MELBA Journal Machine Learning for Biomedical Imaging

From Prompts to Pipelines: Evaluating LLM-Generated Medical Image Segmentation Baselines

Jasmin **Arjomandi** [1], Luisa Neubig [1], Franziska Mathis-Ullrich [1], Andreas M. Kist [1]

1 Department Artificial Intelligence in Biomedical Engineering, Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU), Germany

Abstract

Large Language Models (LLMs) are increasingly applied to automate complex tasks, but their potential for generating automated medical image segmentation pipelines remains largely underexplored. We present a systematic evaluation of open- and closed-source LLMs in generating U-Net–based segmentation frameworks across six diverse 2D medical image datasets, spanning endoscopy, fluoroscopy, dermoscopic photography, MRI, and fundus imaging. Building on an earlier study, we analyze state-of-the-art 2025 reasoning-enabled models and compare them to non-reasoning LLMs and a strong nnU-Net v2 baseline. Compared to their 2024 predecessors, the 2025 models demonstrated marked improvements in robustness, code quality, and segmentation accuracy across modalities. Our results show that reasoning-augmented LLMs achieve faster convergence, fewer execution errors, and higher Dice scores, while complex datasets with fine structures (e.g., retinal vessels) and volumetric data remain challenging. We also confirmed robustness under repeated runs by comparing one reasoning and one non-reasoning model from the same family, where despite GPT-4o's consistent, template-like code outputs under multiple runs as the non-reasoning model, GPT-o4-mini-high showed significantly lower run-to-run variability in validation loss and tighter Dice score distributions, demonstrating that chain-of-thought reasoning markedly improves both accuracy and stability. These findings highlight the potential of reasoning-enabled LLMs to automate segmentation workflows with high accuracy and explainability, paving the way for their integration into medical imaging pipelines. Our code is available at `https://github.com/ankilab/LLM_based_Segmentation.git`.

Keywords

Large language Models (LLMs), Medical Image Segmentation, Reasoning, Vibe Coding, Prompt Engineering, Chain-of-Thought

Article informations

`https://doi.org//10.59275/j.melba.2026-9369`

Volume 2026, Received: 2025-07, Published 2026-03

Corresponding author: andreas.kist@fau.de

1. Introduction

Large *Language Models (LLMs)* have demonstrated remarkable capabilities in natural language processing, code generation, and reasoning tasks in recent years, enabling their integration into increasingly diverse scientific and technical domains (Huang and Chang, 2023; Plaat et al., 2024). The medical imaging field, which traditionally relied on convolutional neural networks (CNNs) such as U-Net (Ronneberger et al., 2015; Isensee et al., 2021) and its variants (Zhou et al., 2018), is now beginning to explore LLMs as potential tools for enhancing workflows in segmentation, diagnosis, and report generation (Wang and Ke, 2024; Bai et al., 2024; Neubig et al., 2025). However, the integration of LLMs into medical image analysis pipelines raises fundamental questions about their effectiveness, limitations, and potential to contribute to or even automate complex tasks.

In recent LLM developments, chain-of-thought (CoT) reasoning has emerged as a powerful technique to elicit intermediate, step-by-step logic from otherwise opaque models (Wei et al., 2022; Huang and Chang, 2023). By prompting LLMs to "think aloud," CoT enables them to decompose complex tasks into a sequence of reasoning steps, improving accuracy on challenging problems such as arithmetic, commonsense, and multi-hop inference (Wei et al., 2022; Huang

and Chang, 2023; Plaat et al., 2024). Surveys have shown that CoT not only enhances performance but also offers greater transparency and interpretability in model decision-making (Wang et al., 2025; Plaat et al., 2024). Building on these advances, reasoning-augmented LLMs—such as DeepSeek-R1, which uses reinforcement learning to incentivize coherent thought traces—demonstrate near state-of-the-art reasoning capabilities while retaining inference efficiency (Guo et al., 2025; Marjanović et al., 2025). In the medical contexts, CoT reasoning can be used for transparency and interpretability of medical diagnostics (Wang et al., 2025). Thinking further, this strategy can be applied to prompt LLMs not only to select suitable neural network architectures and adjust preprocessing steps for different imaging modalities, but also to articulate the reasoning behind these choices and to autonomously debug segmentation pipelines, which is especially important in clinical and medical image-based applications.

Segmentation remains a core challenge in medical imaging, especially for applications such as tumor delineation, organ boundary detection, and lesion quantification.

State-of-the-art segmentation architectures, such as nnU-Net (Isensee et al., 2021), have achieved considerable success when trained on custom, annotated datasets, offering strong baseline performance across a wide range of tasks. In contrast, foundation models like MedSAM (Ma et al., 2024) can perform zero-shot or few-shot segmentation without extensive task-specific fine-tuning. However, deploying either approach in practice often requires substantial coding expertise and manual adaptation to different datasets and imaging modalities. This limitation motivates research into whether LLMs can automatically generate robust and potentially problem-tailored segmentation baselines, and provide reasoning about architectural design choices (Wang and Ke, 2024; Kumar et al., 2025).

LLMs have recently shown promise in medical domains by enabling interactive diagnostic support (Wang et al., 2024), domain-specific adaptation (Liu et al., 2023), and reasoning-based multimodal segmentation systems such as LISA (Lai et al., 2024) and MedSeg-R (Huang et al., 2025). Additionally, large-scale efforts, such as HuatuoGPT-Vision (Chen et al., 2024a) and M3D (Bai et al., 2024), have highlighted the potential of aligning visual and textual representations in complex medical datasets. These studies indicate a growing interest in leveraging the reasoning and generative abilities of LLMs for medical image analysis.

Despite these advances, few works systematically analyze how different LLMs perform in automating segmentation pipelines. Most prior research has focused either on directly applying LLMs to perform image analysis tasks (Sun et al., 2024) or on fine-tuning them for domain-specific diagnostic reasoning (Liu et al., 2023). The question remains whether LLMs can serve as a reliable backbone for generating and executing segmentation code with minimal human intervention in the spirit of "vibe coding".

This work aims to fill this gap, and investigate whether large language models can autonomously generate robust, executable, and generalizable U-Net–based segmentation pipelines for diverse medical imaging tasks with minimal human intervention, by providing a comprehensive evaluation of several open- and closed-source LLMs in generating U-Net–based medical image segmentation baselines. Building upon our earlier study on LLM-driven baselines (Arjomandi et al., 2025), we extend the scope by additionally analyzing state-of-the-art models released in 2025, comparing those to 2024-released LLMs, as well as comparing the output of reasoning models using chain of thought with the non-reasoning, and examining their performance across diverse datasets, modalities and segmentation tasks from different medical domains, including endoscopic laryngoscopy, videofluoroscopic swallowing studies, dermoscopic skin lesion imaging, pelvic MRI for uterine myomas, brain tumor MRI, and retinal fundus photography; capturing a broad range of imaging modalities and anatomical targets. In doing so, we aim to shed light on the practical utility of LLMs for medical image segmentation, their potential for explainability through reasoning, and identify future research directions for their integration into clinical workflows.

2. Related Works

Deep learning has driven significant progress in medical image segmentation, with architectures such as U-Net (Ronneberger et al., 2015) and its derivatives achieving widespread adoption across tasks like tumor delineation and organ segmentation (Isensee et al., 2021; Zhou et al., 2018). Extensions like nnU-Net introduced self-configuring pipelines capable of adapting preprocessing, network architecture, and training settings to diverse datasets, enabling state-of-the-art results without extensive manual tuning (Isensee et al., 2021). Despite these advances, implementing and adapting these frameworks requires considerable expertise in programming and machine learning, posing a barrier for many medical researchers or direct clinical use.

The emergence of Large Language Models (LLMs) has sparked growing interest in their potential for automating medical imaging workflows. Early explorations have applied LLMs to assist in medical report generation (Wang and Ke, 2024) and provide interactive diagnostic support systems such as ChatCAD, which integrates GPT-based reasoning with computer-aided diagnosis tools (Wang et al., 2024). More recently, LLMs have been combined with vision encoders to perform segmentation tasks, as seen in LISA, which leverages LLM reasoning to refine mask generation based on user-provided prompts, demonstrating the potential for interactive, reasoning-aware segmentation

systems (Lai et al., 2024), and MedSeg-R, which integrates a vision–language model with reasoning capabilities to interpret complex clinical instructions and produce pixel-level segmentation masks (Huang et al., 2025).

Efforts to adapt general-purpose LLMs for medical imaging include HuatuoGPT-Vision, which injects large-scale medical visual knowledge into multimodal LLMs, achieving strong performance on chest X-ray diagnosis and segmentation benchmarks (Chen et al., 2024a). M3D extends this approach to 3D medical imaging by fusing volumetric features with textual clinical data for multi-task learning, including segmentation and report generation (Bai et al., 2024). TriMedLM further advances this direction by aligning 3D voxel representations with language-based reasoning, showing improved explainability and accuracy in volumetric datasets (Chen et al., 2024b).

Other works have explored lightweight LLM integration for weakly supervised segmentation. Cai et al. (Cai et al., 2024) introduced a method combining multi-label contrastive learning with LLM-derived semantic guidance, narrowing the gap between weak and fully supervised performance. In parallel, semi-supervised frameworks such as LLM-SegNet employ LLMs to provide task-informed priors for training with limited annotations, improving accuracy in 3D medical imaging tasks (Wang and Ke, 2024). However, the use of LLMs to autonomously generate executable segmentation code for architectures like U-Net remains underexplored, particularly in ensuring generalization across diverse medical datasets. Our study addresses this gap by demonstrating how LLMs can be harnessed for medical image segmentation with minimal user intervention.

Our earlier work evaluated eight open and closed source LLMs released in 2024 for automatic generation of U-Net-based segmentation pipelines based on natural-language prompts, analyzing their code quality, error rates with self-correction, and segmentation performance on three datasets (Arjomandi et al., 2025). While models such as GPT o1 Preview and Claude 3.5 Sonnet produced error-free code and competitive results, some struggled to generate functional scripts without manual intervention.

The present study builds on this foundation by systematically comparing eleven reasoning and non-reasoning LLMs, including recently released 2025 models, and evaluating their performance and error rates across six datasets from diverse imaging modalities. This expanded analysis positions our work at the intersection of LLM-based code synthesis and reasoning-aware segmentation, offering deeper insights into how LLMs can be utilized for medical image segmentation with minimal user involvement.

3. Methods

3.1 Selected LLMs

In addition to the eight 2024 selected LLMs (Arjomandi et al., 2025) (including GPT-4 (Achiam et al., 2023), 4o (Hurst et al., 2024) and o1 Preview (OpenAI, b), Claude 3.5 Sonnet (Anthropic, a), Gemini 1.5 Pro (Google, a), GitHub Copilot (GitHub), Bing Microsoft Copilot (Microsoft) (closed source), and Llama 3.1 405B (open source) (Meta AI, a)) with GPT o1-preview being the only reasoning-model, we evaluated eleven state-of-the-art 2025 released models (including the closed source models: Claude 4 Sonnet (Anthropic, b), DeepSeek R1 (Guo et al., 2025) and V3 (Liu et al., 2024), GPT o3 and O4-mini-high (OpenAI, a), Gemini 2.5 Pro (Google, b), Grok 3 mini and Grok 3 (xAI), and the open source models: Llama 4 Maverick (Meta AI, b), Mistral Medium 3 (Mistral AI) and Qwen 3_235B (AbaLab)), with DeepSeek V3 and Grok 3 (without thinking mode enabled) being the non-reasoning models (Figure 1).

3.2 Datasets

In addition to the Benchmark for Automatic Glottis Segmentation (BAGLS) (Gómez et al., 2020), with endoscopic video frames from laryngoscopy exams annotated for glottis segmentation, the Bolus Swallowing dataset (Neubig et al., 2022) with videofluoroscopic swallowing studies annotated for bolus region segmentation, and the Brain Meningioma MRI dataset (Akter et al., 2024) with single-slice grayscale MR images with tumor segmentation masks used previously (Arjomandi et al., 2025), we incorporated three additional datasets to test the robustness and generalizability of LLM-generated U-Net-based models across imaging modalities. These include the HAM10000 ISIC Skin Cancer dataset (International Skin Imaging Collaboration; Tschandl et al., 2018), which consists of dermoscopic RGB images annotated for skin lesion segmentation, and poses challenges due to high intra-class variability in lesion appearance and indistinct boundaries blending with healthy skin ; the UMD Uterine Myoma MRI dataset (Pan et al., 2024), comprising pelvic MRI scans labeled "3" for uterine myoma; and a combined retinal vessel segmentation dataset (Pradosh123) formed by harmonizing DRIVE (DRIVE Grand Challenge), HRF (Odstrcilik et al., 2013), CHASE_DB1 (Owen et al., 2009), and STARE (Hoover et al., 2000) fundus photography datasets, annotated for retinal blood vessels, a task requiring detection of extremely fine vascular structures with high sensitivity to image quality and noise (Figure 1).

Together, these datasets span a wide range of modalities (endoscopy, fluoroscopy, dermoscopy, MRI, and fundus imaging) and segmentation challenges, from small and fine structures (retinal vessels) to large, irregular regions (skin

lesions, tumors) and motion artifacts (swallowing studies). This diversity provides a rigorous benchmark for assessing the adaptability of LLM-generated pipelines to modality-specific preprocessing, architectural design choices, and segmentation tasks.

All input data were preprocessed, with resizing and intensity normalization applied by the LLMs in the dataset scripts, to match the input specifications of the LLM-generated models. To ensure balanced representation across modalities and test the generalizability of LLM-generated pipelines, a random subset of 5002 images was selected from the BAGLS, Bolus Swallowing and Skin Cancer datasets, while 999 grayscale slices were included from the Brain Meningioma dataset and 73 retinal fundus images. For the Uterine Myoma MRI dataset, only axial slices containing myoma masks were extracted along with their corresponding binary annotations, yielding 5457 single-slice MRI images. The inclusion of datasets with both large (BAGLS, Myoma) and limited sample sizes (retinal vessels) allowed us to assess model robustness under varying data availability conditions. For the skin lesion (HAM10000) and retinal vessel segmentation datasets, RGB images were preserved and correctly handled in the LLM pipelines. As instructed per prompt, all images were resized and normalized to a $[0, 1]$ range as defined in each model's code (despite minor differences in function calls, the normalization procedure was identical across all generated scripts). Dataset splitting strategies varied: all 2025 models consistently applied `train_test_split` to divide data into training (80%), validation (10%), and test (10%) subsets, whereas some 2024 models used `random_split`, potentially influencing reproducibility. For a standard and fair evaluation of the model performances, one separate randomly selected test subset (10%) was held out from each dataset prior to any training, on which all of the models were ultimately re-evaluated for comparison.

