Journal of Computing and Electronic Information Management
ISSN: 2413-1660 | Vol. 20, No. 3, 2026

Accelerating Deep Learning Inference for Brain Tumor Segmentation: A Review of Architectures, Frameworks, and Clinical Translation

Junyi He

School of Computer Science and Technology, Henan Polytechnic University, Jiaozuo 454000, China

Abstract: Brain tumor segmentation has transitioned from an accuracy-dominant research agenda to a constrained multi-objective optimization problem in which segmentation quality, latency, memory footprint, robustness to missing modalities, and deployment reproducibility must be optimized jointly rather than sequentially. Focusing on developments published from 2023 to 2026, this review analyzes inference acceleration from a system perspective that links architectural evolution, model-compression and compilation strategies, and hardware-aware deployment constraints within a single causal framework. Specifically, we synthesize evidence on the shift from CNN/nnU-Net baselines to Transformer hybrids and linear-time State Space Model families, evaluate the practical effects of mixed precision, quantization, pruning, distillation, and compiled runtimes, and examine how modality-missing robustness interacts with graph stability and therefore with real-world compiler efficiency. On the basis of this synthesis, we formulate a reproducible evaluation protocol for acceleration claims that reduces cross-paper comparability errors and makes reported latency evidence clinically interpretable. We conclude with a forward-looking engineering roadmap indicating how the field can move from benchmark-centric speed demonstrations to reliable real-time segmentation systems suitable for heterogeneous hospital infrastructure.

Keywords: Brain umor segmentation; Inference efficiency; TensorRT; TVM; PyTorch 2.0.

1. Introduction

Inference acceleration has become an increasingly critical engineering dimension in neural network-based brain tumor segmentation, since deployable systems must jointly satisfy constraints on latency, memory footprint, modality robustness, and reporting reproducibility rather than optimizing segmentation quality alone [1, 3, 4, 5, 38]. Addressing this dimension requires a system-level analysis that links architectural design, compression and compilation strategies, and hardware-aware evaluation within a unified framework [6, 7, 8, 9]. The present analysis is accordingly structured around four analytical axes. The first axis concerns the architectural and engineering changes that most substantially altered achievable inference speed between 2023 and 2026. The second concerns the extent to which reported efficiency gains originate from model-level design decisions versus deployment-stack engineering, together with the hardware and software conditions under which these contributions interact. The third formalizes the criteria by which acceleration claims must be measured and reported in order for cross-paper comparisons to remain valid and clinically interpretable, given the documented sensitivity of latency figures to hardware configuration, precision mode, and input protocol [10, 11, 12]. The fourth characterizes the conditions under which inference efficiency can be maintained when one or more imaging modalities are absent—a scenario that arises routinely in multi-center clinical workflows—and links this robustness requirement to the graph-stability constraints of production compilers [13, 14, 15, 16, 17].

The scope of the analysis is therefore centered on inference-time acceleration and deployability, treating Dice coefficient as a necessary quality-retention constraint rather than the primary optimization target [9, 18, 19].

To ensure that this review remains systematically update-friendly and internally reproducible, we define a structured retrieval and inclusion protocol. Literature was drawn from IEEE Xplore, PubMed, arXiv, MICCAI proceedings, Medical Image Analysis, Image and Vision Computing, and IEEE Transactions on Medical Imaging, with a time window spanning January 2023 to the current update date. A paper was included if it reported results on brain tumor MRI segmentation together with at least one efficiency-relevant metric, including inference latency, floating-point operation count (FLOPs), peak memory consumption, throughput, energy draw, or model parameter count; studies reporting segmentation accuracy alone, without any deployment-relevant efficiency evidence, were excluded. For each included paper, we annotate the following metadata: publication status (journal, conference, or preprint), the hardware platform used for evaluation, the numerical precision mode, batch size, input volume dimensions and patching strategy, and the runtime framework or compilation backend. This structured metadata collection is specifically designed to make cross-paper speed comparisons valid and to prevent the propagation of non-comparable latency claims that have been observed across prior acceleration literature.

2. Backbone Evolution Under Efficiency Pressure

2.1. CNN and nnU-Net Family: Strong Baselines, Limited Global Context

CNN encoder-decoder architectures rooted in U-Net and attention-augmented U-Net variants [20, 21], particularly the self-configuring nnU-Net family [9, 22, 23, 24], continue to serve as robust and practically preferred deployment baselines owing to their mature kernel implementations, well-characterized training dynamics, and broad compatibility with production-grade inference frameworks. Their principal

architectural limitation lies in the efficiency of global-context modeling over large three-dimensional volumes, where the effective receptive field is constrained by the depth of the convolutional stack rather than by explicit long-range sequence dependency mechanisms. This limitation does not, however, preclude clinical relevance: when CNNs are combined with precision reduction to INT8 or BF16, operator fusion through compiled backends, and structured pruning targeting supported sparsity patterns, they can satisfy stringent latency targets in conservative clinical IT environments where hardware diversity, regulatory audit requirements, and institutional change-management constraints favor simpler and more interpretable deployment graphs over architecturally novel alternatives [25, 26, 27, 28, 29].

