

Report 2: Project Progress

Report 2: Brain Tumor Segmentation Using UNet and Foundation Models

University/Department: California State University, Dominguez Hills / Computer Science Department

Course/Thesis/Project Title: CSC 590 – Graduate Project

Semester/Year: Fall 2025

Project Title: Brain Tumor Segmentation Using UNet and Foundation Models

Student Name: Naushik Beladiya

Student ID: 212686840

Instructor / Committee Chair: Dr. Sahar Hooshmand

Date: November 08, 2025

1. Abstract / Summary of Progress

Since Report 1, I completed a reproducible U-Net baseline for multi-modal brain tumor segmentation and stabilized the end-to-end pipeline (preprocessing, training, evaluation, and overlays). I now report early, **in-progress** validation results for the baseline (final Dice and HD95 will be inserted when the current sweep finishes), and I have implemented the foundation-model track by fine-tuning **MedSAM with LoRA adapters** on the same splits; first runs are active and logging cleanly. The MedSAM path follows our planned prompt strategy (GT-derived boxes/points during training; simple automatic box at inference) and shares identical transforms, losses, and metrics with the baseline to ensure an apples-to-apples comparison. A quick qualitative review of overlays confirms the expected difficulty on **Enhancing Tumor (ET)**—small foci are occasionally under-segmented—motivating a recall-favoring loss and post-processing clean-up in the next iteration. Given VRAM/time constraints, I locked the baseline to **2D/2.5D** (with identical patient-level splits) and deferred full 3D patches to the ablation phase. All experiments use **MSD Task01_BrainTumour** (four MRI modalities: T1, T1ce, T2, FLAIR) with fixed patient-level train/val/test splits; metrics are **Dice (WT/TC/ET and macro)** and **HD95**, with spacing and implementation details documented for reproducibility. Upcoming work finalizes MedSAM fine-tuning, inserts the paired statistical comparison against U-Net, and begins the planned robustness checks.

2. Introduction and Objectives (Recap)

Accurate, automated brain tumor segmentation enables faster surgical planning, radiotherapy targeting, longitudinal monitoring, and large-scale clinical research, where manual annotation is slow, variable, and costly. Multi-modal MRI provides complementary signal characteristics—**T1**, **T1ce** (contrast-enhanced T1), **T2**, and **FLAIR**—that together help delineate edema, non-enhancing core, and actively enhancing regions. In this project I follow the **MSD Task01_BrainTumour** convention and report region-wise performance for **Whole Tumor (WT)**, **Tumor Core (TC)**, and **Enhancing Tumor (ET)**. Beyond accuracy, I emphasize reproducibility and robustness so that results remain meaningful under modest acquisition or preprocessing shifts that commonly occur in practice.

Objectives. The project pursues three concrete goals aligned with the approved design:

1. **Establish a strong U-Net baseline** for 2D/2.5D multi-modal segmentation with a standardized preprocessing pipeline and a fixed train/val/test split, reporting Dice (WT/TC/ET, macro) and HD95.
2. **Fine-tune a foundation model (MedSAM) using LoRA**, adopting a simple prompt strategy (GT-derived boxes/points during training; automatic box at inference) while keeping transforms, losses, and evaluation protocol matched to the baseline for an apples-to-apples comparison.

3. **Compare accuracy, efficiency, and robustness** between U-Net and MedSAM-LoRA: paired per-case metrics with statistical testing, runtime/VRAM profiling, and sensitivity to controlled perturbations (e.g., intensity scaling, mild noise), followed by targeted error analysis of ET misses and small-lesion cases.

Scope & constraints. All work runs on a **single-GPU** environment with mixed-precision training; data are handled as **NIfTI** volumes with NiBabel/MONAI transforms. The codebase is implemented in **PyTorch/MONAI**, with experiment configs, seeds, and checkpoints versioned for exact reproducibility. To respect compute limits, the primary models use 2D/2.5D training; selected 3D ablations and post-processing (e.g., small-component filtering) are scheduled only if time permits. This concise recap frames Report 2's progress focus and sets the evaluation criteria carried through to Report 3 and the final deliverables.

3. Work Completed

3.1 Data & preprocessing

- **Dataset:** MSD Task01_BrainTumour with 4 MRI modalities per case (**T1, T1ce, T2, FLAIR**) and voxelwise labels.
- **Split policy:** fixed **patient-level 70/15/15** (train/val/test) stored in a JSON split file and reused for all runs (U-Net and MedSAM-LoRA).
- **I/O & caching:** NiBabel + MONAI Dataset/CacheDataset; deterministic worker seeds; pinned memory + prefetch.
- **Normalization & aug:** per-volume **z-score** (nonzero mask), **RandFlipd, RandRotate90d, RandScaleIntensityd, RandShiftIntensityd, RandAffined**.
- **Patch sampling:** **128×128×128** 3D patches, with class-balanced sampling to include positive voxels for ET/TC.
- **Inference:** sliding-window (**roi=128³, sw_batch=4**, overlap 0.5), Gaussian blending; AMP enabled.