3.3 System Configurations

All experiments were conducted on a machine equipped with an NVIDIA GeForce RTX 3090 GPU, 24GB GDDR6X memory, running Python 3.10, PyTorch 2.2.0, and CUDA 11.8. The LLMs were unaware of the system configurations or availability of a GPU.

3.4 Prompt Engineering

To guide LLMs in generating segmentation pipelines, we designed a carefully engineered prompt (Appendix D) that specifies detailed, bullet-pointed instructions for generating all required scripts (dataset.py, model.py, train.py, and main.py). They specified the use of U-Net-based architectures (model.py), data loading and preprocessing routines (dataset.py), and training and validation loops (train.py). The loss function, hyperparameters and architecture were set by the LLM. To avoid overwhelming the LLMs, confusion, or inefficient token utilization, prompts were kept concise yet explicit (Giray, 2023) and were tested multiple times on each model before finalizing the instructions to ensure consistent and reproducible results. The prompt is available on our GitHub repository.

3.5 Code Generation and Error Tracking

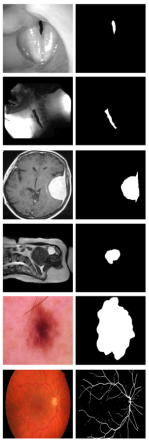
LLMs were tasked with generating ready-to-run code for each dataset. Errors encountered during execution were fed back verbatim to the LLM for self-correction, without providing additional contextual information or any introductory statements, unless stuck in a repeated error loop (occurring only for Gemini 1.5 Pro and Llama 3.1 405B from the 2024 models). No data augmentation, refinements, or hyperparameter tuning was done by us if not proposed, i.e., coded, by the LLM. Number and types of errors, iterations needed to obtain executable code, and any additional interactions for each LLM were tracked and recorded (see GitHub). The generated architectures were primarily U-Net variants with varying depths and feature maps, as defined by each LLM. No manual fine-tuning was applied to maintain comparability. Loss function and optimizer, as well as batch sizes, learning rates, and epochs, were determined directly from each LLM's generated code. The interactions happened directly via each LLM's chat interface or via Poe interface (Poe) (specified in all chat logs, see GitHub). Any referencing and memory features were turned off in the model settings for all experiments.

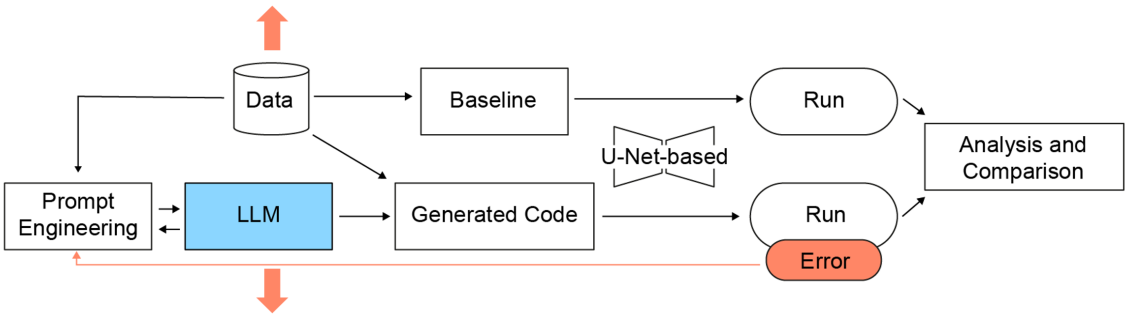
3.6 Evaluation Metrics and Visualization Pipeline

We evaluated the models on a standardized test set using the Dice coefficient as the primary, commonly used metric to measure overlap between predicted segmentation masks and ground truth masks. Training and validation losses were also saved and visualized to assess convergence behavior, using a custom pipeline importing and plotting all saved loss trajectories and Dice scores, and performing inference visualization for qualitative assessment. For baseline comparison, we trained an nnU-Net v2 (Isensee et al., 2021) on the same datasets and hardware setup, using its default 2D configuration, under a fixed computational budget per dataset, and evaluated the checkpoint with the best validation performance.

3.7 Comparison of 2024 vs. 2025 Models

For each dataset, we compared the average training and validation losses of all generated U-Net-based models by LLMs from 2024 against those from 2025. We computed per-epoch averages and standard deviation across each

Image GT	Dataset	Format	Segmentation Mask	Modality
	BAGLS	Endoscopic video frames	Glottis	Endoscopy (laryngoscopy)
	Bolus Swallowing Dataset	Videofluoroscopic images	Bolus region	Fluoroscopy
	Brain Tumor	Single-slice grayscale images	Meningioma tumor masks	MRI
	UMD Uterine Myoma	Pelvic MRI single slices	Uterine myoma masks	MRI
	HAM10000 ISIC Skin Cancer	Dermoscopic RGB images	Skin lesion masks	Dermoscopy
	Retinal Vessel Segmentation	Fundus photography images	Retinal vessel masks	Fundus imaging



Company	2024 Models	2025 Models	#Parameters
OpenAI	GPT-4, GPT-4o (non-Reasoning) GPT o1-Preview (Reasoning)	GPT o3, GPT o4-mini-high (Reasoning)	GPT 4 ~1.8T GPT 4o (& o1) ~200B –
Anthropic	Claude 3.5 Sonnet	Claude 4 Sonnet	~175B –
Google DeepMind	Gemini 1.5 Pro	Gemini 2.5 Pro	~ >200B
GitHub	Copilot	–	–
Microsoft	Bing Microsoft Copilot	–	–
Meta (LLaMA)	Llama 3.1	Llama 4 Maverick	405B –
DeepSeek	–	DeepSeek V3 (non-reasoning) DeepSeek R1 (reasoning)	671B (37B per Token)
xAI	–	Grok 3 mini (reasoning) Grok 3 (non-reasoning)	2.7T
Mistral AI	–	Mistral Medium 3	–
Qwen (Alibaba)	–	Qwen 3_235B	235B

Figure 1: Overview of the study design, datasets, and LLMs used. Top: Six medical imaging datasets spanning diverse modalities (endoscopy, fluoroscopy, MRI, dermoscopy, and fundus imaging) and segmentation targets used to evaluate LLM-generated U-Net-based segmentation pipelines. Middle: Workflow illustrating prompt engineering, code generation by large language models (LLMs), and subsequent training, execution, and analysis. Bottom: Overview of the evaluated LLMs, grouped by company and year of release (2024 vs. 2025), highlighting reasoning and non-reasoning models, along with officially disclosed or estimated number of model parameters.

year's models for the first 20 epochs run on each dataset, and plotted the two mean curves (± standard deviation) to highlight shifts in convergence speed and final loss between the two LLM supersets. To compare and visualize any shifts in performance, we aggregated all validation and test Dice scores from each year's LLM-generated model's outputs for each dataset and plotted them as boxplots for 2024 vs. 2025.

3.8 Reasoning vs Non-Reasoning Models

To test whether "reasoning-optimized" architectures exhibit better performance and learning dynamics than LLMs without CoT reasoning or prompt decomposition or more "general-purpose" LLMs, we partitioned the same LLMs into two groups. Reasoning models are comprised of: GPT o1-Preview (2024), Claude 4 Sonnet, DeepSeek R1, GPT o3, GPT o4-mini-high, Gemini 2.5 Pro, Grok 3 Mini, Llama 4 Maverick, Mistral Medium 3, and Qwen 3_235B (2025). Non-reasoning models: GPT-4o, GPT4, Llama 3.1 405B, Gemini 1.5 Pro, Copilot, Claude 3.5 Sonnet, Bing Microsoft Copilot (2024), plus DeepSeek V3 and Grok 3 (2025). We again averaged losses per epoch across models within each group, calculated and plotted their mean and standard deviations (± std), and directly compared how reasoning-oriented designs differ in training behavior and generalization on validation.

Separately, we appended the validation and test Dice scores for each group and visualized them as boxplots to highlight differences in learning dynamics tied to architectural focus.

3.9 Run-to-Run Variability Ablation Study

Considering the stochastic nature of LLMs, in order to ensure prompt generalizability and evaluate the stability and consistency of our LLM-driven pipelines across multiple runs, we performed a targeted ablation study on two representative sub-models of the same model: GPT-4o (non-reasoning) and GPT o4-mini-high (Chain-of-Thought reasoning), by re-prompting and repeating the full generation, training, and evaluation process 10 times on each. Any referencing and memory features were turned off from the model settings for all experiments. The models were re-prompted from scratch ten times separately to generate the dataset, model, training, and main scripts. We then trained each resulting U-Net pipeline as before, recorded any runtime errors, and evaluated performance on the shared 10% hold-out test sets. Across the ten runs per model, we tracked variability in the error counts and types during script execution, architectural and hyperparameter choices, as well as model performance (losses and test Dice scores). The inter-and intra-model variability was then compared and visualized.

3.10 Controlled Epoch Evaluation

Each LLM's chosen number of training epochs is treated as part of the automated hyperparameter selection. However, since allowing variable training budgets can introduce bias into comparative performance, we conducted an additional "constant-epochs" experiment, where we re-ran all LLM-generated pipelines as well as the nnU-Net v2 baseline on the BAGLS dataset for a fixed 120 epochs, keeping every other LLM-suggested hyperparameter unchanged. This allows us to assess whether the observed differences under each model's self-selected schedule persist when training duration is fixed.

3.11 Statistical Analysis

To assess whether newer (2025) or reasoning-enabled LLMs achieved higher segmentation accuracy than their counterparts, we compared the per-dataset Dice score distributions using one-sided Mann–Whitney U tests. The null hypothesis in each independent test was that the two groups' Dice score distributions have equal medians (i.e., no improvement for the 2025 or reasoning-enabled models) with $\alpha = 0.05$. For each dataset, we report the U-statistic, the raw p-value, and whether $p < \alpha$, indicating statistical significance. For the run-to-run variability testing and ablation study, we did the same to statistically compare one single reasoning (GPT o4-mini-high) and one non-reasoning model (GPT-4o), by using one-sided Mann-Whitney U tests on their average losses per epoch and mean Dice scores per batch for each group, across the 10 runs.

All LLM-generated scripts, logs including CoT, visualizations, prompt engineering log, model-selected architecture, hyperparameter comparison tables, error tracking and further analyses are provided in our GitHub repository: `https://github.com/ankilab/LLM_based_Segmentation.git`.

4. Results

We present a comprehensive comparison of LLM-generated medical image segmentation pipelines across multiple dimensions, including error rates during code execution, training and validation convergence behavior, and segmentation performance on the six diverse medical imaging datasets (see Figure 1). Results are further stratified by model generation (2024 vs. 2025) and reasoning capability (reasoning vs. non-reasoning models), along with a run-to-run variability test on one reasoning and one non-reasoning model of the same family (GPT) to asses the robustness of the findings.

Our first step was to evaluate whether the LLM-generated code executed directly or if errors were encountered during

runtime. The errors included syntax errors, import issues, size or type mismatches, and data handling errors, which were recorded, categorized, and fed back into the LLM for self-correction (Figure 2).

In the 2024 LLMs (Figure 2a), Gemini 1.5 Pro and Llama 3.1 405B produced the highest number of overall errors, with the most prevalent error types being `RuntimeError` and `ValueError`, whereas GPT o1-Preview (reasoning) and Claude 3.5 Sonnet produced fully executable code on their first attempt without any errors. Certain LLM-generated scripts (Llama 3.1 and Gemini 1.5 Pro) required more intensive debugging, and in some cases, supplementary information had to be provided during error feedback (see LLM chat logs on GitHub) to help the model break out of repetitive error loops (Arjomandi et al., 2025), thereby reducing the overall efficiency of the generated medical image analysis workflows by these models.

In contrast, the 2025 models (Figure 2b) demonstrated a substantial reduction in error frequency and severity, with the maximum total number of errors being 4 in Grok 3 mini, Qwen 3, and Llama 4 Maverick, making `NameError` the most prevalent one. Several reasoning-enabled LLMs (including DeepSeek, GPT o4-mini-high and Mistral medium 3) produced 1-3 errors (in case of DeepSeek R1 being not an error but the miscalculation of mean appended Dice scores per batch for validation and test sets, unlike the instructions, saving for each sample instead of mean per batch, which needed manual editing), while some newer versions of earlier models demonstrated significant gains in robustness. For instance, Gemini 2.5 Pro and Llama 4 achieved zero and four errors, respectively, in contrast to their earlier counterparts, with the highest number of total errors of 15 and 13, respectively, representing a substantial improvement from the previous generation. Importantly, models like Grok 3, Gemini 2.5 Pro, Claude 4 Sonnet, and GPT o3 generated code that executed flawlessly "out-of-the-box" without requiring corrections. Interestingly, Grok 3 exhibited no errors, whereas its smaller reasoning-enabled variant Grok 3 mini encountered a limited number of issues.

Overall, reasoning-augmented 2025 LLMs not only produced fewer errors but also resolved errors more effectively when feedback loops were used, without the need for additional input. The majority of errors in these models arose from dataset-specific preprocessing steps—such as handling array-to-tensor conversions and data loading routines—rather than core architectural issues, reflecting these models' improved capacity to adapt code logic across varied imaging modalities, and in contrast to the higher error counts and diversity of error types in the 2024 LLMs, highlights their capacity for autonomous debugging and code synthesis.

After confirming that the LLM-generated scripts run without causing any errors, we investigated the training and validation process of all models on each dataset. Although each LLM employs unique training configurations and architectural choices, the validation loss curves highlight clear differences in performance dynamics over the training epochs, where the number of epochs was chosen by each LLM. As shown in Figure 3 (note that the absolute y-values are not directly comparable across pipelines, as curves show loss values from different formulations (Table 1 and 2) and should be interpreted only in terms of relative convergence speed and cohort-level trends), for the BAGLS and Uterine Myoma datasets as examples, the 2025 LLMs demonstrated improved convergence behavior and lower final validation losses compared to their 2024 counterparts, and in some cases faster convergence compared to the baseline.

The same can be observed across all datasets (Appendix A and B, Figures 11 and 12). The convergence behavior of reasoning-enabled models (such as GPT o4-mini-high, GPT o3, Mistral Medium 3 and Qwen 3), as well as Grok 3 (non-reasoning), achieved rapid reductions in validation loss, reaching levels below 10^{-2} within the first 15-20 epochs. Claude 4 Sonnet and Gemini 2.5 pro closely followed nnU-Net, reaching convergence in fewer epochs than the baseline for most datasets, followed by DeepSeek R1 (reasoning) and V3 (non-reasoning) with medium convergence. Certain models, such as Llama, the DeepSeek models, and Grok 3 Mini exhibited less consistent training dynamics, with noticeable fluctuations in validation loss and poorer convergence toward lower loss values, sometimes with mild instability in early training epochs. In contrast, some 2024 models showed slower convergence and plateaued at higher loss values, with Bing Microsoft Copilot and GitHub Copilot showing almost no convergence, indicating suboptimal training dynamics, whereas earlier versions of the 2025 high-performing LLMs, such as GPT o1-Preview, GPT-4o, and Claude 3.5 Sonnet, already demonstrated strong performance, achieving validation loss trajectories that were comparable to their 2025 successors and outperforming most other models from the same generation.