2.2. Transformer Hybrids: Better Global Modeling, Higher Compute Burden

Transformer-based segmentation architectures advanced boundary delineation and long-range contextual modeling [30, 31, 32, 33, 34], addressing a recognized limitation of purely convolutional designs whose receptive fields are bounded by kernel stack depth. However, the full self-attention mechanism introduces quadratic computational and memory complexity with respect to token count—a penalty that becomes particularly severe in three-dimensional MRI volumes where the spatial token sequence is orders of magnitude longer than in natural image benchmarks [9, 30, 35]. Unless token reduction strategies such as windowed attention, deformable sampling, or hybrid convolution-attention blocks are incorporated into the architecture, this complexity translates directly into GPU memory pressure and inference latency that post-training optimization alone cannot sufficiently recover. Consequently, the practical gains of Transformer architectures in segmentation quality are genuine but require careful co-design between architectural choices and the targeted runtime stack to avoid capacity bottlenecks that would disqualify these models from latency-constrained clinical deployment scenarios [7, 8, 16, 36].

2.3. SSM/Mamba and Linear-Time Trends: Main Acceleration Frontier

State Space Model (SSM)-based architectures, most notably the Mamba family and its medical imaging adaptations [37, 38, 39, 40], emerged as the central efficiency frontier from 2024 to 2026, precisely because they decouple global-context propagation from the quadratic cost of self-attention through hardware-aware selective state scanning with linear time complexity in sequence length. Recent medical adaptations, including VM-UNet variants and cascade Mamba-convolution hybrids [16, 39, 40], report segmentation quality competitive with comparably parameterized Transformer architectures while simultaneously achieving substantial reductions in peak GPU memory and aggregate FLOPs. More recent directions such as gated differential linear attention [41] extend this paradigm further, achieving high-fidelity segmentation decoder behavior with explicit linear-time complexity guarantees that are more amenable to formal deployment-time latency analysis. The most significant architectural shift of the 2024-2026 period is therefore the movement away from full global attention toward linear-time global modeling, often realized through hybrid scan-convolution blocks that preserve local spatial inductive biases while enabling efficient long-range dependency capture at clinically relevant scales [28, 38, 41, 42].

2.4. Missing-Modality-Robust Architectures as Acceleration Enablers

In realistic multi-center clinical workflows, the simultaneous availability of all four standard BraTS modalities-T1, T1c, T2, and FLAIR-cannot be assumed, making modality-robust architectures a prerequisite for deployable systems rather than an optional design consideration [13, 14, 15, 16, 17, 43]. Region-aware fusion networks such as RFNet [14] and more recent adaptive multi-granular designs exemplified by AMGFormer [13] address this challenge by constructing fusion pathways that degrade gracefully under arbitrary modality-missing patterns, thereby reducing the catastrophic performance drops that characterize single-pathway models when expected inputs are absent [16, 17, 44]. A critical but underappreciated engineering constraint is that these robustness modules must preserve a stable computational graph topology across all missing-modality configurations: dynamic branching on input availability or input-conditional operator routing disables the kernel fusion passes on which TensorRT and TVM depend, causing realized inference speed on deployment hardware to fall substantially below what isolated benchmark profiling would suggest, and thereby undermining the latency guarantees on which clinical deployment planning is based [6, 7, 36, 45].

Taken together, the architectural evidence presented in Section 2 indicates that acceleration-relevant model choice cannot be reduced to a binary comparison between "high-accuracy" and "lightweight" designs; rather, the decisive factor is whether a backbone can preserve global context under realistic memory ceilings while maintaining a static-enough execution graph to remain compilable across modality conditions and hardware generations.

3. Inference Acceleration Stack

Building on this architectural analysis, Section 3 examines why reported latency gains in recent literature are best interpreted as stack-level emergent effects generated by interactions among model design, numerical precision policy, compiler behavior, and system-level dataflow overhead, rather than as isolated improvements attributable to any single optimization component.