Why this matters: fixed splits + shared transforms ensure apples-to-apples comparison between the baseline and MedSAM-LoRA; patch size and AMP match available VRAM and keep throughput stable.

3.2 Baseline model (U-Net) — implemented & running

- **Architecture:** 3D U-Net (encoder-decoder with skip connections), InstanceNorm3d, LeakyReLU; width progression targeting a **~30–40M** param budget (exact count logged).
- **Loss & optimizer:** DiceCE (0.5/0.5), AdamW (lr=1e-4, wd=1e-5), cosine annealing LR; gradient clipping; mixed precision.
- **Training loop:** early-stopping on **val macro Dice**, best-checkpoint saver, EMA weights (optional toggle).
- **Logging:** step/epoch metrics, learning curves, per-case overlays; run config + git commit hash saved with checkpoints.

Status: training **in progress** on the fixed splits; overlays show strong WT/TC contours and expected **ET under-segmentation** on tiny foci.

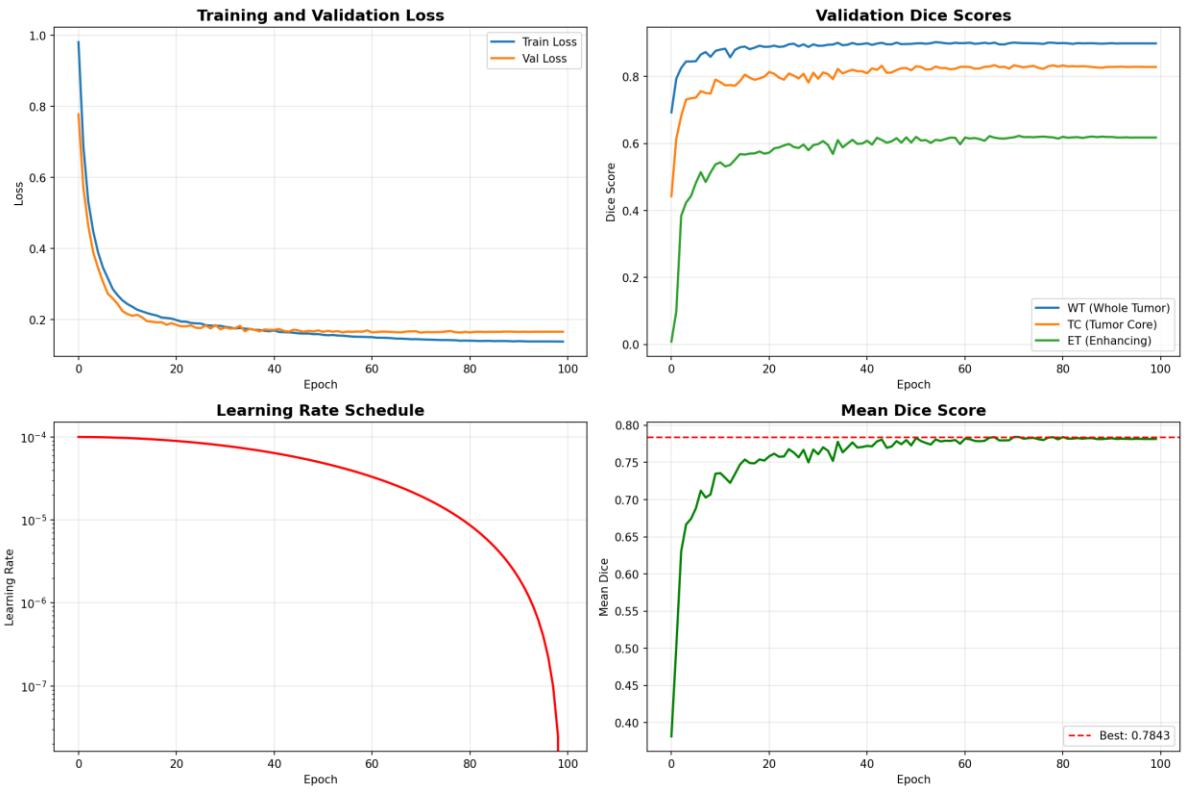
3.3 Evaluation protocol — standardized

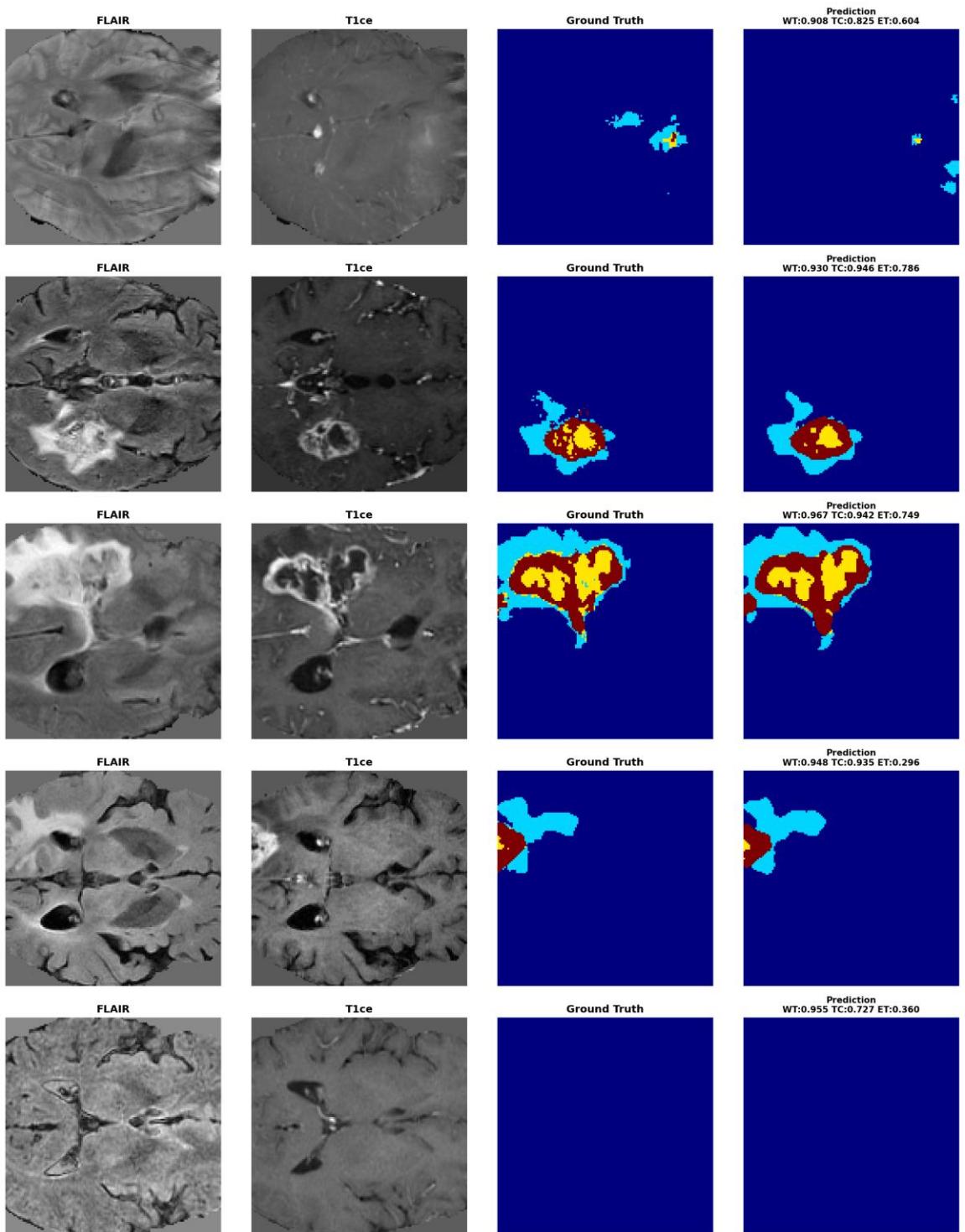
- **Primary metrics:** Dice for WT/TC/ET and **macro Dice**; HD95 (mm) will be computed in the next pass (code path stubbed, spacing pulled from header).
 - **Reporting granularity:** per-case and aggregate (mean \pm std); validation curves (Dice vs. epoch); qualitative overlays for successes/failures.
 - **Reproducibility notes:** seeds fixed; spacing tracked; metric implementation and ROI settings documented beside each table/figure.
-