Comparing the models under a fixed number of epochs, Figure 4 shows validation losses across 120 epochs on the BAGLS dataset for all models. In addition, we indicated the number of epochs suggested by each LLM in each training loss. As in our original comparison, reasoning-enabled 2025 models still converge faster and to lower loss values than non-reasoning or 2024 models, even when all models share the same training budget. The relative convergence speeds and final loss levels remain consistent with the patterns seen under each model's self-selected epoch count, indicating that variable epoch schedules do not notably bias our core performance comparisons.

To evaluate performance and segmentation accuracy, we computed Dice coefficients for all models on the held-out test sets across the six datasets, comparing LLM-generated

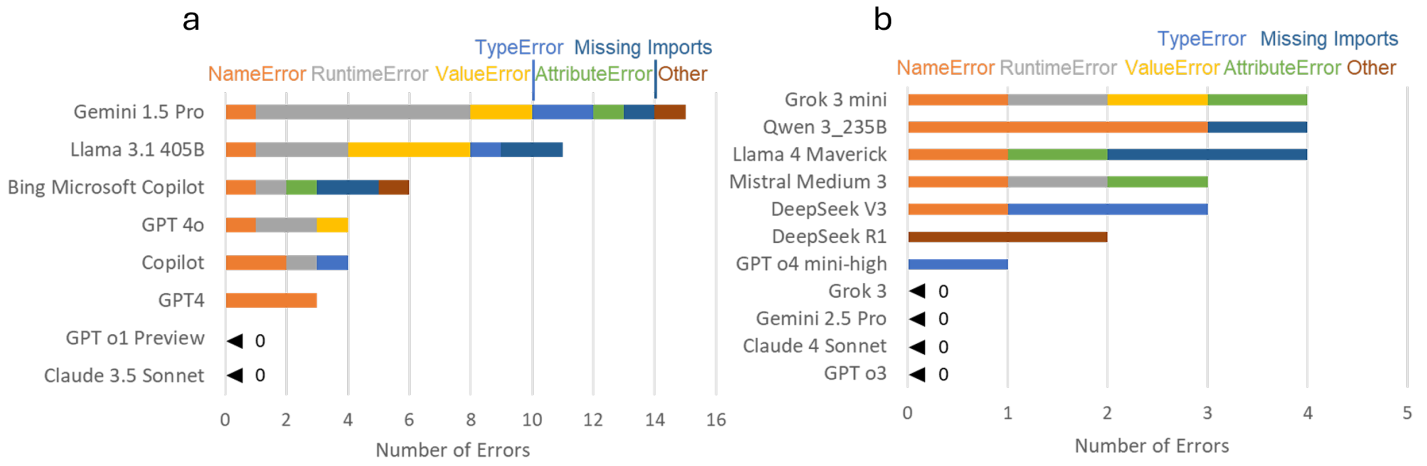


Figure 2: Comparison of the number and type of errors across LLMs until successful code execution for a) 2024 selected models and b) 2025 selected models.

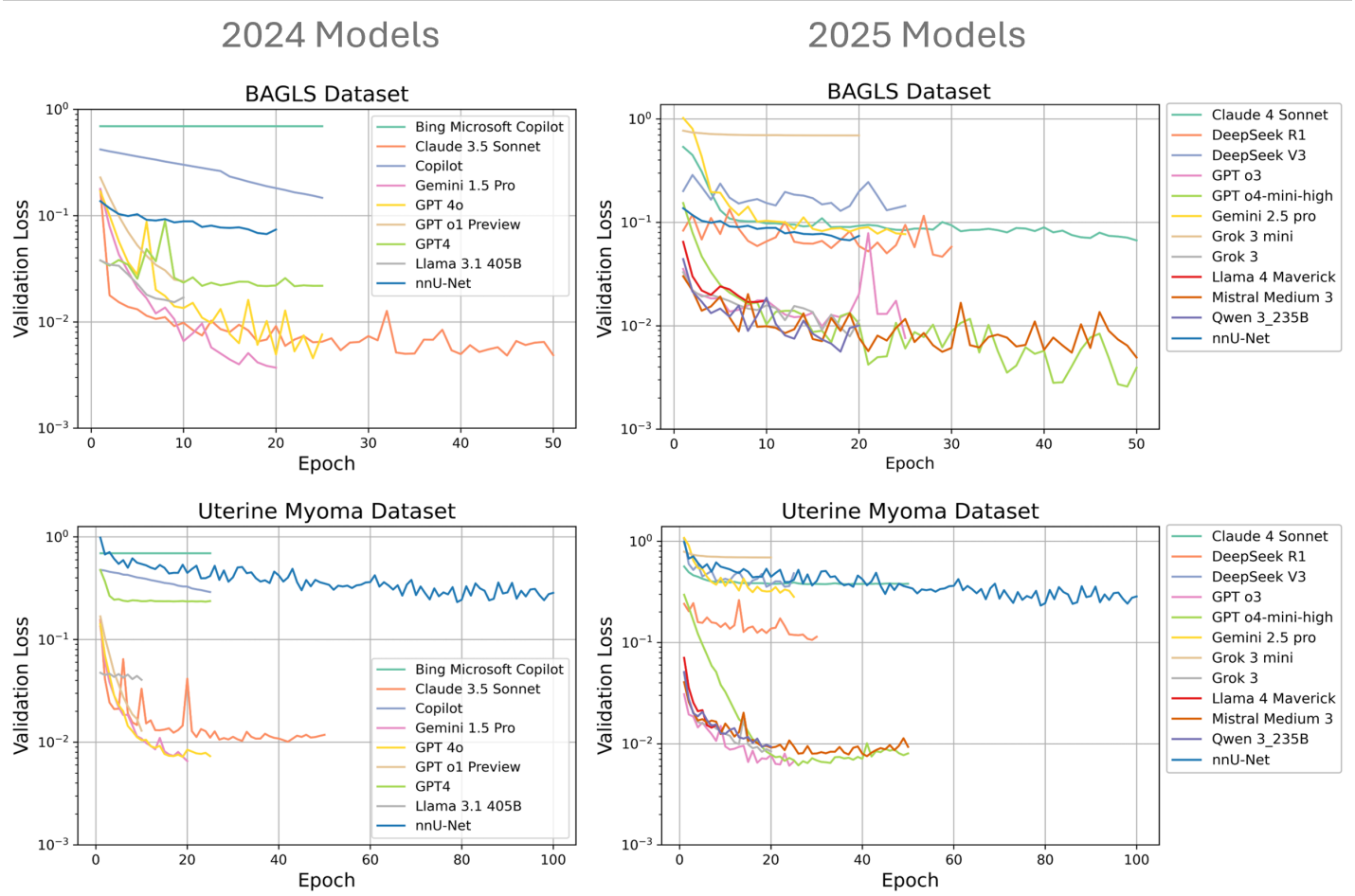


Figure 3: Validation loss per epoch for 2024 (left) and 2025 (right) LLM-generated segmentation pipelines, exemplarily shown on the BAGLS dataset (top) and Uterine Myoma MRI dataset (bottom), with nnU-Net v2 (total loss shown) as baseline (blue). Number of epochs were determined by each LLM. Note that curves show loss values from different formulations (BCE, BCE+Dice, etc.) and absolute values are not directly comparable.

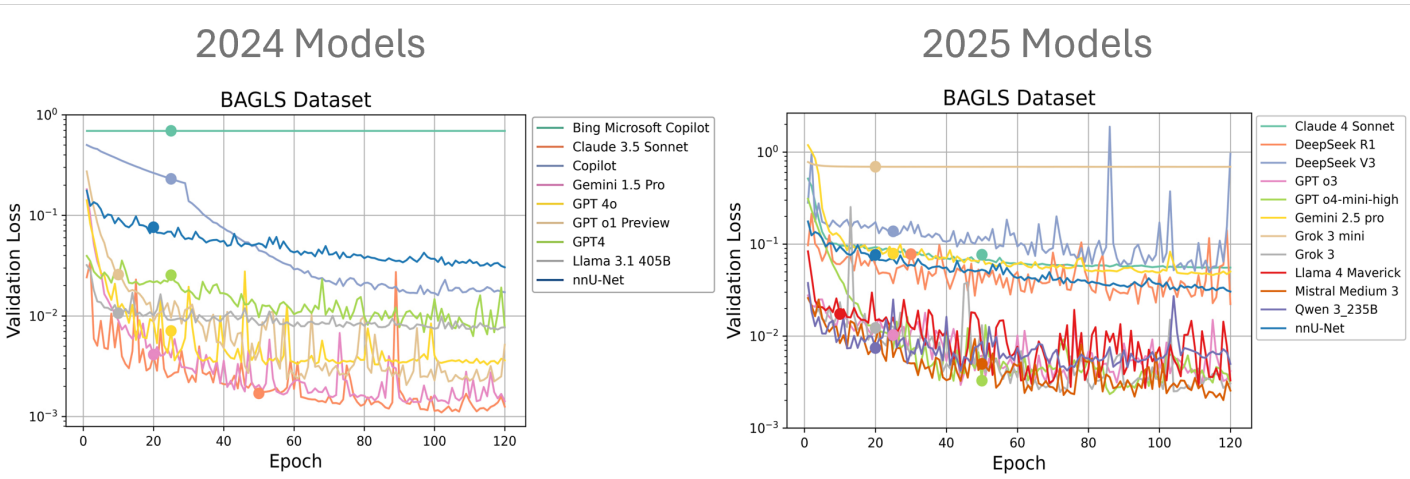


Figure 4: Validation loss per epoch for 2024 and 2025 LLM-generated segmentation pipelines, exemplarily shown on the BAGLS dataset for a constant number of epochs, with nnU-Net v2 (total loss shown) as baseline (blue). Initial number of epochs chosen by each model is marked with a dot on the respective curves. Note that curves show loss values from different formulations (BCE, BCE+Dice, etc.) determined by each model and absolute values are not directly comparable.

pipelines to the nnU-Net v2 baseline (Figure 5. For visualization with Dice scores sorted by medians separately on each dataset, see Appendix C, Figure 13). The analysis also distinguishes reasoning-enabled and non-reasoning models within and across generations.

For the 2024 models (Figure 5), GPT o1-Preview, the only reasoning-enabled LLM in this generation, consistently achieved high Dice scores with medians close to nnU-Net v2 across most datasets, along with Gemini 1.5 Pro—despite it's high error rate—, and Claude 3.5 (non-reasoning) showing moderate to high performance (medians ~ 0.75–0.90), being the highest performing models on the complex retinal vessel dataset, and in the case of the Bolus dataset even outperforming the nnU-Net baseline along with the GPTs. GPT-4o and GPT-4 also performed well as non-reasoning models, achieving median Dice scores between 0.75 and 0.90 but with slightly more variability, especially on complex tasks such as retinal vessel segmentation, where they performed poorly. GitHub Copilot and Bing Copilot performed poorly across all datasets, often with medians below 0.30, following Llama with only a moderate performance on the Bolus and Skin Cancer dataset. The results can also be observed qualitatively in Figure 6, where the inferred prediction masks from the more advancede GPT models (o1-preview and 4o) as well as Claude 3.5 and Gemini 1.5 show Dice scores close to the nnU-Net prediction (except for the retina dataset where most models struggled), whereas Copilot and Bing Copilot failed learning, and Llama 3.1 was unable to capture detailed spatial information due to the small size (0.53M parameters) and shallow chosen bottleneck (256 ch) (Table 2).

Most 2025 models (Figure 5) demonstrated substantial improvements and higher mean Dice scores (in case of the Retina and BAGLS dataset significantly higher, with $U = 70.0$, $p = 0.0016$, and $U = 65.0$, $p = 0.0045$ respectively), particularly among reasoning-enabled LLMs. Claude 4 Sonnet and GPT o4-mini-high, GPT o3 achieved the highest Dice scores, matching, or on the Bolus and Skin Cancer datasets slightly exceeding, nnU-Net. Other reasoning models, including Mistral Medium 3 and Qwen 3, also performed robustly and consistently high across modalities, which can also be seen in the qualitative inference (Figure 7). Greater variability in Dice scores was observed on the more complex Retina and Uterine Myoma datasets, reflecting the increased difficulty of fine-structure, slice-wise segmentation of volumetric data or the presence of thin or ambiguous boundaries (Figure 5). Llama 4 Maverick showed similarly poor performance to its earlier 3.1 version on complex datasets and slightly improved performance on other datasets, particularly the Uterine Myoma dataset, while Gemini 2.5 Pro, despite running error-free, performed comparably or slightly worse than its predecessor in some cases (Utreine Myoma and Skin Cancer dataset) (Figure 7), and while DeepSeek V3, as a non-reasoning model, performed worse than its reasoning-enabled counterpart R1 on the more complex datasets (Retina and Uterine Myoma dataset) but better on all other datasets, Grok 3 mini (reasoning-enabled) also surprisingly underperformed compared to the non-reasoning Grok 3 on most datasets (Figure 5, 7).

