simplifies development, as a single version of the application is usable on Windows, Mac, and Linux machines. This approach is also potentially more secure, as it minimizes how much patient health data must be downloaded to the local machine.

Further development could include the active learning features discussed above, or extending the application to use other cloud services outside of HuggingFace, such as Amazon Web Services or Google Cloud Platform for compute and data storage.

By making heavy use of Gradio, an open-source library for building lightweight user interfaces, we were able to develop the application with very limited resources (one developer over two weeks). This model could be used by research groups or small development teams to quickly build bespoke user interfaces when off-the-shelf tools such as Slicer do not support a particular workflow.

5 Conclusion

This project applies off-the-shelf computer vision models to MRI tumor tracking. We show strong zero-shot performance by SAM2, which we improved through finetuning and refinement by nnU-Net. We also demonstrate the rapid development and deployment of a software application for medical image annotation. This application, accessible in a web browser and hosted in the cloud, could be a lightweight alternative to existing desktop-based solutions for medical imaging visualization and segmentation.

References

1. Wang, Y., Lombardo, E., Thummerer, A., Blöcker, T., Fan, Y., Zhao, Y., Papadopoulou, C.I., Hurkmans, C., Tijssen, R.H.N., Görts, P.A.W., Tetar, S.U., Cusumano, D., Intven, M.P.W., Borman, P., Riboldi, M., Dudáš, D., Byrne, H., Placidi, L., Fusella, M., Jameson, M., Palacios, M., Cobussen, P., Finazzi, T., Haasbeek, C.J.A., Keall, P., Kurz, C., Landry, G., Maspero, M.: TrackRAD2025 challenge dataset: Real-time tumor tracking for MRI-guided radiotherapy. Medical Physics. 52, e17964 (2025). https://doi.org/10.1002/mp.17964.
2. Magnetic Resonance-Guided Prostate Stereotactic Body Radiation Therapy With Daily Online Plan Adaptation: Results of a Prospective Phase 1 Trial and Supplemental Cohort - PubMed, https://pubmed.ncbi.nlm.nih.gov/35847547/, last accessed 2025/09/03.
3. Ravi, N., Gabeur, V., Hu, Y.-T., Hu, R., Ryali, C., Ma, T., Khedr, H., Rädle, R., Rolland, C., Gustafson, L., Mintun, E., Pan, J., Alwala, K.V., Carion, N., Wu, C.-Y., Girshick, R., Dollár, P., Feichtenhofer, C.: SAM 2: Segment Anything in Images and Videos, http://arxiv.org/abs/2408.00714, (2024). https://doi.org/10.48550/arXiv.2408.00714.
4. Ronneberger, O., Fischer, P., Brox, T.: U-Net: Convolutional Networks for Biomedical Image Segmentation, http://arxiv.org/abs/1505.04597, (2015). https://doi.org/10.48550/arXiv.1505.04597.
5. nnU-Net: a self-configuring method for deep learning-based biomedical image segmentation | Nature Methods, https://www.nature.com/articles/s41592-020-01008-z, last accessed 2025/09/03.
6. Abid, A., Abdalla, A., Abid, A., Khan, D., Alfozan, A., Zou, J.: Gradio: Hassle-Free Sharing and Testing of ML Models in the Wild, http://arxiv.org/abs/1906.02569, (2019). https://doi.org/10.48550/arXiv.1906.02569.

7. Spaces Overview, https://huggingface.co/docs/hub/en/spaces-overview, last accessed 2025/09/03.
8. Ryali, C., Hu, Y.-T., Bolya, D., Wei, C., Fan, H., Huang, P.-Y., Aggarwal, V., Chowdhury, A., Poursaeed, O., Hoffman, J., Malik, J., Li, Y., Feichtenhofer, C.: Hiera: A Hierarchical Vision Transformer without the Bells-and-Whistles, http://arxiv.org/abs/2306.00989, (2023). https://doi.org/10.48550/arXiv.2306.00989.
9. Virtanen, P., Gommers, R., Oliphant, T.E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S.J., Brett, M., Wilson, J., Millman, K.J., Mayorov, N., Nelson, A.R.J., Jones, E., Kern, R., Larson, E., Carey, C.J., Polat, İ., Feng, Y., Moore, E.W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E.A., Harris, C.R., Archibald, A.M., Ribeiro, A.H., Pedregosa, F., van Mulbregt, P.: SciPy 1.0: fundamental algorithms for scientific computing in Python. Nat Methods. 17, 261–272 (2020). https://doi.org/10.1038/s41592-019-0686-2.
10. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: PyTorch: An Imperative Style, High-Performance Deep Learning Library, http://arxiv.org/abs/1912.01703, (2019). https://doi.org/10.48550/arXiv.1912.01703.
11. How to fine-tune: Focus on effective datasets, https://ai.meta.com/blog/how-to-fine-tune-llms-peft-dataset-curation/, last accessed 2025/09/04.
12. Macdonald, J.A., Zhu, Z., Konkel, B., Mazurowski, M.A., Wiggins, W.F., Bashir, M.R.: Duke Liver Dataset: A Publicly Available Liver MRI Dataset with Liver Segmentation Masks and Series Labels. Radiol Artif Intell. 5, e220275 (2023). https://doi.org/10.1148/ryai.220275.
13. (PDF) TotalSpineSeg: Robust Spine Segmentation with Landmark-Based Labeling in MRI, https://www.researchgate.net/publication/389881289_TotalSpineSeg_Robust_Spine_Segmentation_with_Landmark-Based_Labeling_in_MRI, last accessed 2025/09/04. https://doi.org/10.13140/RG.2.2.31318.56649.
14. Pérez-García, F., Sparks, R., Ourselin, S.: TorchIO: A Python library for efficient loading, preprocessing, augmentation and patch-based sampling of medical images in deep learning. Computer Methods and Programs in Biomedicine. 208, 106236 (2021). https://doi.org/10.1016/j.cmpb.2021.106236.
15. Adding a new `nnUNetTrainer` variant for extensive data augmentations · Issue #2733 · MIC-DKFZ/nnUNet, https://github.com/MIC-DKFZ/nnUNet/issues/2733, last accessed 2025/09/04.
16. 3D Slicer image computing platform, https://slicer.org/, last accessed 2025/09/15.