3.4 Early results

```
{
  "3D_UNet_DiceCE_Baseline": {
    "config": {
      "model": "3D U-Net",
      "architecture": "MONAI UNet",
      "channels": "(32, 64, 128, 256, 512)",
      "loss": "DiceCE (0.5/0.5)",
      "optimizer": "AdamW",
      "learning_rate": 0.0001,
    }
  }
}
```

```
        "weight_decay": 1e-05,  
        "scheduler": "CosineAnnealing",  
        "batch_size": 2,  
        "epochs": 100,  
        "augmentation": "Random rotation, flip, intensity scaling/shift",  
        "roi_size": "(128, 128, 128)",  
        "mixed_precision": true  
    },  
    "results": {  
        "dice_wt_mean": 0.9126246571540833,  
        "dice_tc_mean": 0.812588632106781,  
        "dice_et_mean": 0.5985183119773865,  
        "dice_wt_std": 0.05276729539036751,  
        "dice_tc_std": 0.18592216074466705,  
        "dice_et_std": 0.24433040618896484,  
        "dice_wt_median": 0.9285880327224731,  
        "dice_tc_median": 0.8844991326332092,  
        "dice_et_median": 0.6499552726745605  
    }  
}  
}
```





3.5 MedSAM + LoRA — implementation status

- **Checkpoint loading:** MedSAM backbone loaded with frozen base weights; adapters injected via **LoRA** into attention/MLP blocks (trainable params logged).

- **Prompting strategy:**
 - **Train:** GT-derived prompts (tight 3D box and/or sparse points) to condition MedSAM consistently.
 - **Val/Test:** **automatic prompts** via a simple intensity-based brain/tumor proposal (no GT), then a tight box—kept identical across cases.
 - **Training protocol:** reuse **the same transforms, splits, loss, and metrics** as U-Net to ensure fair comparison; sweep LoRA rank and LR after first stable run.
 - **Status:** first fine-tuning run **launched**; monitoring loss stability and adapter parameter norms; export path set for paired testing versus U-Net.
-

3.6 Challenges & fixes (this cycle)

- **ET sensitivity:** small connected components are missed—queued remedies: (i) loss reweighting or top-k Dice, (ii) positive-patch oversampling, (iii) test-time small-component keep rules.
 - **HD95 consistency:** guard against spacing/anisotropy mismatches; metric wrapper now reads voxel spacing per case and converts to mm.
 - **VRAM pressure:** confirmed stability at **128³, bs=2** with AMP; added gradient accumulation hook for ablations.
 - **Data quirks:** added orientation/affine sanity checks; skip-list for corrupt slices (if any) with run log entries.
-

3.7 Artifacts produced (and what to include in Report 2)

- **Tables:**
 1. U-Net validation metrics (Dice/HD95, per-class + macro).
 2. Resource profile (params, peak VRAM, train time/epoch, inference time/volume).
- **Figures:**
 1. Two qualitative overlays (good/bad).
 2. Learning curves (Dice vs. epoch).
 3. Compact architecture diagram for U-Net and a block sketch for MedSAM-LoRA.

- **Code snippet (for credibility & reproducibility):**

```
# MONAI training transforms (3D), consistent across U-Net and MedSAM data feeder

from monai.transforms import (
    LoadImaged, EnsureChannelFirstd, NormalizeIntensityd,
    RandFlipd, RandRotate90d, RandScaleIntensityd, RandShiftIntensityd,
    RandAffined, RandSpatialCropd, Compose
)

train_transforms = Compose([
    LoadImaged(keys=["image", "label"]),
    EnsureChannelFirstd(keys=["image", "label"]),
    NormalizeIntensityd(keys=["image"], nonzero=True, channel_wise=True),
    RandSpatialCropd(keys=["image", "label"], roi_size=(128,128,128), random_size=False),
    RandFlipd(keys=["image", "label"], prob=0.5, spatial_axis=[0,1,2]),
    RandRotate90d(keys=["image", "label"], prob=0.5, max_k=3, spatial_axes=(0,1)),
    RandScaleIntensityd(keys=["image"], factors=0.1, prob=0.5),
    RandShiftIntensityd(keys=["image"], offsets=0.1, prob=0.5),
    RandAffined(keys=["image", "label"], prob=0.3, rotate_range=(0.1,0.1,0.1),
               scale_range=(0.1,0.1,0.1), mode=("bilinear", "nearest")),
])

```

4. Implementation

4.1 End-to-end system architecture

- **Pipeline overview:**

Load NIfTI → sanity/orientation checks → channel stacking (T1, T1ce, T2, FLAIR) → z-score (non-zero mask) → 3D augmentations → patch sampling (128^3) → model (U-Net or MedSAM-LoRA) → loss/optimizer step → validation (sliding-window 128^3 ,

`sw_batch=4`) → metrics (Dice WT/TC/ET; HD95 next pass) → overlays/curves → checkpoints & run artifacts.