Taken together, these results indicate that reasoning-augmented LLMs—except in few cases—more often produce more accurate and stable segmentation pipelines across diverse tasks, while non-reasoning models offer competitive but slightly less consistent baselines. The 2025 generation

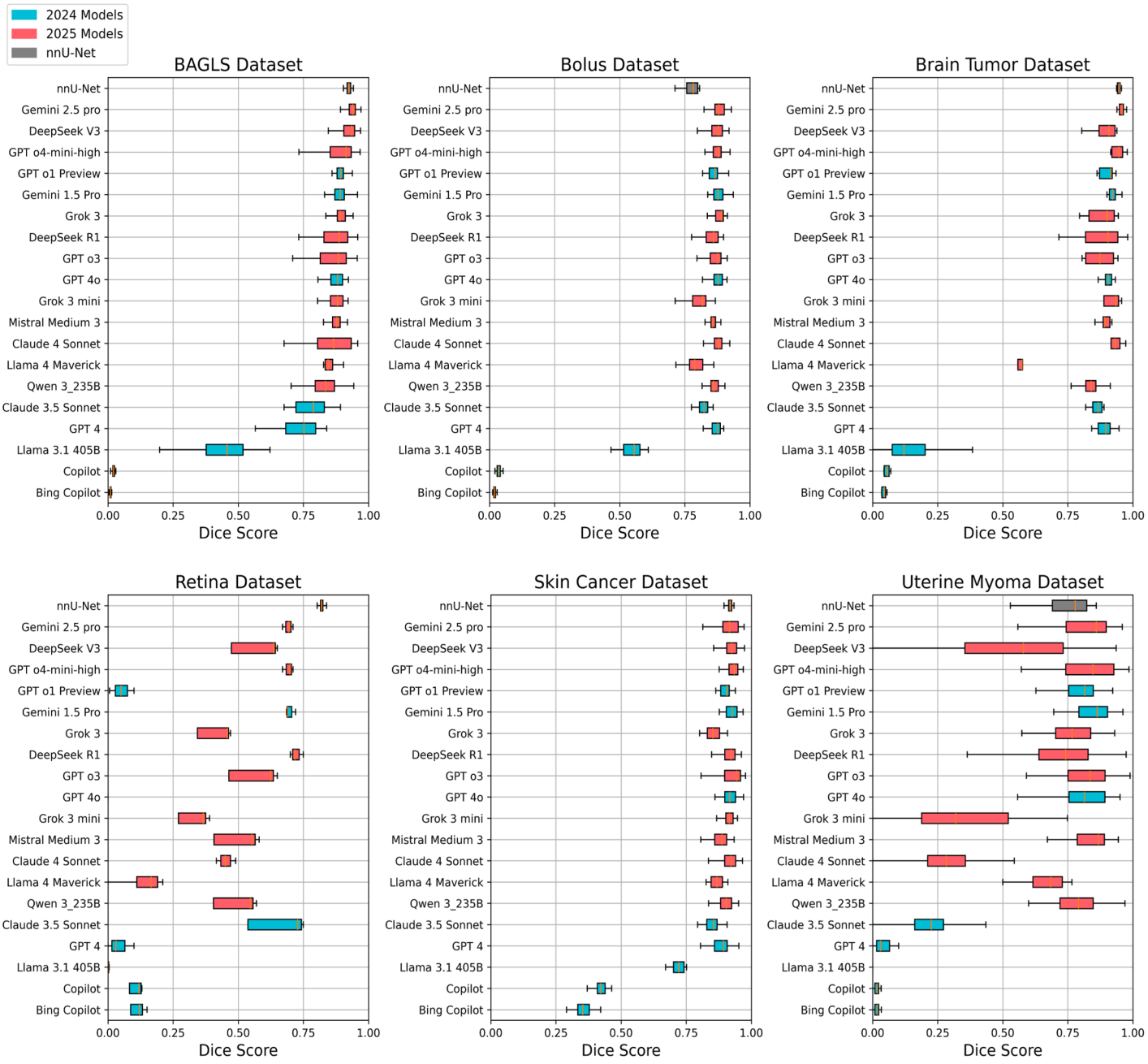


Figure 5: Comparison of the test Dice scores across all selected 2024 (blue) and 2025 (red) models, for each of the six datasets. Dice scores are sorted by median from high to low for BAGLS, other plots following the same order for easier visual comparison. nnU-Net is shown at the top as the baseline.

as a whole marks a remarkable step forward, bringing LLM-generated pipelines closer to expert-designed frameworks matching or outperforming nnU-Net.

We were next interested in architectural design across LLMs. Comparing the model architectures, selected hyperparameters, data loading, training, and main scripts, the 2025 LLM-generated pipelines showed both convergent design patterns and meaningful differences in their architectural and training configurations (Table 1). All 2025 models operated on 256×256 inputs, used four encoding and four decoding stages (except Llama 4 Maverick, which used three), and adopted U-Net–style double-convolutional blocks with BatchNorm and ReLU activations. However, the depth of the bottleneck and the number of encoder channels varied: DeepSeek R1, DeepSeek V3, and Claude 4 Sonnet employed a 1024-channel bottleneck, whereas GPT o3 and GPT o4-mini-high used only 512 channels. Qwen 3_235B stood out by inserting a CBR block (Conv→BN→ReLU twice) in place of the standard double-conv.

All models were trained with the Adam optimizer, but

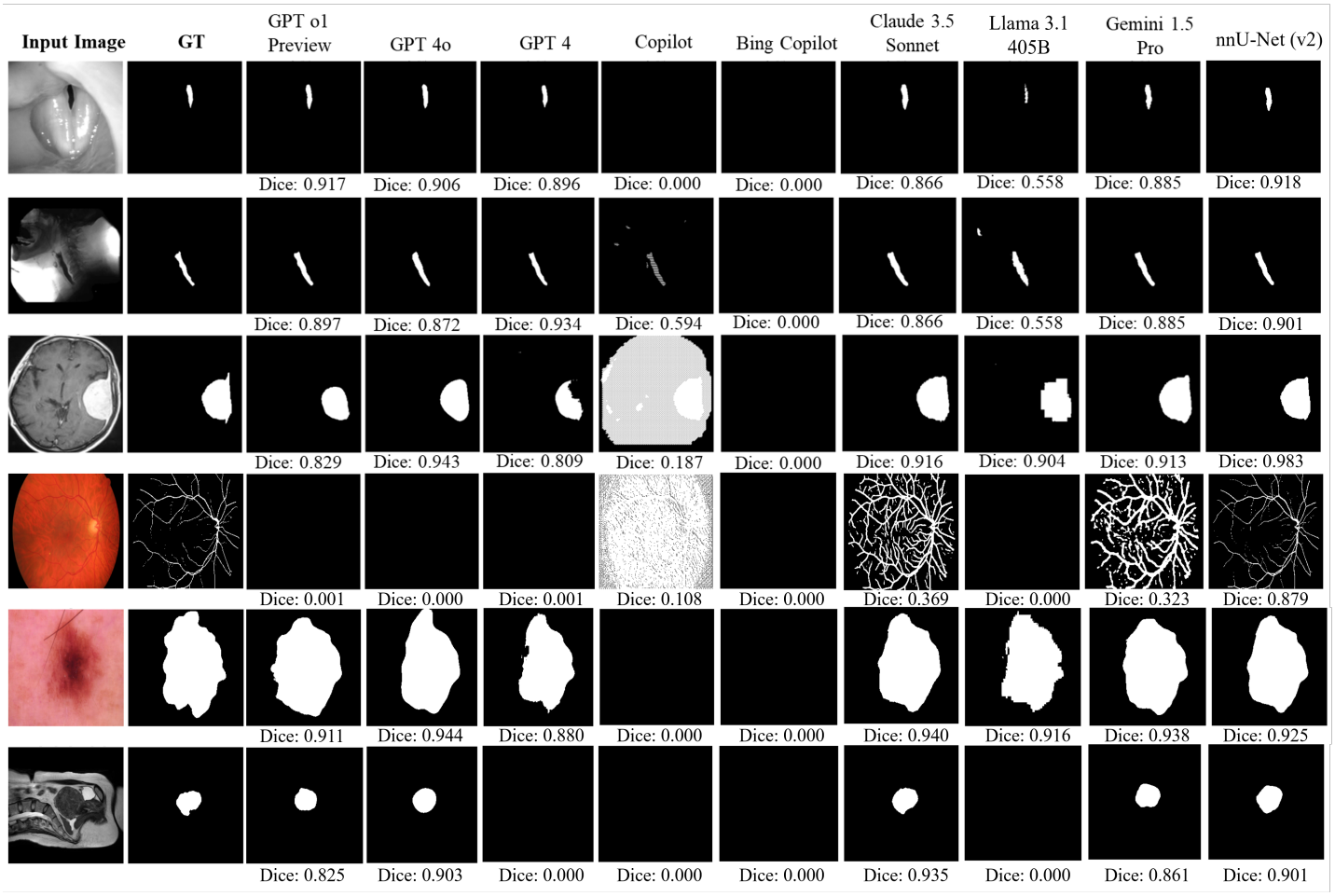


Figure 6: Inferred prediction masks comparison across 2024 selected models. Each horizontal set shows the performance of each model on an unseen, rather challenging sample image from each of the six datasets, from top to bottom row: BAGLS, Bolus, Brain Tumor, Retina, Skin Cancer, and Uterine Myoma datasets.

differed in learning rate and the use of weight decay. GPT o3, DeepSeek V3, Mistral Medium 3, and Qwen 3_235B used $\mathrm{lr} = 10^{-3}$, while GPT o4-mini-high, Claude 4 Sonnet, Gemini 2.5 Pro, and Grok 3 adopted $\mathrm{lr} = 10^{-4}$; DeepSeek R1 and Claude 4 Sonnet also applied a small weight decay of 10^{-5}. Batch sizes ranged from 8 for most models, 16 for Grok 3, 3-mini, Mistral Medium 3, and Qwen 3, up to 32 (Llama 4 Maverick), reflecting memory constraints. Epoch counts were likewise heterogeneous, with GPT o4-mini-high, Claude 4 Sonnet, and Mistral Medium 3 training for 50 epochs versus Llama 4 Maverick's 10 epoch choice, similar to the 2024 version Llama 3.1.

The 2025 models employed three main categories of loss functions. Four models—GPT o3, GPT o4-mini-high, Grok 3 Mini, and Qwen 3_235B—used a straightforward `BCEWithLogitsLoss`. Five models combined cross-entropy with Dice: DeepSeek V3 applied a `BCEDiceLoss`, DeepSeek R1 used a `DiceBCELoss` (70% BCE + 30% Dice), Claude 4 Sonnet adopted a balanced combined loss ($\alpha = 0.5$), and Gemini 2.5 Pro coupled `BCEWithLogitsLoss` + `DiceLoss`. The remaining three models—Grok 3, Llama 4 Maverick, and Mistral Medium 3—relied on standard `BCELoss`, reflecting a simpler pixel-wise objective.

These configuration choices impacted both parameter counts and training times. Models with deeper bottlenecks (e.g., DeepSeek V3/R1, Claude 4 Sonnet, GPT o4-mini-high) had ≈ 31 million parameters and required longer to train, compared to lighter models (GPT o3, Llama 4 Maverick) with $\approx$ 7–8 million parameters.

The 2024 models, as summarized in Table 2, also adhered loosely to a U-Net template, most employing four encoder–decoder stages, `DoubleConv` blocks (`BatchNorm2d` in GPT-4o, Copilot, and Gemini 1.5 Pro), and `ConvTranspose2d` upsampling, with bottleneck widths ranging from 256 to 1024 channels and parameter counts spanning 0.5 M to over 31 M; all models trained with a simple binary cross-entropy loss (Arjomandi et al., 2025). In contrast, the 2025 models uniformly applied `BatchNorm2d`

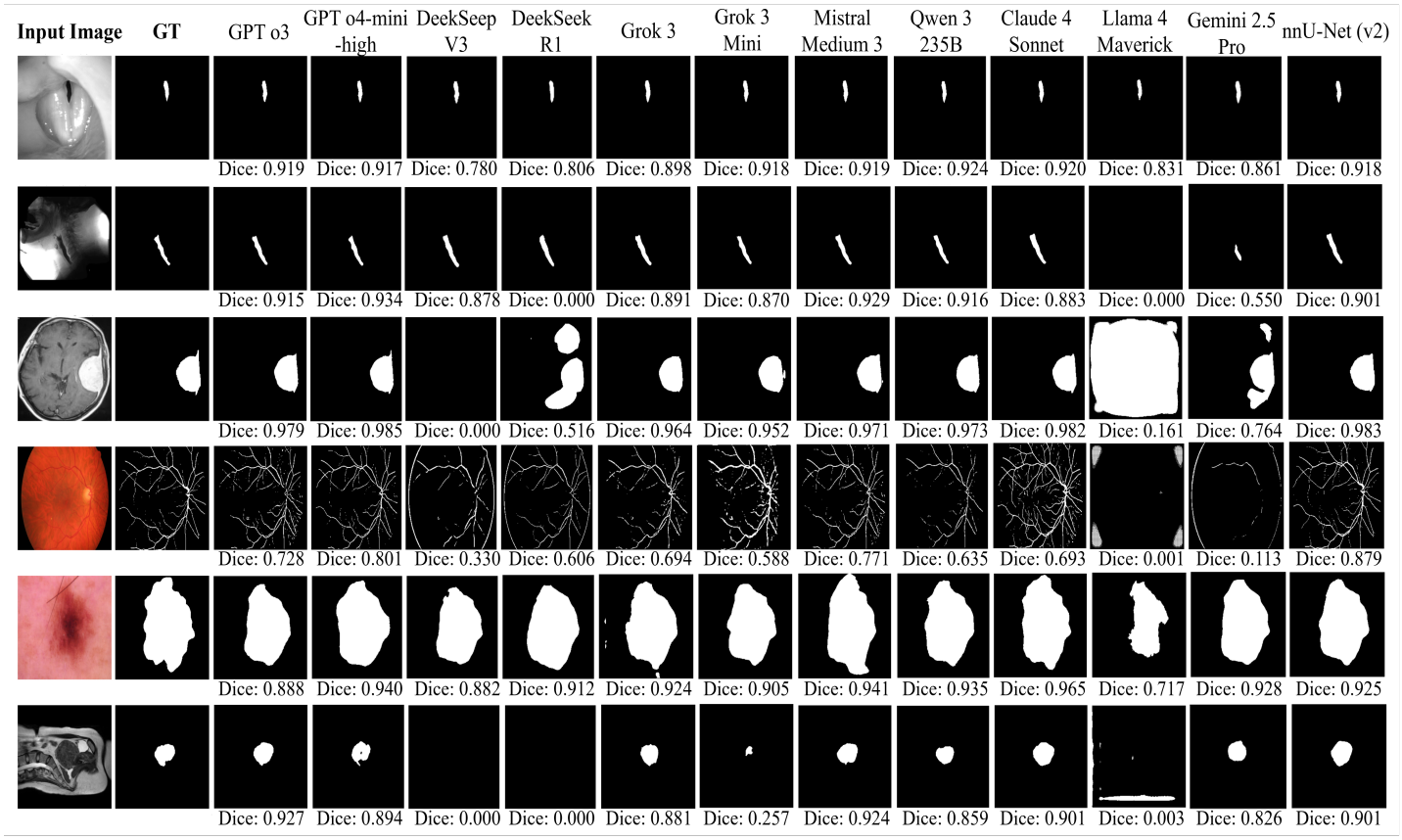


Figure 7: Inferred prediction masks comparison across 2025 selected models. Each horizontal set shows the performance of each model on an unseen, rather challenging sample image from each of the six datasets, from top to bottom row respectively BAGLS, Bolus, Brain Tumor, Retina, Skin Cancer, and Uterine Myoma datasets.

after each convolution, and consolidated around larger 1024-channel bottlenecks for reasoning models, paired with mixed `BCE+Dice` objectives, resulting in longer training but higher final Dice scores, whereas lighter non-reasoning variants used 512-channel bottlenecks and pure `BCEWithLogitsLoss` with smaller parameter counts, yielding faster training but slightly lower segmentation accuracy, reflecting a shift toward more consistent, high-capacity architectures and more sophisticated loss functions.