3.1. Model-Level Acceleration

At the model level, quantization represents the most consistently impactful compression technique, reducing weight and activation representations from FP32 to INT8 or BF16 either through post-training quantization (PTQ) with representative calibration data or through quantization-aware training (QAT) when segmentation accuracy sensitivity is high [25, 27, 28, 46, 47]. Structured pruning and channel sparsity offer complementary latency reductions, but their effectiveness is strongly backend-dependent: only frameworks with native sparse tensor kernel support-such as NVIDIA CUTLASS-based sparse matmul extensions-can translate theoretical parameter sparsity into realized throughput gains at inference time, whereas unstructured pruning typically yields no wall-clock benefit without hardware-level sparse execution units [27, 29, 48]. Knowledge distillation and lightweight decoder designs provide a different trade-off axis, reducing inference cost by

transferring representational capacity from a larger teacher architecture to a compact student without requiring the full data volumes or training budgets of the original training procedure [17, 43, 49, 50]. Hardware-aware neural architecture search has increasingly been applied to navigate the accuracy-latency Pareto frontier jointly [51, 52, 53], enabling model families that are explicitly co-optimized for specific deployment hardware classes rather than designed under the implicit assumption of an idealized benchmark environment.

3.2. Compiler and Runtime-Level Acceleration

At the compiler and runtime level, NVIDIA TensorRT delivers the most consistently documented practical gains for medical image segmentation deployed on NVIDIA hardware pipelines [7, 36, 45], primarily through layer fusion, precision calibration workflows, and engine serialization that eliminate per-inference Python interpreter overhead and enable sustained kernel throughput at deployment batch sizes. The complete end-to-end workflow of this process, from model parsing to optimized runtime deployment, is illustrated in Figure 1. Apache TVM [6, 35] covers complementary territory by enabling custom operator scheduling, automatic cache-level memory access tuning, and backend portability to non-NVIDIA targets including ARM processors and FPGA-based clinical imaging terminals, making it the preferred tool when deployment hardware is heterogeneous across institutional sites. The PyTorch 2.0 compilation stack, built around TorchDynamo bytecode graph capture and TorchInductor kernel generation [8, 54, 55, 56], provides an operationally lower-barrier path from research code to compiled deployment by substantially reducing the model preparation and operator registration overhead that would otherwise precede compilation-derived latency improvements. Across all three backends, the dominant pattern in the reviewed literature is that clinically meaningful latency reductions almost never originate from a single intervention; rather, they emerge from the synergistic composition of model compression, mixed-precision arithmetic, and compiled execution, a stack synergy whose cumulative benefit systematically exceeds what individual ablations of each component would suggest [25, 28].