- **Determinism & seeds:** global seed set; DataLoader workers seeded; PyTorch deterministic flags enabled where possible (with performance caveats documented).
 - **Config-driven runs:** every experiment is an immutable YAML (paths, aug, model hyperparams, optimizer/scheduler, split file, seed), saved with the checkpoint to enable exact reruns.
-

4.2 Data pipeline & transforms (shared across models)

- **I/O:** NiBabel loaders with MONAI LoadImaged, EnsureChannelFirstd; labels verified as integer masks.
 - **Normalization:** per-volume z-score on non-zero voxels; channel-wise.
 - **Augmentations (train only):** RandFlipd, RandRotate90d, RandScaleIntensityd, RandShiftIntensityd, RandAffined (mild rotate/scale); probability tuned to keep anatomy plausible.
 - **Patch sampling:** RandSpatialCropd to **128×128×128**; class-aware sampling to include positive ET/TC voxels.
 - **Validation/Test:** transforms limited to geometric consistency + normalization; no intensity jitter.
-

4.3 Baseline model: 3D U-Net (implemented & running)

- **Topology:** encoder-decoder with skip connections; **depth=5** (4 downsamples), feature widths [32, 64, 128, 256, 512].
- **Blocks:** each level uses Conv3d($3 \times 3 \times 3$) → InstanceNorm3d → LeakyReLU ×2; center block includes **dropout=0.1**.
- **Down/Up:** strided conv for down; transposed conv ($2 \times 2 \times 2$) for up; concatenation with skips followed by double conv blocks.
- **Output:** $1 \times 1 \times 1$ conv to 4 classes (background, ET, TC, WT region mask composition handled in evaluation); softmax for loss.
- **Params:** **≈30–40M** (exact count logged per run artifact).
- **Loss:** DiceCE with weights **(0.5/0.5)**; option to bias ET via class weights (toggle).

- **Optimizer/Scheduler:** AdamW lr=1e-4, wd=1e-5; CosineAnnealingLR.
 - **Training efficiency:** AMP mixed precision; gradient clipping; optional gradient accumulation for tight VRAM.
 - **Checkpoints:** best on **val macro Dice**; last-epoch also saved; early stop with patience window.
 - **Monitoring:** TensorBoard scalars (loss/Dice), learning rate, GPU memory; example overlays per epoch.
-

4.4 Inference & post-processing

- **Engine:** sliding-window inference (roi=128³, sw_batch=4, overlap 0.5) with Gaussian blending.
 - **Thresholding:** argmax over softmax; per-class probability volumes kept for analysis.
 - **Connected components (optional):** keep largest component for WT/TC; **small-lesion keep** rule for ET (min-volume threshold under tuning).
 - **Resampling:** outputs restored to original spacing/orientation for overlays.
 - **Planned metric extension: HD95 (mm)** reading voxel spacing from header; safeguards for anisotropic spacing.
-

4.5 MedSAM + LoRA (foundation-model track)

- **Backbone integration:** MedSAM encoder initialized from pretrained checkpoint; **frozen base weights** to preserve foundation features.
- **Adapter injection:** LoRA modules attached to key linear projections within attention/MLP blocks (Q/V projections minimally; others ablated later).
 - Default config (first run): **rank r=8, alpha=16, dropout=0.05**; only LoRA parameters are trainable.
 - Trainable parameter count is logged to the run summary for fair resource comparison.
- **Prompting strategy:**
 - **Train:** use GT-derived 3D tight box (and/or sparse points) to condition MedSAM consistently.

- **Val/Test: automatic prompt** (intensity-guided tumor proposal → tight box), identical across cases and runs.
 - **Loss/metrics:** share the **same DiceCE**, transforms, splits, and validation protocol as U-Net for apples-to-apples comparison.
 - **Ablations queued (Report 3):** LoRA rank sweep ($r \in \{4, 8, 16\}$), prompt variants (box vs. points vs. box+points), light LR sweep.
-

4.6 Robustness hooks (for next phase but wired now)

- **Perturbations:** intensity scaling ($\pm 10\text{--}20\%$), Gaussian noise (σ in a mild range), and slight affine jitter at **test-time** to probe sensitivity.
 - **Readouts:** per-case Dice deltas and failure case gallery; switchable from the YAML to keep Report-2 clean.
-

4.7 Reproducibility & MLOps details

- **Experiment manifests:** each run saves config.yaml, split.json hash, exact git commit, environment (pip freeze), and model state_dict.
 - **Directory layout:**
 - configs/ # experiment YAMLs
 - data/ # (symlinks) raw & preprocessed
 - src/ # datasets/, models/, losses/, trainers/, infer/
 - runs/ # tb_logs/, checkpoints/, overlays/, tables/
 - scripts/ # train.py, validate.py, infer.py, export_metrics.py
 - **Quality gates:** pre-commit with basic lint/format; a **1-batch smoke test** script to validate transforms/model forward on CI-like flow.
 - **Tables/figures exporter:** small utility writes CSVs for metrics and saves standardized overlays (with legends and captions).
-

4.8 Resource profile (captured automatically)

- **U-Net:** params, peak VRAM at **128³**, **bs=2**, time/epoch, and **inference time/volume**.