Figure 8 and Table 3 summarize the aggregate performance differences between reasoning and non-reasoning LLMs, as well as the 2024 and 2025 model cohorts. Reasoning-enabled models converged notably faster and to lower validation losses than non-reasoning models (Fig. 8a), and the 2025 generation outperformed the 2024 models across epochs (Fig. 8b). In terms of segmentation accuracy, reasoning models achieved higher median Dice scores and tighter distributions on both validation and test sets (U = 15, p = 0.002; U = 18, p = 0.005) compared to non-reasoning models (Fig. 8c). Similarly, the 2025 models exhibited significantly better Dice performance than the 2024 generation on validation and test data (U = 12, p = 0.001; U = 14, p = 0.003), underscoring the impact of both reasoning capabilities and architectural advances in the newer cohort (Fig. 8d).

Also as shown in Table 3 in the test-set evaluation, median Dice scores improved from 2024 to 2025 in all six tasks, with reasoning-enabled models showing slightly higher median Dice scores than non-reasoning models (significantly higher on the Retina and BAGLS datasets). All other statistical tests for 2024 vs. 2025 or reasoning vs. non-reasoning for each dataset yielded $p > 0.05$.

To assess how sensitive our LLM-generated pipelines are to nondeterministic code generation and probe the stability of our LLM-driven pipelines, we repeated the end-to-end process, including prompting, code generation, training, and evaluation, ten times each for GPT-4o (non-reasoning) and GPT-o4-mini-high (CoT reasoning), as being two representative models of the same family. Table 4 and 5 summarize the configuration choices and outcomes across all 10 runs for the non-reasoning and reasoning model, respectively.

Across both models, the batch size 8 and input resolution (256×256) remained identical in every run, confirming

Table 1: Architectural and training configurations of 2025 LLM–generated segmentation pipelines (split into two sub-tables). Total training time for each model shows the mean training time (± standard deviation) over all six datasets.

Feature	GPT o3	GPT o4-mini-high	DeepSeek V3	DeepSeek R1	Claude 4 Sonnet
Batch Size	8	8	8	8	8
Epochs	25	50	25	30	50
Optimizer & LR	Adam(1e-3)	Adam(1e-4)	Adam(1e-3)	Adam(1e-3,wd=1e-5)	Adam(1e-4,wd=1e-5)
Loss Function	BCEWLogits	BCEWLogits	BCEDice	DiceBCE (70/30)	CombLoss (α=0.5)
# Encoder Stages	4	4	4	4	4
# Decoder Stages	4	4	4	4	4
Conv Block	(Conv→BN→ReLU)×2	DoubleConv	DoubleConv	DoubleConv	DoubleConv
Bottleneck	256→512 ch	1024 ch	512 ch	1024 ch	1024 ch
Final Layer	Conv2d(32→1)	Conv2d(64→1)	Conv2d(64→1)	Conv2d(64→1)	Conv2d(64→1)
Encoder Channels	32,64,128,256	64,128,256,512	64,128,256,512,512	64,128,256,512,1024	64,128,256,512,1024
Decoder Channels	256,128,64,32	512,256,128,64	512,256,128,64	512,256,128,64	512,256,128,64
Total Parameters	7.8 M	31.0 M	31.4 M	31.4 M	31.0 M
Training Time	0:40:44(± 14.0')	1:06:08(± 23.4')	0:26:34(± 16.4')	0:43:49(± 25.6')	1:30:01(± 32.7')

Feature	Gemini 2.5 Pro	Grok 3 mini R	Grok 3	LLaMA 4 Maverick	Mistral Medium 3	Qwen 3_235B
Batch Size	8	16	16	32	16	16
Epochs	25	20	20	10	50	20
Optimizer & LR	Adam(1e-4)	Adam(1e-4)	Adam(1e-3)	Adam(1e-3)	Adam(1e-3)	Adam(1e-3)
Loss Function	BCE+Dice	BCEWLogits()	BCE	BCE	BCE	BCEWLogits
# Encoder Stages	4	4	4	4	4	4
# Decoder Stages	4	4	4	3	4	4
Conv Block	DoubleConv	DoubleConv	DoubleConv	DoubleConv	DoubleConv	CBR
Bottleneck	1024 ch	512→1024→1024	512→1024	256→512	512→512	512→1024 ch
Final Layer	Conv2d(64→1)	Conv2d(64→1)+Sig	Conv2d(64→1)+Sig	Conv2d(64→1)+Sig	1×1 Conv(64→1)+Sig	1×1 Conv(64→1)
Encoder Channels	64,128,256,512	64,128,256,512	64,128,256,512	64,128,256,512	64,128,256,512	1,64,128,256,512
Decoder Channels	512,256,128,64	512,256,128,64	512,256,128,64	256,128,64	512,256,128,64	1024,512,256,128,64
Total Parameters	17.3 M	31.0 M	31.0 M	7.7 M	13.4 M	31.0 M
Training Time	0:29:21(± 18.1')	0:24:21(± 14.9')	0:35:13(± 13.8')	0:12:24(± 7.8')	0:54:15(± 41.8')	0:22:48(± 11.3')

Table 2: Architectural and training configurations of 2024 LLM–generated segmentation pipelines. Total training time for each model shows the mean training time (± standard deviation) over all six datasets.

Feature	GPT-4	GPT-4o	GPT-o1	Claude 3.5	Copilot	Bing Copilot	LLaMA 3.1	Gemini 1.5 Pro
Batch Size	16	8	16	16	16	16	32	8
Epochs	25	25	10	50	25	25	10	20
Optimizer & LR	Adam(1e-3)	Adam(1e-4)	Adam(1e-4)	Adam(1e-3)	Adam(1e-4)	Adam(1e-4)	Adam(1e-3)	Adam(1e-4)
Loss Function	BCEWLogits	BCE	BCE	BCE	BCE	BCE	BCE	BCE
# Encoder Stages	4	5	4	4	4	4	3	4
# Decoder Stages	3	4	4	4	4	4	3	4
Conv Block	DoubleConv	CBR×2	DoubleConv	DoubleConv	CBR×2	Conv→ReLU×2	Conv→ReLU	CBR×2
Bottleneck	512 ch	1024 ch	1024 ch	1024 ch	512 ch	512 ch	256 ch	1024 ch
Final Layer	Conv2d(64→1)	Conv2d(64→1)	Conv2d(64→1)	Conv2d(64→1)	Conv2d(64→1)	Conv2d(64→1)	Conv2d(64→1)	Conv2d(64→1)
Encoder Channels	64,128,256,512	64,128,256,512,1024	64,128,256,512,1024	64,128,256,512,1024	64,128,256,512	64,128,256,512	64,128,256	64,128,256,512
Decoder Channels	512,256,128,64	512,256,128,64	512,256,128,64	512,256,128,64	512,256,128,64	512,256,128,64	256,128,64	512,256,128,64
Total Parameters	7.7 M	31.0 M	31.0 M	31.0 M	6.2 M	6.1 M	0.53 M	31.0 M
Training Time	0:24:34(± 16.7')	0:15:49(± 7.5')	0:13:00(± 7.8')	1:28:05(± 32.1')	0:03:04(± 2.4')	0:08:17(± 3.9')	0:05:54(± 4.5')	0:17:46(± 9.8')

Table 3: Comparison of median Dice scores (with IQR) between 2024 and 2025 models across six datasets. Mann–Whitney U test results are reported for statistical comparison. Significant results are marked in bold.

Dataset	**2024 Median (IQR)**	**2025 Median (IQR)**	**Δ Median**	**U**	**p-value**	**Significant?**
BAGLS	0.8396 (0.4793–0.8960)	0.8910 (0.8408–0.9277)	+0.0514	65.0	0.0454	**Yes**
Bolus Swallowing	0.8447 (0.5919–0.8778)	0.8691 (0.8470–0.8864)	+0.0244	52.0	0.2724	No
Brain Tumor	0.9033 (0.7884–0.9360)	0.9123 (0.8381–0.9420)	+0.0090	54.0	0.2211	No
Skin Lesion	0.9062 (0.8296–0.9229)	0.9124 (0.8821–0.9357)	+0.0062	61.0	0.0864	No
Uterine Myoma	0.7391 (0.0399–0.8348)	0.7540 (0.5591–0.8614)	+0.0149	54.0	0.2211	No
Retina Fundus	0.5413 (0.3230–0.6500)	0.8176 (0.8033–0.8234)	+0.2763	70.0	0.0164	**Yes**

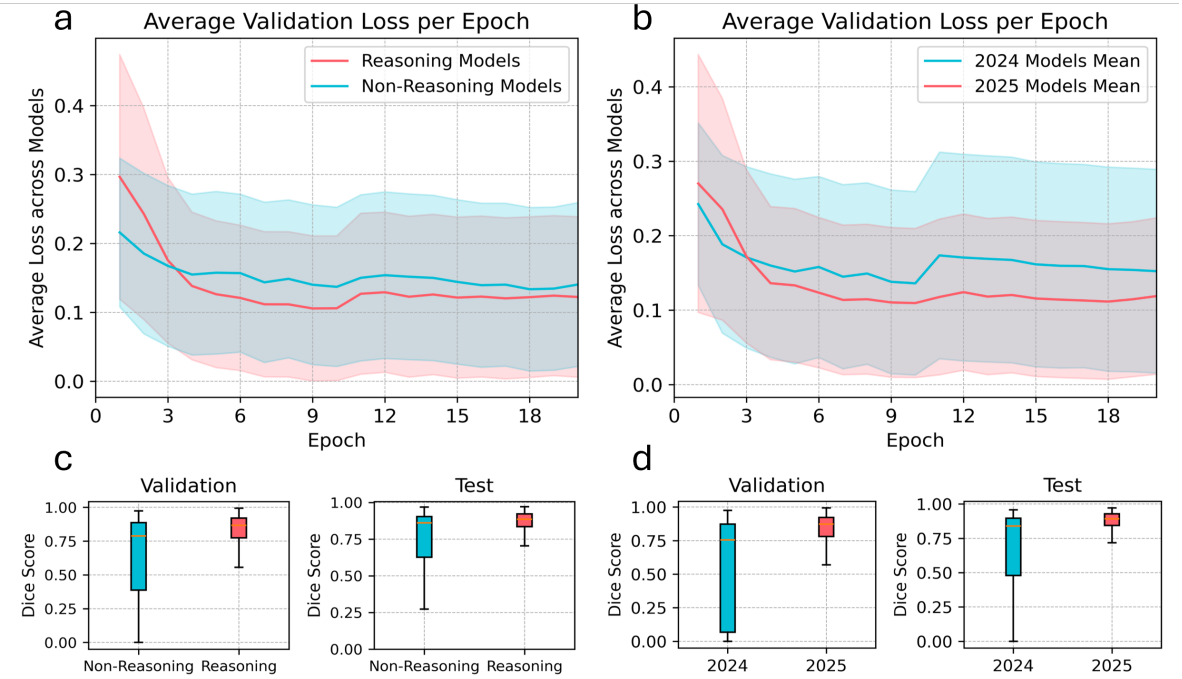


Figure 8: Mean validation loss per epoch for (a) reasoning (GPT o1-Preview (2024), and Claude 4 Sonnet, DeepSeek R1, GPT o3, GPT o4-mini-high, Gemini 2.5 Pro, Grok 3 Mini, Llama 4 Maverick, Mistral Medium 3, and Qwen 3_235B (2025)) (red) vs. non-reasoning (GPT-4o, GPT4, Llama 3.1 405B, Gemini 1.5 Pro, Copilot, Claude 3.5 Sonnet, Bing Microsoft Copilot (2024), plus DeepSeek V3 and Grok 3 (2025)) (blue) models, and (b) 2024 (blue) vs. 2025 (red) generations. (c) Validation and test Dice scores for reasoning vs. non-reasoning models (validation: U= 50.0, p=0.356; test: U=58.0, p=0.153), (not significant). (d) Validation and test Dice scores for 2024 vs. 2025 models (validation: U=57.0, p=0.014; test: U=65.0, p=0.045) (significant), shown exemplarily for the BAGLS dataset.

consistency in core data handling. However, the number of training epochs chosen by each run varied: GPT-4o split its runs between 20, 25, and 30 epochs (5 out of 10 runs choosing 20 epochs), while GPT-o4-mini-high opted for longer schedules (up to 50 epochs) 5 out of 10 runs, reflecting small architectural or hyperparameter differences. Optimizer choice remained fixed (`Adam`) for all runs, though learning rates occasionally toggled between 10^{-3} and 10^{-4}. Loss functions were uniformly `Binary Cross-Entropy (BCE)` for GPT-4o, whereas GPT-o4-mini-high alternated between `BCE` and `BCEWithLogitsLoss` in each run, otherwise showing notable consistency.

Architecturally, while every GPT-4o run adhered strictly to four encoding and four decoding stages, GPT o4-mini-high inserted a fifth encoder stage in five out of ten runs, producing deeper 5-stage U-Nets in those cases. Only the remaining GPT o4-mini-high runs retained the standard four-stage structure. This highlights that GPT-4o's generated architectures were uniform in depth, whereas GPT o4-mini-high—despite occasional variations—still delivered more consistent and higher segmentation performance (see Figure 9). Minor variations in the convolutional block implementation (e.g., `DoubleConv` vs. `CBR` wrappers with Batch normalization after every convolution) appeared in 2 of the 10 GPT-4o runs, but remained consistent over the 10 GPT-o4-mini-high runs, although these did not materially affect model capacity. Parameter counts remained ~ 31M for GPT-4o, but ranged between ~ 13–31M for GPT-o4-mini-high, in one case even introducing a larger bottleneck and raising its parameter count to ~ 125M, reflecting the exploratory nature of a reasoning model when given the same prompt, with freedom of choice in parameters.

Error occurrences also differed (Table 4 and 5): Like the initial run, GPT-4o generated one or two runtime/value errors in 6 of 10 runs, whereas GPT-o4-mini-high produced errors in only 4 runs, corrected by the model itself via self-debugging after one to two loops.

Despite these small differences, the test Dice scores for GPT-4o varied considerably (Figure 9) (median 0.87, IQR 0.77–0.95), and validation loss curves showed a higher spread and standard deviation (Figure 10.a). In contrast, GPT-o4-mini-high exhibited much tighter performance in the Dice scores (median 0.92, IQR 0.87–0.94) (Figure 9), and validation losses consistently fell below 0.02 after epoch 20, with a smaller standard deviation across all runs (Figure 10.b).

Statistically comparing the runs on the reasoning vs non-reasoning model, the Mann-Whitney U test showed no

Table 4: Architectural and training configurations of GPT-4o generated segmentation pipelines in 10 runs (split into two sub-tables). Training times are shown for the Brain Tumor dataset for comparison.

Feature	GPT 4o #1	GPT 4o #2	GPT 4o #3	GPT 4o #4	GPT 4o #5
Batch Size	8	8	8	8	8
Epochs	30	25	20	20	20
Optimizer & LR	Adam(1e-4)	Adam(1e-3)	Adam(1e-4)	Adam(1e-4)	Adam(1e-4)
Loss Function	BCE	BCE	BCE	BCE	BCE
# Encoder Stages	4	4	4	4	4
# Decoder Stages	4	4	4	4	4
Conv Block	Conv Block	DoubleConv	CBR	DoubleConv	CBR
Bottleneck	512→1024 ch	512→1024 ch	512→1024 ch	512→1024 ch	512→1024 ch
Final Layer	1×1 Conv+Sig	1×1 Conv+Sig	1×1 Conv+Sig	1×1 Conv+Sig	1×1 Conv+Sig
Encoder Channels	1,64,128,256,512	1,64,128,256,512	1,64,128,256,512	1,64,128,256,512	1,64,128,256,512
Decoder Channels	1024,512,256,128,64	1024,512,256,128,64	1024,512,256,128,64	1024,512,256,128,64	1024,512,256,128,64
Total Parameters	31.03 M	31.04 M	31.04 M	31.04 M	31.04 M
Training Time	0:08:54	0:07:48	0:06:44	0:06:12	0:06:14
Errors	0	1 ValueError	0	1 RuntimeError	1 ValueError

Feature	GPT 4o #6	GPT 4o #7	GPT 4o #8	GPT 4o #9	GPT 4o #10
Batch Size	8	8	8	8	8
Epochs	25	25	20	20	25
Optimizer & LR	Adam(1e-4)	Adam(1e-4)	Adam(1e-4)	Adam(1e-4)	Adam(1e-4)
Loss Function	BCE	BCE	BCE	BCE	BCE
# Encoder Stages	4	4	4	4	4
# Decoder Stages	4	4	4	4	4
Conv Block	DoubleConv	(Conv→BN→ReLU)×2	DoubleConv	DoubleConv	DoubleConv
Bottleneck	512→1024 ch	512→1024 ch	512→1024 ch	512→1024 ch	512→1024 ch
Final Layer	1×1 Conv+Sig	1×1 Conv+Sig	1×1 Conv+Sig	1×1 Conv+Sig	1×1 Conv+Sig
Encoder Channels	1,64,128,256,512	1,64,128,256,512	1,64,128,256,512	1,64,128,256,512	1,64,128,256,512
Decoder Channels	1024,512,256,128,64	1024,512,256,128,64	1024,512,256,128,64	1024,512,256,128,64	1024,512,256,128,64
Total Parameters	31.04 M	31.04 M	31.04 M	31.03 M	31.04 M
Training Time	0:08:04	0:07:46	0:06:15	0:06:07	0:07:49
Errors	1 ValueError	0	0	1 Other	1 ValueError

Table 5: Architectural and training configurations of GPT-o4-mini-high generated segmentation pipelines in 10 runs (split into two sub-tables). Training times are shown for the Brain Tumor dataset for comparison.

Feature	GPT o4 high #1	GPT o4 high #2	GPT o4 high #3	GPT o4 high #4	GPT o4 high #5
Batch Size	8	8	8	8	8
Epochs	20	20	50	25	20
Optimizer & LR	Adam(1e-4)	Adam(1e-3)	Adam(1e-4)	Adam(1e-3)	Adam(1e-3)
Loss Function	BCE	BCEWithLogits	BCE	BCEWithLogits	BCE
# Encoder Stages	4	4	5	5	5
# Decoder Stages	4	4	4	5	4
Conv Block	DoubleConv	DoubleConv	(Conv+BN+ReLU)×2	(Conv+BN+ReLU)×2	(Conv+BN+ReLU)×2
Bottleneck	512→1024 ch	512→1024 ch	512→1024 ch	1024→2048 ch	Down (512→512)
Final Layer	1×1 Conv+Sig	1×1 Conv	1×1 Conv	1×1 Conv	1×1 Conv
Encoder Channels	1→64→128→256→512	1→64→128→256→512	64,128,256,512,512	64,128,256,512,1024	64,128,256,512,512
Decoder Channels	512→256→128→64	512→256→128→64	512,256,128,64	1024,512,256,128,64	256,128,64,64
Total Parameters	31.38 M	31.04 M	13.39 M	125.8 M	13.39 M
Training Time	0:07:09	0:06:11	0:48:19	0:29:10	0:05:07
Errors	1 RuntimeError	0	0	2: Runtime+ValueError	2: Attribute+ValueError

Feature	GPT o4 high #6	GPT o4 high #7	GPT o4 high #8	GPT o4 high #9	GPT o4 high #10
Batch Size	8	8	8	8	8
Epochs	50	50	20	50	50
Optimizer & LR	Adam(1e-3)	Adam(1e-4)	Adam(1e-3)	Adam(1e-4)	Adam(1e-4)
Loss Function	BCE	BCEWithLogits	BCE	BCE	BCEWithLogits
# Encoder Stages	5	5	4	4	4
# Decoder Stages	4	4	4	4	4
Conv Block	(Conv+BN+ReLU)×2	(Conv+BN+ReLU)×2	(Conv+BN+ReLU)×2	(Conv+BN+ReLU)×2	(Conv+BN+ReLU)×2
Bottleneck	Down (512→512)	Down (512→512)	512→1024 ch	512→512 ch	512→512 ch
Final Layer	1×1 Conv+Sig	Conv2d (64→1)	Conv2d(64→1)	Conv2d(64→1)	Conv2d (64→1)
Encoder Channels	64,128,256,512,512	64,128,256,512,512	64,128,256,512	64,128,256,512	64,128,256,512
Decoder Channels	512,256,128,64	256,128,64,64	512,256,128,64	512,256,128,64	512,256,128,64
Total Parameters	13.39 M	13.39 M	31.04 M	13.39 M	13.39 M
Training Time	0:48:22	0:48:27	0:20:27	0:12:21	0:13:28
Errors	0	0	0	1 ValueError	0

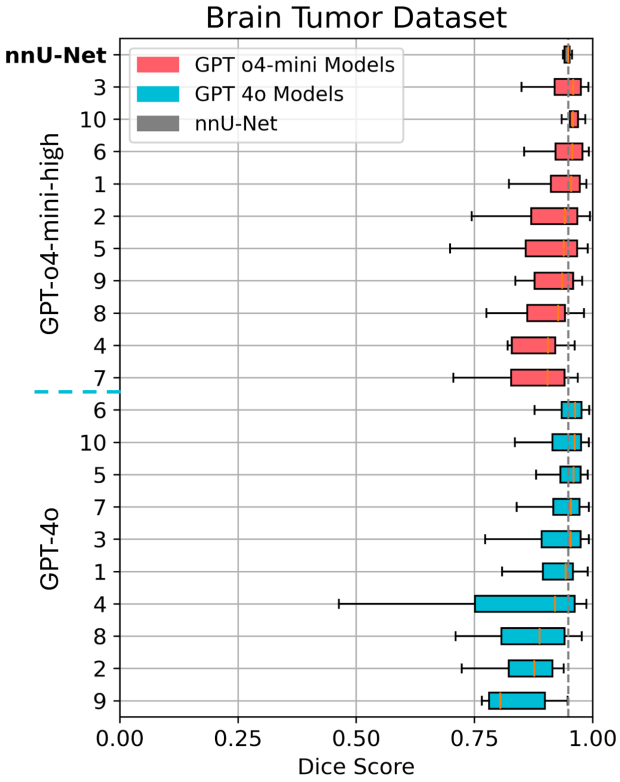


Figure 9: Run-to-run Model variability testing across 10 independent runs for one reasoning model (GPT o4-mini-high) (red) and one non-reasoning model (GPT-4o) (blue), for one examplary dataset, showing test Dice scores across runs (sorted by medians), with nnU-Net as the baseline (gray). The gray dashed vertical line marks the median test Dice score of nnU-Net for comparison.

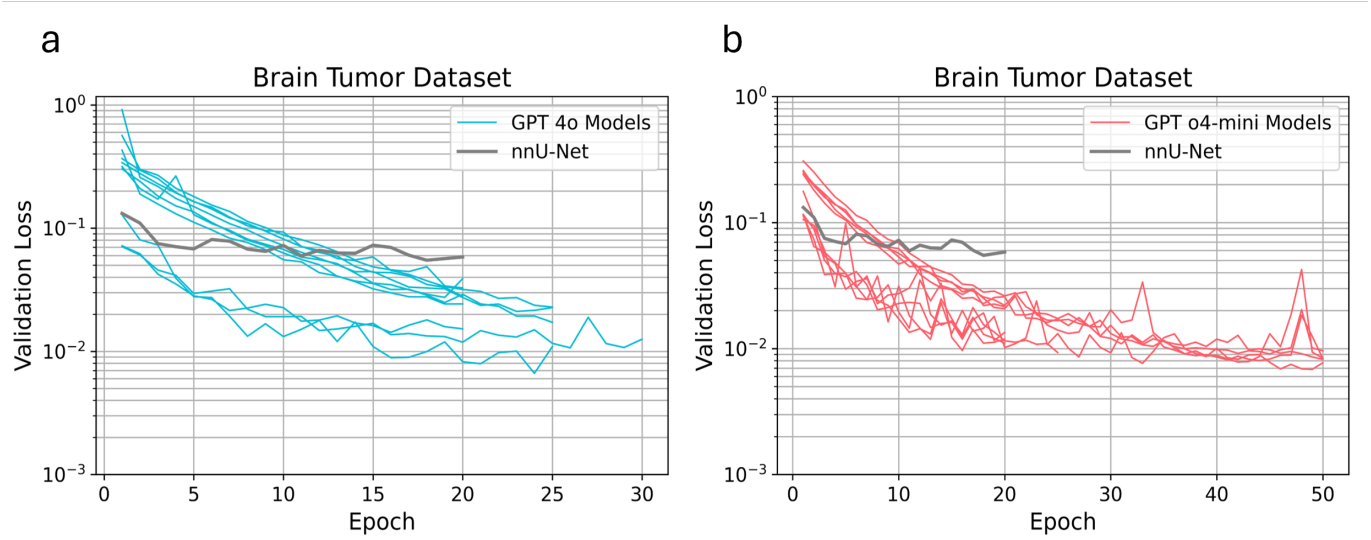


Figure 10: Run-to-run Model variability testing across 10 independent runs for one non-reasoning model (GPT-4o) (blue), and one reasoning model (GPT o4-mini-high) (red), for one examplary dataset, showing validation losses per epoch for the non-reasoning model runs (a) and the reasoning model runs (b), with nnU-Net baseline (total loss shown) in gray (note that validation splits might differ slightly).

significant difference between GPT-4o and GPT-o4-mini-high on segmentation accuracy, comparing the mean Dice scores per batch over the 10 runs ($U = 5321$, $p = 0.596$). In contrast, their mean validation losses per epoch over the 10 runs displayed a significant reduction ($U = 1087$, $p = 8.3 \times 10^{-4}$) in GPT o4-mini-high, indicating that the chain-of-thought enabled model converged more effectively despite similar final Dice performance.

Together, these findings demonstrate that although GPT-4o's code generation is mechanically more uniform as a non-reasoning model, chain-of-thought reasoning in GPT-o4-mini-high yields markedly better reproducibility and peak accuracy under repeated, end-to-end generation, underscoring the value of reasoning guidance for robust, automatic segmentation pipeline synthesis.

5. Discussion and Conclusion

To the best of our knowledge, this study provides the first comprehensive and systematic evaluation of LLMs for autonomously generating U-Net–based medical image segmentation pipelines across multiple diverse datasets and modalities, set out to address a central research question: Can large language models autonomously generate robust and generalizable U-Net–based segmentation pipelines for diverse medical imaging tasks with minimal human intervention? By systematically evaluating both reasoning-enabled and standard LLMs across six heterogeneous datasets, including videofluoroscopic and endoscopic frames, dermoscopic photographs, and volumetric MRI rendered as 2D slices. We also considered both CoT versus standard LLMs and tracking improvements from the 2024 to 2025 model generations. Our results demonstrate that reasoning-augmented LLMs, particularly Claude 4 Sonnet, GPT o4-mini-High and GPT o3, followed by Mistral Medium 3, Qwen 3 and Grok 3, achieve faster convergence, lower validation losses, and higher Dice scores than both their non-reasoning peers and earlier 2024 models. These gains were most pronounced on complex tasks involving fine structures (e.g., retinal vessels) and volumetric data rendered as 2D slices (e.g., Uterine Myoma), where reasoning models consistently outperformed lighter, non-reasoning architectures.

Interestingly, while reasoning models often produced architectures and hyperparameters well-suited for the task, their advantage was not universal. On more complex datasets such as retinal vessel segmentation and Uterine Myoma MRI slices, performance varied widely, suggesting that certain modality-specific challenges remain difficult to capture even for state-of-the-art LLMs. Moreover, occasional instability in models like Grok 3 Mini and DeepSeek V3, despite promising validation curves, underscores that validation metrics alone may not guarantee generalization during inference, particularly on the edge cases or complex ones.

An additional insight emerged from our run-to-run variability experiments on two representative models (reasoning vs non-reasoning) from the GPT family. GPT-4o, while architecturally consistent across runs, showed greater variability in final performance than GPT o4-mini-high. The latter often produced more diverse architectures, including deeper or wider networks, but yielded more stable results. This highlights a critical distinction that consistency in code generation does not necessarily ensure consistency in ultimate segmentation outcomes. Rather, reasoning-enabled models appear to exhibit better inductive priors, leading to more robust performance even when architectural decisions vary.

These findings suggest that reasoning-enabled LLMs are approaching the level of expert-designed segmentation pipelines and could serve as valuable tools for rapidly generating baseline segmentation frameworks. However, fully replacing domain-specific solutions, such as problem-tailored nnU-Net, still requires addressing gaps in data augmentation, hyperparameter tuning, preprocessing consistency, and postprocessing strategies within LLM-generated scripts.