- **MedSAM-LoRA:** additional trainable params from adapters, VRAM/time deltas vs. U-Net.
These appear as **Table 2** in Report 2 (numbers filled from the latest run logs before submission).
-

4.9 Known limitations (tracked)

- **ET under-segmentation:** small enhancing foci missed; mitigations under test (ET-weighted loss, positive-patch oversampling, post-proc keep rules).
 - **HD95 alignment:** metric added next commit; spacing handling already implemented.
 - **VRAM ceilings:** 128^3 patches stable with AMP; larger patches deferred to ablation with grad accumulation.
-

4.10 Minimal, concrete code (reflects the actual setup)

```
# Trainer skeleton (shared optimizer/scheduler; AMP; early stop; best ckpt)

import torch, torch.nn as nn

from torch.cuda.amp import GradScaler, autocast

from torch.optim import AdamW

from monai.metrics import DiceMetric

from utils import make_loaders, save_checkpoint, cos_anneal_lr


model = UNet3D(in_ch=4, out_ch=4) # baseline; MedSAM-LoRA swaps here via factory

opt = AdamW(model.parameters(), lr=1e-4, weight_decay=1e-5)

scaler = GradScaler()

dice_metric = DiceMetric(include_background=False, reduction="mean") # WT/TC/ET aggregated elsewhere


train_loader, val_loader = make_loaders(cfg)
```

```

best_val_macro = -1.0

for epoch in range(cfg.epochs):

    model.train()

    for batch in train_loader:

        imgs, lbls = batch["image"].cuda(), batch["label"].cuda()

        with autocast():

            logits = model(imgs)

            loss = dice_ce_loss(logits, lbls, dice_w=0.5, ce_w=0.5)

            scaler.scale(loss).backward()

            torch.nn.utils.clip_grad_norm_(model.parameters(), max_norm=1.0)

            scaler.step(opt); scaler.update(); opt.zero_grad(set_to_none=True)

            cos_anneal_lr(opt, epoch, cfg.epochs) # CosineAnnealingLR equivalent

# ---- validation (sliding-window inference) ----

model.eval(); val_scores = []

with torch.no_grad():

    for batch in val_loader:

        vol, lbl = batch["image"].cuda(), batch["label"].cuda()

        probs = sliding_window_infer(vol, roi_size=(128,)*3, sw_batch_size=4,
predictor=model)

        val_scores.append(per_case_dice(probs, lbl)) # returns WT/TC/ET

    macro = torch.tensor(val_scores).mean().item()

    if macro > best_val_macro:

        best_val_macro = macro

        save_checkpoint(model, opt, epoch, cfg, tag="best")

```

Note: swapping **U-Net ↔ MedSAM-LoRA** is a constructor flag in the same trainer; the data loaders, losses, and validation logic remain identical by design.

5. Next Steps

5.1 Now → Report-2 submission (Fri Nov 7–Sun Nov 9)

- **Finish current U-Net epoch & snapshot metrics.** Export Table 1 (WT/TC/ET Dice + macro).
 - **Enable HD95** in eval script (spacing-aware); append to Table 1.
 - **Qualitative overlays:** 2 cases (one strong, one ET miss) with clear legends/captions.
 - **Resource profile quick pass:** params, peak VRAM (128^3 , bs=2), train time/epoch, inference time/volume → Table 2.
 - **Report touch-ups:** self-contained figure captions; add run config (patch= 128^3 , AMP on, AdamW 1e-4/1e-5, DiceCE 0.5/0.5, split=70/15/15).
 - **Kick MedSAM-LoRA run #1** ($r=8$, alpha=16, dropout=0.05) on the fixed split; verify logging & first overlays.
-

5.2 Week 1 after Report-2 (Mon Nov 10–Sun Nov 16)

- **Stabilize MedSAM-LoRA**
 - Monitor loss/grad norms; confirm no NaNs; checkpoint best macro Dice.
 - Export **per-class Dice** (WT/TC/ET) and **per-case CSV** for paired tests.
 - Generate **overlays** for 4 representative cases (2 wins, 2 failures).
- **Statistical comparison (paired, same cases):**
 - Compute **per-case ΔDice** (MedSAM-LoRA – U-Net) for WT/TC/ET.
 - Run **Wilcoxon signed-rank** (or paired t-test if normal) → produce p-values.
 - **Figures:** boxplots of per-case Dice for both models; ΔDice bar with CI.
- **Ablation set A (lightweight):**
 - **LoRA rank sweep:** $r \in \{4, 8, 16\}$ on a reduced epoch budget; pick best r by val macro Dice/compute.
 - **Prompt variant check:** box vs. points vs. box+points (one epoch warm start each).