We acknowledge limitations in our study, such as focusing exclusively on U-Net–style architectures and 2D segmentation tasks, omitting more sophisticated or task-optimized architectures (e.g., transformers, attention modules, or 3D CNNs). Furthermore, for this study, we focused solely on the out-of-the-box capabilities of commercial and open-source LLMs, without any augmentation, fine-tuning, or adaptation to the medical domain, which is, however, typical of end-user applications. The inherent stochasticity of LLM outputs, progressive refinements and improved performance over time only allow snapshots in time (see our 2024 vs. 2025 comparison).
Also, since each LLM autonomously set its own number of training epochs, comparative results inherently reflect both architectural and reasoning differences, as well as different training durations, which consequently is an uncontrolled variable that must be considered when interpreting performance gaps. Our controlled 120-epoch experiment on BAGLS confirms, however, that the superior convergence and final performance of reasoning-enabled models persists under uniform budgets, reinforcing the validity of our original findings. Achieving reproducible pipeline generation requires careful control of LLM parameters (e.g., lowering the temperature), combined with careful and iteratively refined prompt engineering and refinement to stabilize model behavior.

Future work should focus on extending these approaches by combining LLM-generated baselines with automated refinement pipelines, including hyperparameter optimization and domain adaptation routines, and exploring hybrid workflows that combine the explainability and reasoning

strengths of LLMs. Additional evaluation metrics (e.g., Hausdorff distance or volumetric IoU) could also be used to capture complementary aspects of boundary accuracy and volumetric consistency. As multimodal LLMs with stronger reasoning capabilities emerge, there is clear potential for these systems to not only design but also validate and iteratively improve medical image analysis workflows with minimal human oversight. Especially, one can think of an agentic framework, where results are fed back into an LLM-based conversation about that particular dataset or problem at hand to ensure very strong baseline performance.

In conclusion, LLMs and, particularly, reasoning-enabled models, represent a significant step toward automating medical image segmentation workflows, reducing the programming burden, and accelerating baseline development. As LLM architectures continue to evolve and integrate multimodal capabilities, and as the field of medical AI moves toward more transparent, modular, and agentic systems; LLMs are likely to become integral components of the future segmentation development pipeline, and we anticipate that these models will play an increasingly central role in clinical research and practice, serving as agile assistants for prototype development, reproducible baselines, and rapid experimentation.

Acknowledgments

This work was partly supported by the Bavarian State Ministry of Health, Care and Prevention (StMGP) under the project EndoKI and the German Research Foundation DFG (KI 2630/8-1).

Ethical Standards

The work follows appropriate ethical standards in conducting research and writing the manuscript, following all applicable laws and regulations regarding treatment of animals or human subjects.

Conflicts of Interest

We declare we do not have conflicts of interest.

Data availability

Five of the six datasets used in this study are publicly available: BAGLS, Brain Meningioma MRI, HAM10000 ISIC Skin Cancer, UMD Uterine Myoma MRI, and the combined retinal vessel segmentation dataset (DRIVE, HRF, CHASE_DB1, STARE), and can be accessed through the given links in the references. The Bolus Swallowing dataset is an in-house medical dataset containing videofluoroscopic studies, where personal patient information cannot be fully anonymized; therefore, it is not publicly accessible.

References

Poe AI. `https://https://poe.com/`. Accessed July, 2025.

AbaLab. Qwen 3_235B. `https://qwenlm.github.io/blog/qwen3/`. Accessed July 11, 2025.

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Atika Akter, Nazeela Nosheen, Sabbir Ahmed, Mariom Hossain, Mohammad Abu Yousuf, Mohammad Ali Abdullah Almoyad, Khondokar Fida Hasan, and Mohammad Ali Moni. Robust clinical applicable CNN and U-Net based algorithm for MRI classification and segmentation for brain tumor. *Expert Systems with Applications*, 238: 122347, 2024.

Anthropic. Claude 3.5 Sonnet. `https://www.anthropic.com/news/claude-3-5-sonnet`, a. Accessed July 11, 2025.

Anthropic. Claude 4 Sonnet. `https://www.anthropic.com/claude/sonnet`, b. Accessed July 11, 2025.

Jasmin Arjomandi, Luisa Neubig, and Andreas M. Kist. LLM-driven baselines for medical image segmentation: A systematic analysis. In *Proceedings of the BVM Workshop*, pages 50–56. Springer, 2025.

Fan Bai, Yuxin Du, Tiejun Huang, Max Q.-H. Meng, and Bo Zhao. M3D: Advancing 3D medical image analysis with multi-modal large language models. *arXiv preprint arXiv:2404.00578*, 2024.

Wentian Cai, Yijiang Li, Yandan Chen, Jing Lin, Zihao Huang, Ping Gao, Thippa Reddy Gadekallu, Wei Wang, and Ying Gao. Enhancing weakly supervised semantic segmentation with multi-label contrastive learning and LLM features guidance. *IEEE Journal of Biomedical and Health Informatics*, 2024.

Junying Chen, Chi Gui, Ruyi Ouyang, Anningzhe Gao, Shunian Chen, Guiming Hardy Chen, Xidong Wang, Zhenyang Cai, Ke Ji, Xiang Wan, et al. Towards injecting medical visual knowledge into multimodal LLMs

at scale. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 7346–7370, 2024a.

LinYuan Chen, Xingjian Han, Siyuan Lin, Huafeng Mai, and Huaijin Ran. TriMedLM: Advancing three-dimensional medical image analysis with multi-modal LLM. In *2024 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 4505–4512. IEEE, 2024b.

DRIVE Grand Challenge. Digital retinal images for vessel extraction (DRIVE). `https://drive.grand-challenge.org/DRIVE/`. Accessed July 11, 2025.

Louie Giray. Prompt engineering with ChatGPT: A guide for academic writers. *Annals of Biomedical Engineering*, 51(12):2629–2633, 2023.

GitHub. GitHub Copilot. `https://github.blog/news-insights/product-news/github-copilot-meet-the-new-coding-agent/`. Accessed July 11, 2025.

Pablo Gómez, Andreas M. Kist, Patrick Schlegel, David A. Berry, Dinesh K. Chhetri, Stephan Dürr, Matthias Echternach, Aaron M. Johnson, Stefan Kniesburges, Melda Kunduk, et al. BAGLS, a multihospital benchmark for automatic glottis segmentation. *Scientific Data*, 7(1): 186, 2020.

Google. Gemini 1.5 Pro. `https://blog.google/technology/ai/google-gemini-next-generation-model-february-2024/`, a. Accessed July 11, 2025.

Google. Gemini 2.5 Pro. `https://deepmind.google/models/gemini/pro/`, b. Accessed July 11, 2025.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

A. D. Hoover, Valentina Kouznetsova, and Michael Goldbaum. Locating blood vessels in retinal images by piecewise threshold probing of a matched filter response. *IEEE Transactions on Medical Imaging*, 19(3):203–210, 2000.

Jie Huang and Kevin Chen-Chuan Chang. Towards reasoning in large language models: A survey. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 1049–1065, 2023.

Yu Huang, Zelin Peng, Yichen Zhao, Piao Yang, Xiaokang Yang, and Wei Shen. MedSeg-R: Reasoning segmentation in medical images with multimodal large language models. *arXiv preprint arXiv:2506.10465*, 2025.

Aaron Hurst, Adam Lerer, Adam P. Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, A. J. Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. GPT-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

International Skin Imaging Collaboration. International skin imaging collaboration (ISIC) archive. `https://www.isic-archive.com/`. Accessed July 11, 2025.

Fabian Isensee, Paul F. Jaeger, Simon A. A. Kohl, Jens Petersen, and Klaus H. Maier-Hein. nnU-Net: A self-configuring method for deep learning-based biomedical image segmentation. *Nature Methods*, 18(2):203–211, 2021.

Gurucharan Marthi Krishna Kumar, Aman Chadha, Janine D. Mendola, and Amir Shmuel. MedVisionLlaMA: Leveraging pre-trained large language model layers to enhance medical image segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1114–1124, 2025.

Xin Lai, Zhuotao Tian, Yukang Chen, Yanwei Li, Yuhui Yuan, Shu Liu, and Jiaya Jia. LISA: Reasoning segmentation via large language model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9579–9589, 2024.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

Zhengliang Liu, Aoxiao Zhong, Yiwei Li, Longtao Yang, Chao Ju, Zihao Wu, Chong Ma, Peng Shu, Cheng Chen, Sekeun Kim, et al. Tailoring large language models to radiology: A preliminary approach to LLM adaptation for a highly specialized domain. In *International Workshop on Machine Learning in Medical Imaging*, pages 464–473. Springer, 2023.

Jun Ma, Yuting He, Feifei Li, Lin Han, Chenyu You, and Bo Wang. Segment anything in medical images. *Nature Communications*, 15(1):654, 2024.

Sara Vera Marjanović, Arkil Patel, Vaibhav Adlakha, Milad Aghajohari, Parishad BehnamGhader, Mehar Bhatia, Aditi Khandelwal, Austin Kraft, Benno Krojer, Xing Han Lù, et al. DeepSeek-R1 thoughtology: Let's think about LLM reasoning. *arXiv preprint arXiv:2504.07128*, 2025.

Meta AI. LLaMA 3.1 405B. https://ai.meta.com/blog/meta-llama-3-1/, a. Accessed July 11, 2025.

Meta AI. LLaMA 4 Maverick. https://ai.meta.com/blog/llama-4-multimodal-intelligence/, b. Accessed July 11, 2025.

Microsoft. Bing Microsoft Copilot. https://www.microsoft.com/en-us/microsoft-copilot/blog/product/bing/. Accessed July 11, 2025.

Mistral AI. Mistral Medium 3. https://mistral.ai/news/mistral-medium-3. Accessed July 11, 2025.

Luisa Neubig, René Groh, Melda Kunduk, Deirdre Larsen, Rebecca Leonard, and Andreas M. Kist. Efficient patient orientation detection in videofluoroscopy swallowing studies. In *Bildverarbeitung für die Medizin 2022*, pages 129–134, Wiesbaden, 2022. Springer Fachmedien Wiesbaden. ISBN 978-3-658-36932-3.

Luisa Neubig, Deirdre Larsen, Melda Kunduk, and Andreas M. Kist. Unstructured electronic health records of dysphagic patients analyzed by large language models. *IEEE Journal of Translational Engineering in Health and Medicine*, 2025.

Jan Odstrcilik, Radim Kolar, Attila Budai, Joachim Hornegger, Jiri Jan, Jiri Gazarek, Tomas Kubena, Pavel Cernosek, Ondrej Svoboda, and Elli Angelopoulou. Retinal vessel segmentation by improved matched filtering: Evaluation on a new high-resolution fundus image database. *IET Image Processing*, 7(4):373–383, 2013.

OpenAI. GPT o3. https://openai.com/index/introducing-o3-and-o4-mini/, a. Accessed July 11, 2025.

OpenAI. GPT o1. https://openai.com/o1/, b. Accessed July 11, 2025.

Christopher G. Owen, Alicja R. Rudnicka, Robert Mullen, Sarah A. Barman, Dorothy Monekosso, Peter H. Whincup, Jeffrey Ng, and Carl Paterson. Measuring retinal vessel tortuosity in 10-year-old children: Validation of the computer-assisted image analysis of the retina (CAIAR) program. *Investigative Ophthalmology & Visual Science*, 50(5):2004–2010, 2009.

Haixia Pan, Minghuang Chen, Wenpei Bai, Bin Li, Xiaoran Zhao, Meng Zhang, Dongdong Zhang, Yanan Li, Hongqiang Wang, Haotian Geng, et al. Large-scale uterine myoma MRI dataset covering all FIGO types with pixel-level annotations. *Scientific Data*, 11(1):410, 2024.

Aske Plaat, Annie Wong, Suzan Verberne, Joost Broekens, Niki van Stein, and Thomas Back. Reasoning with large language models: A survey. *arXiv preprint arXiv:2407.11511*, 2024.

Pradosh123. Retinal vessel segmentation combined dataset. https://www.kaggle.com/datasets/pradosh123/retinal-vessel-segmentation-combined. Accessed July 11, 2025.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015: 18th International Conference, Munich, Germany, October 5–9, 2015, Proceedings, Part III 18*, pages 234–241. Springer, 2015.

Wenfang Sun, Yingjun Du, Gaowen Liu, Ramana Kompella, and Cees G. M. Snoek. Training-free semantic segmentation via LLM-supervision. *arXiv preprint arXiv:2404.00701*, 2024.

Philipp Tschandl, Cliff Rosendahl, and Harald Kittler. The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions. *Scientific Data*, 5(1):1–9, 2018.

Junchi Wang and Lei Ke. LLM-Seg: Bridging image segmentation and large language model reasoning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1765–1774, 2024.

Peng Wang, Wenpeng Lu, Chunlin Lu, Ruoxi Zhou, Min Li, and Libo Qin. Large language model for medical images: A survey of taxonomy, systematic review, and future trends. *Big Data Mining and Analytics*, 8(2):496–517, 2025.

Sheng Wang, Zihao Zhao, Xi Ouyang, Tianming Liu, Qian Wang, and Dinggang Shen. Interactive computer-aided diagnosis on medical image using large language models. *Communications Engineering*, 3(1):133, 2024.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V. Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837, 2022.

xAI. Grok 3. https://docs.x.ai/docs/models. Accessed July 11, 2025.

Zongwei Zhou, Md Mahfuzur Rahman Siddiquee, Nima Tajbakhsh, and Jianming Liang. UNet++: A nested U-Net architecture for medical image segmentation. In *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support: 4th International Workshop, DLMIA 2018, and 8th International Workshop, ML-CDS, Held in Conjunction with MICCAI 2018, Granada, Spain, September 20, 2018, Proceedings*, pages 3–11. Springer, 2018.