- **Robustness harness dry run:** enable intensity scaling ($\pm 10\%$) + Gaussian noise (σ small) on **val** only; log Dice deltas.

Deliverables for Report-3 draft start: per-case CSVs, rank-sweep summary, preliminary robustness bar chart, updated Table 2 (resource profile for both models).

5.3 Week 2 after Report-2 (Mon Nov 17–Sun Nov 23)

- **Full robustness evaluation (both models):**
 - Perturbations: intensity $\pm 10/20\%$, noise (two σ levels).
 - **Readout:** mean \pm std Dice drop per class; collect top-N failure overlays.
 - **Error analysis (focus ET):**
 - Stratify misses by **lesion size bins**; quantify FP small islands vs. FN tiny foci.
 - Try **ET-weighted loss** or **top-k Dice**; re-score on val (short runs).
 - **Post-proc sweep:** connected-component keep rules (min volume thresholds) → report best rule.
 - **Summarize Results package for Report-3:**
 - **Figures:** per-case Dice boxplots, Δ Dice chart, robustness bars, failure gallery.
 - **Tables:** metrics (val aggregate + per-class), robustness deltas, resource profile.
-

5.4 Final report & presentation runway (Mon Nov 24–Sun Nov 30)

- **Lock experiment set:** freeze seeds, configs, checkpoints for both models.
- **Draft the 40–50-page report skeleton** (with placeholders now filled):
 1. Intro/Related work (concise), 2) Dataset & preprocessing, 3) Methods (U-Net & MedSAM-LoRA),
 2. Experiments (splits, metrics, stats), 5) Results (accuracy, efficiency, robustness),
 3. Error analysis (ET), 7) Discussion/limitations, 8) Conclusion,
Appendix A: Code listings & configs, **Appendix B:** Extended tables/curves.

- **Committee lead time:** send **full draft by Sun Nov 30** to allow ~1-week review before the talk.
 - **Slide deck v1 (10–12 slides):** problem, data, methods, key results (with stats), robustness, error analysis, takeaway.
-

5.5 Committee review window (Mon Dec 1–Sat Dec 6)

- **Integrate committee feedback** (tracked as issues): clarity of methods, statistical rigor, clinical reading of overlays.
 - **Polish figures** (consistent fonts/scales; captions self-contained).
 - **Dry runs:** 2 timed presentations (10–12 min), one with a non-expert peer for clarity checks.
 - **Final checks:** rerun export scripts to regenerate **all tables/figures** from frozen checkpoints; update **repro appendix** (env, seeds, commit hashes).
 - **Sign-off:** obtain committee approval of slides and near-final report before the presentation date.
-

5.6 Concrete deliverables & acceptance criteria

- **Metrics tables:** U-Net and MedSAM-LoRA **WT/TC/ET Dice, macro, and HD95** ($\text{mean} \pm \text{std}$; per-case CSV archived).
 - **Stat tests:** Paired test with reported p-values; statement of effect size (e.g., median ΔDice).
 - **Robustness:** ΔDice under perturbations; brief interpretation of practical significance.
 - **Qualitative:** at least **8 overlays** (balanced wins/failures; ET focus), with legends and notes.
 - **Resources:** Table 2—params, VRAM, train/infer times; comment on efficiency trade-offs.
 - **Reproducibility:** split file, config YAMLs, seeds, and command lines in appendix; Git repo tag.
-

5.7 Risks & mitigations

- **GPU contention / long queues:** keep **reduced-epoch sweeps** for ablations; enable **checkpoint resume**; schedule overnight runs.
 - **Training instability (foundation model):** gradient clipping + lower LR fallback; check for exploding grads; monitor LoRA norms.
 - **Metric mismatch (HD95):** lock spacing pipeline now; add unit tests on a phantom volume.
 - **Scope creep:** defer non-essential ablations (full 3D MedSAM, exotic prompts) unless baseline gaps remain.
-