Appendix A. 2024 and 2025 individual Models Validation Losses

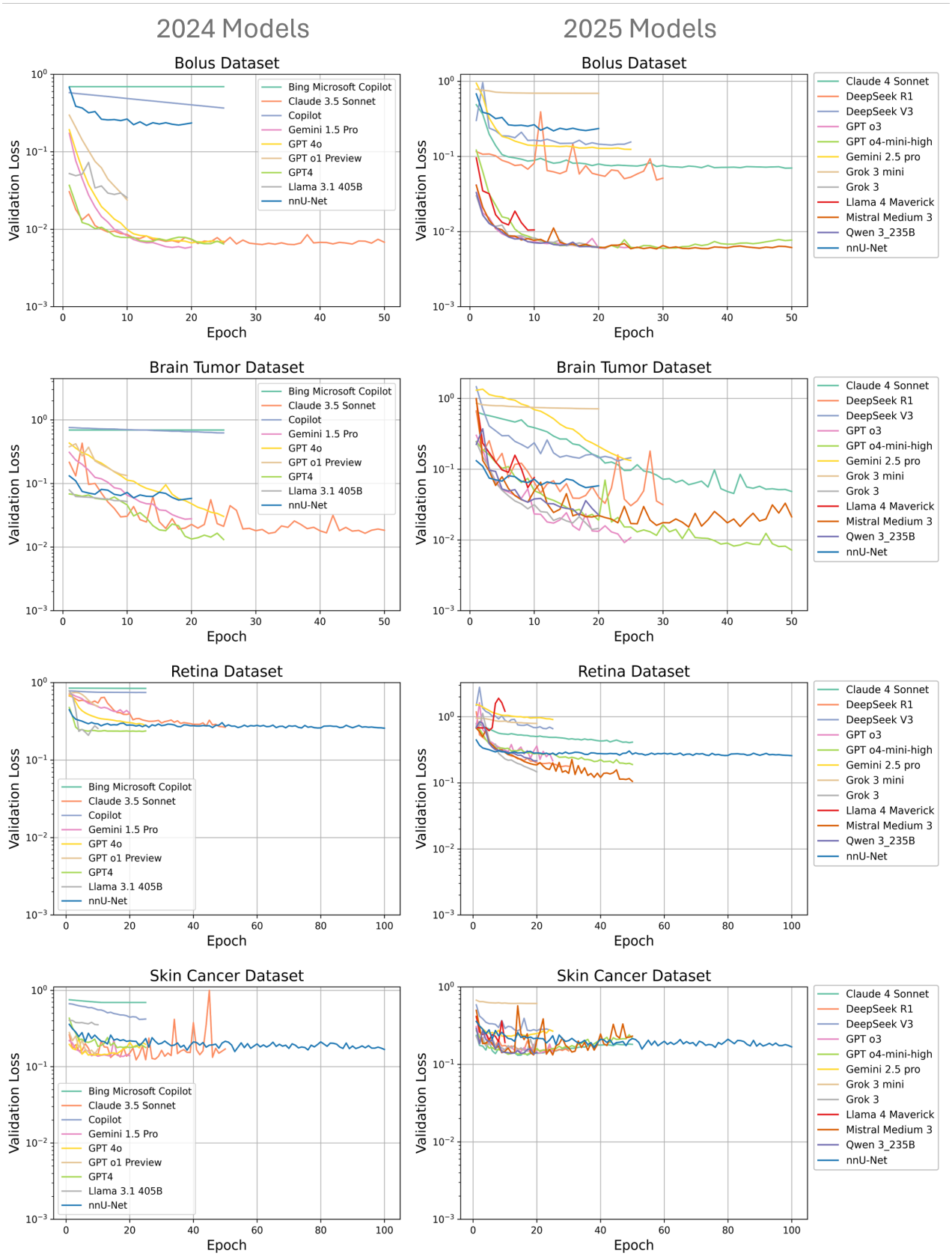


Figure 11: Validation loss per epoch for 2024 (left) and 2025 (right) LLM-generated segmentation pipelines on 4 of the datasets, with nnU-Net v2 (total loss shown) as a baseline (blue). Number of epochs were determined by each LLM. Note that curves show loss values from different formulations and absolute values are not directly comparable.

Appendix B. 2024 vs 2025 Models Validation Losses on each Dataset (simplified visualization)

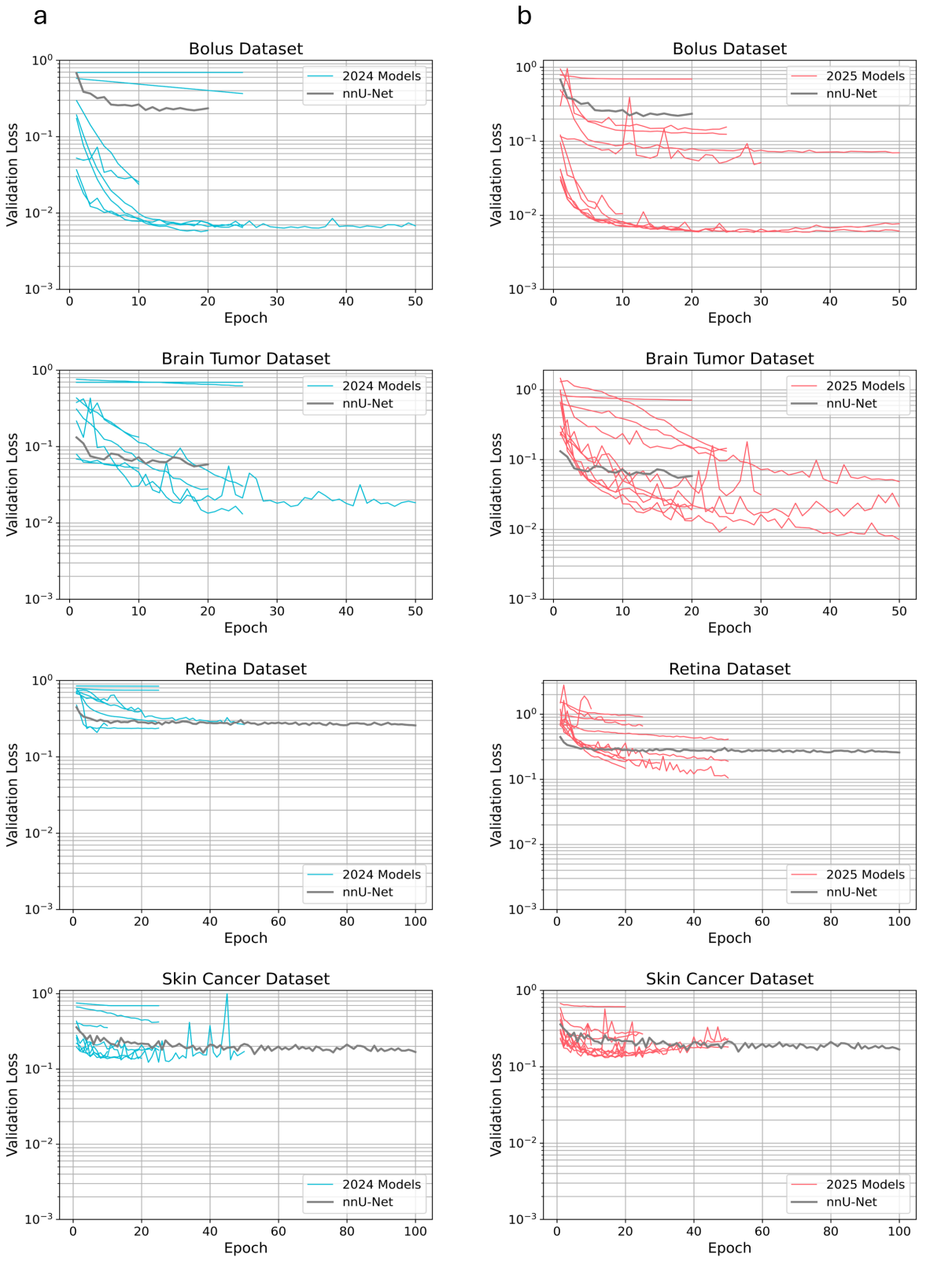


Figure 12: Validation loss per epoch for 2024 (blue) (a) and 2025 (red) (b) LLM-generated segmentation pipelines, shown on 4 of the datasets, with nnU-Net v2 (total loss shown) as a baseline (gray). Number of epochs were determined by each LLM. Note that curves show loss values from different formulations and absolute values are not directly comparable.

Appendix C. 2024 vs 2025 Models Dice Scores on each Dataset (sorted by median values for each dataset)

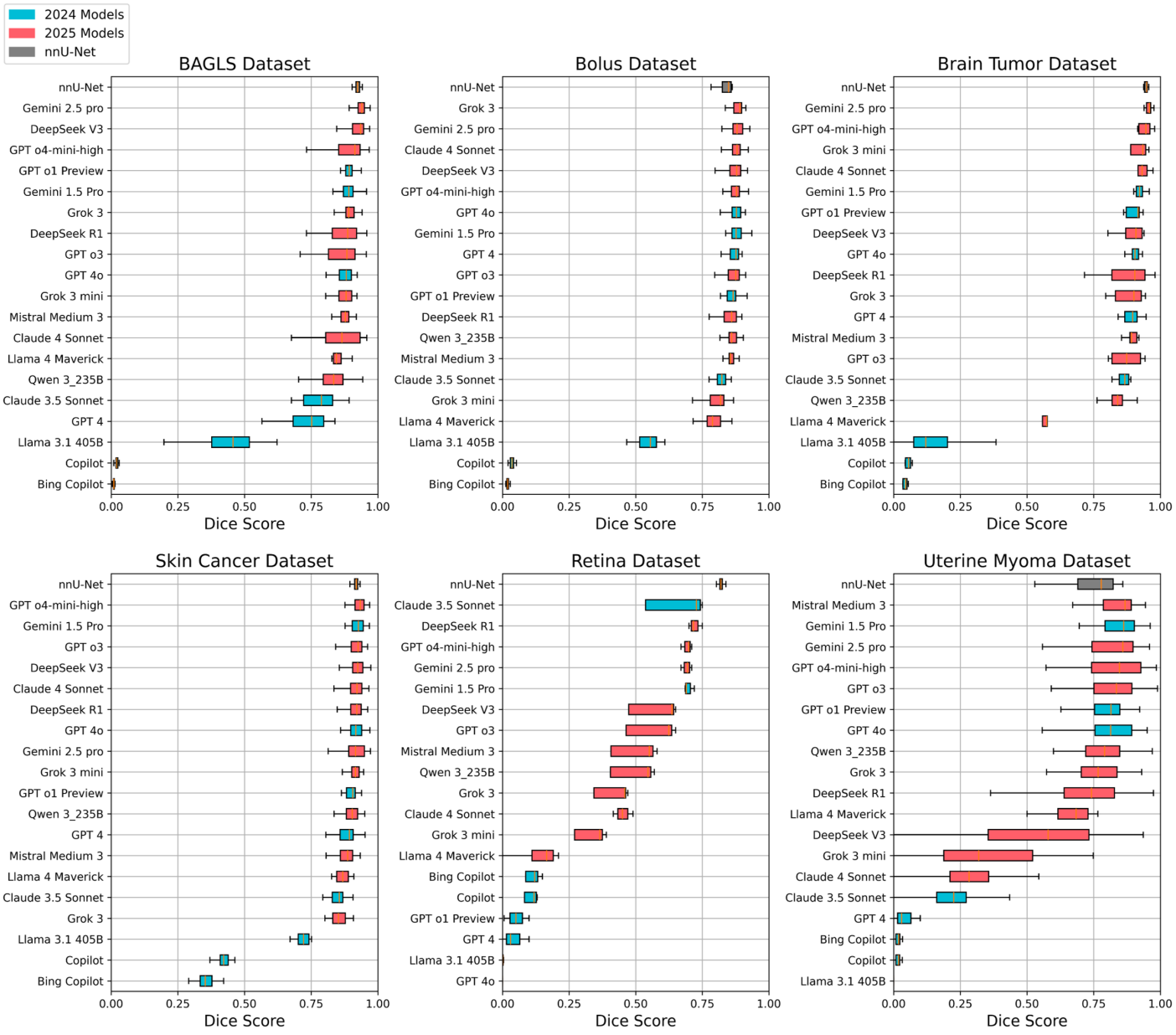


Figure 13: Comparison of the test Dice scores across all selected 2024 (blue) and 2025 (red) models, for each of the six datasets. Dice scores are sorted by median from high to low. nnU-Net is shown at the top as the baseline.

Appendix D. Prompt Template

LLM input prompt text given below. Instructions in brackets can be customized or tailored to needs:

```
in separate scrips below, provide a complete ready-to-run Python implementation
for a U-Net architecture using PyTorch, for binary segmentation of [grayscale] images.
The dataset should consist of [grayscale] images and corresponding [binary] masks,
which are located in 2 different folders, [OR in the same path (the masks have the
same name as the images, or with a suffix, like: "number.png" and the masks
"number_seg.png"], around [5000] samples in total. access and split directly using
train_test_split on the index list and Subset, [and do NOT duplicate or copy to new  directory)]. It
should work and provide correct predictions, and save the losses and
dice scores as instructed below. The code should be structured into importable
scripts with the following:

1. A custom Dataset class for loading the [grayscale] images and corresponding [binary] masks.
(only the [png] files, ignore meta files or additional files in the directory)
2. A UNet model class definition using PyTorch, optimized for [binary] segmentation
[(with small foreground, mostly background)]
3. Functions for the training procedure, validation procedure, and Testing procedure.

Considerations for each part:
4. Ensure needed transforms on input image and mask pairs [(Re-size images
   correctly, ensure gray scale and binary...)]
5. Avoid overfitting and data leakage (without using augmentations), choose
   appropriate hyperparameters, learning rate, batch size, etc

• For Training and Validation:
6. Compute training and validation losses.
7. Track and save the average training loss for each epoch in a separate Excel file (train_losses.xlsx)
in a given save path. The first row should contain the epoch numbers, and the second row should contain
the corresponding average training loss for each epoch.
8. Similarly, save the validation loss for each epoch in an Excel file (val_losses.xlsx) with the same
structure.
9. At the end of training, save both the entire model and the model's state dictionary (.pth files)
in the save path
10. Calculate total training time from start to end, and print at the end of training
11. The progress, phase (train/validation), and all info should be shown epoch by epoch using 'tqdm'
progress bars during training, validation, and testing.
12. A function to visualize training losses and validation losses over epochs afterwards, on the same
plot. Save plot in a specified save path as png.
13. During Validation: mean Dice scores should be calculated over each batch, for each epoch, and
stored in one Excel file (epochs as rows and batches as columns). The Excel file
(named \validation_dice_scores") should be saved at the specified save path after validation is done.
14. Do the same for the Testing and store dice the Dice scores the same way and save the excel file
as \test_dice_scores" after testing is done.

• For Testing:
15. A function to visualize the predicted masks, the input image, and the ground truth, for 5 random
samples, after testing is done. (subplots for 5 random samples in one plot, in 5 rows and 3 columns
(title at top: Input Image, Ground Truth, Prediction). Also display image file name string above each
image and mask. save the plot as '.png').

• For Main:
16. Shuffle the data, split the dataset into: 80% training, 10% validation, and 10% test sets using
'train_test_split'. Print the sample sizes of each, and copy the splitted data (images and their
respective masks)
17. DataLoader setup for training, validation, and testing sets.
18. Ensure correct data loading, print data loader sizes.
```

```
19. Also usig torchinfo, print model summary, and total number of model parameters (n Trainable params)

20. All the above functionalities should be implemented in a modular way so that they can be imported
as scripts or classes.
21. All visualizations should just be saved as png. No need to display the plots during running.
22. Ensure that everything can be executed in an ‘if __name__ == "__main__":‘ block with instructions
for loading the dataset, configuring hyperparameters, and saving the trained model.

The directory structure should only be:
unet_segmentation/
|-- dataset.py
|-- model.py
|-- train.py
|-- main.py
```