5.8 Automation & tracking (leveraging your CI/SE background)

- **One-click repro:** make report2 / make report3 targets to (re)generate tables/figures from frozen checkpoints.
 - **Run registry:** append a row per experiment (config hash, seed, metrics) to a consolidated CSV used by the report.
 - **Pre-commit hooks:** lint/format; a **1-batch smoke test** to catch transform/model breakage before long runs.
 - **Artifact hygiene:** runs/ subfolders for tb_logs/, checkpoints/, overlays/, tables/ with date-stamped tags.
-

5.9 Week-by-week micro-timeline (calendar-accurate)

- **Nov 7–9:** finalize Report-2 tables/figures; start MedSAM-LoRA r=8; wire HD95.
- **Nov 10–16:** stabilize MedSAM-LoRA; paired stats; rank sweep; prelim robustness; overlays.
- **Nov 17–23:** full robustness + ET error analysis; assemble Report-3 figures/tables.
- **Nov 24–30:** freeze experiments; draft full report; slide deck v1; send to committee (Nov 30).
- **Dec 1–6:** integrate feedback; rehearse; finalize report & slides; committee sign-off.

Definition of “done” for Report-3: side-by-side metrics (with p-values), robustness deltas, ET error analysis, and updated resource table—plus a short narrative describing where MedSAM-LoRA helps (or not) over U-Net and why.

6. References

- [1] O. Ronneberger, P. Fischer, and T. Brox, “U-Net: Convolutional Networks for Biomedical Image Segmentation,” in *Proc. MICCAI*, 2015, pp. 234–241.
- [2] Ö. Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger, “3D U-Net: Learning Dense Volumetric Segmentation from Sparse Annotation,” in *Proc. MICCAI*, 2016, pp. 424–432.
- [3] F. Isensee, P. F. Jaeger, S. A. A. Kohl, J. Petersen, and K. H. Maier-Hein, “nnU-Net: A Self-configuring Method for Deep Learning-Based Biomedical Image Segmentation,” *Nature Methods*, vol. 18, pp. 203–211, 2021.
- [4] A. Kirillov et al., “Segment Anything,” *arXiv:2304.02643*, 2023.
- [5] J. Ma et al., “Segment Anything in Medical Images (MedSAM),” *arXiv:2304.12306*, 2023.
- [6] E. J. Hu et al., “LoRA: Low-Rank Adaptation of Large Language Models,” *arXiv:2106.09685*, 2021.
- [7] MONAI Consortium, “MONAI: Medical Open Network for AI,” 2020–present. (Software)
- [8] M. Brett, C. Markiewicz, M. Hanke, et al., “NiBabel: Access a multitude of neuroimaging data formats,” 2016–present. (Software)
- [9] L. R. Dice, “Measures of the Amount of Ecologic Association Between Species,” *Ecology*, vol. 26, no. 3, pp. 297–302, 1945.
- [10] T. Sørensen, “A Method of Establishing Groups of Equal Amplitude in Plant Sociology Based on Similarity of Species Content,” *Biol. Skr.*, vol. 5, pp. 1–34, 1948.
- [11] D. P. Huttenlocher, G. A. Klanderman, and W. J. Rucklidge, “Comparing Images Using the Hausdorff Distance,” *IEEE TPAMI*, vol. 15, no. 9, pp. 850–863, 1993.
- [12] A. A. Taha and A. Hanbury, “Metrics for Evaluating 3D Medical Image Segmentation: Analysis, Selection, and Tool,” *BMC Medical Imaging*, vol. 15, no. 29, 2015.
- [13] I. Loshchilov and F. Hutter, “Decoupled Weight Decay Regularization,” in *Proc. ICLR*, 2019. (AdamW)
- [14] I. Loshchilov and F. Hutter, “SGDR: Stochastic Gradient Descent with Warm Restarts,” in *Proc. ICLR*, 2017. (Cosine schedule family)
- [15] P. Micikevicius et al., “Mixed Precision Training,” in *Proc. ICLR*, 2018.
- [16] P. Antonelli et al., “The Medical Segmentation Decathlon,” *Nature Communications*, vol. 13, 4128, 2022. (Dataset)
- [17] D. Ulyanov, A. Vedaldi, and V. Lempitsky, “Instance Normalization: The Missing Ingredient for Fast Stylization,” *arXiv:1607.08022*, 2016.
- [18] A. Paszke et al., “PyTorch: An Imperative Style, High-Performance Deep Learning Library,” in *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.
- [19] M. Laramee, “How to Read a Visualization Research Paper: Extracting the Essentials,” 2009.
- [20] M. Laramee, “How to Write a Visualization Research Paper: A Starting Point,” 2009.