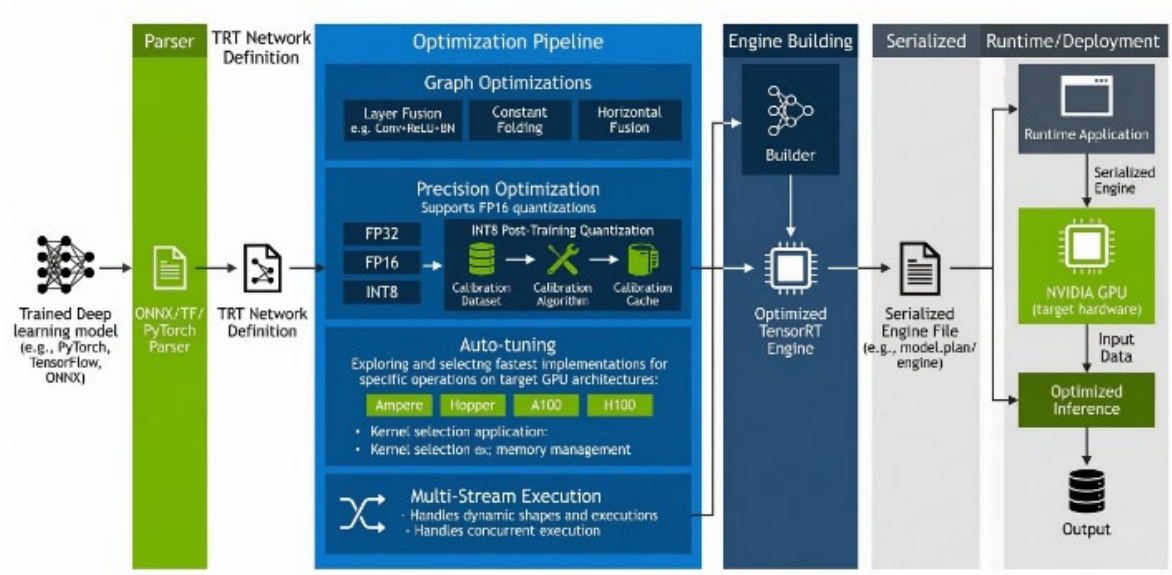

Fig. 1 NVIDIA TensorRT Optimization Pipline

3.3. Hardware and System-Level Factors

At the hardware and system level, the generation of Tensor Cores available at the deployment site is a primary determinant of achievable latency that is largely independent of algorithmic choices: the native BF16 and INT8 Tensor Core throughput of the NVIDIA A100 architecture [10, 11, 57] can produce order-of-magnitude latency differences relative to older hardware under identical model and precision configurations, meaning that benchmark results from recent literature are systematically misleading when naively applied to a different hardware generation without re-profiling. A further critical distinction must be maintained between throughput and latency as optimization objectives, since batch-maximizing throughput tuning inflates server-side performance metrics while leaving per-case inference latency largely unchanged; real-time clinical applications, by contrast, require compliance with p95 tail latency targets measured under single-volume input conditions that accurately reflect the sequential, case-by-case structure of radiological reading workflows, where batch accumulation between cases is clinically inadmissible [39, 60]. Finally, preprocessing pipelines-including skull stripping, intensity normalization across scanners, resampling to isotropic voxel spacing, and rigid co-registration-together with postprocessing steps such as connected-component labeling and coordinate space inversion to native patient space, can account for a disproportionately large fraction of total wall-clock time when the segmentation network itself has already been highly optimized; widely used medical-imaging toolchains such as MONAI and TorchIO further motivate end-to-end pipeline profiling rather than model-only benchmarking as the only valid basis for clinical deployment latency specifications [19, 58, 59, 60].

4. Methodological Guidelines and Development Trends

A latency claim reported in isolation, without a complete characterization of its measurement context, carries no scientific validity for comparative purposes and should not inform clinical procurement or deployment planning. A minimally complete acceleration report must simultaneously disclose six inter-dependent components [9, 12]. The

hardware specification must include the GPU or CPU model, available memory capacity, driver version, and CUDA alongside cuDNN library versions, since even nominally identical GPU models can differ substantially in memory bandwidth, Tensor Core density, and supported precision modes across production generations [10, 11, 57]. The runtime environment must identify the inference framework and backend version, specifying whether execution proceeded under PyTorch eager mode, torch.compile, TensorRT, or TVM, as the same model architecture can exhibit two- to four-fold latency differences across these backends due to fundamentally different kernel fusion and memory scheduling strategies [6, 7, 8, 35, 56]. The precision configuration must enumerate the numerical format applied to weights and activations-whether FP32, FP16, BF16, or INT8-together with the calibration dataset details for any quantized mode, since precision-induced latency reductions are methodologically inseparable from their associated accuracy cost [25, 27, 28, 46, 47]. The input protocol must state whether the model was evaluated on full-volume inputs or sliding-window patches, specifying patch shape, spatial overlap ratio, and inference batch size, as wall-clock time scales nonlinearly with each of these parameters in a manner that is architecture-dependent. The reported metric set must include both median (p50) and 95th-percentile (p95) latency to characterize tail behavior under realistic operating conditions, alongside explicit warmup iteration policies and an unambiguous statement of whether preprocessing execution time is included or excluded from the reported figures [9, 12]. Finally, quality retention must be quantified as the change in Dice coefficient and HD95 relative to an unaccelerated reference under the same evaluation protocol, since acceleration techniques that produce clinically significant accuracy degradation are inadmissible regardless of their speed advantage. The absence of any element in this six-component specification renders cross-paper latency comparison methodologically invalid.

Four convergent trends characterize the 2023-2026 development period. At the architecture level, the dominant movement has been from computationally heavy global self-attention toward linear-time sequential modeling through SSM-family architectures and linear-attention variants, driven by the observation that quadratic attention complexity is fundamentally incompatible with the memory and latency budgets imposed by three-dimensional clinical imaging pipelines operating at full volume resolution [28, 37, 38, 41, 42]. At the deployment level, community practice has shifted from naive eager-mode inference executed at research-time to compiled and serialized execution engines, reflecting a growing recognition that runtime optimization contributes latency reductions of comparable magnitude to those achieved through architectural innovation alone [36, 45, 56]. At the robustness level, system design assumptions have progressed from the implicit full-modality availability presumed in benchmark settings to explicit modality-aware inference, motivated by the epidemiological reality that real-world multi-center imaging datasets exhibit substantial and institutionally variable rates of missing modalities that cannot be adequately addressed by post-hoc data imputation [17, 43, 44]. At the evaluation level, benchmark reporting is incorporating end-to-end deployment metrics including p95 tail latency, preprocessing overhead, and quality retention under compression at an increasing rate, gradually displacing the prior convention of reporting model-only FLOPs or parameter counts that bear limited empirical relationship to observed wall-clock clinical performance.

5. Limitations

Several limitations of this review warrant acknowledgment. The 2025-2026 literature base includes a substantial proportion of preprints that have not yet undergone formal peer review; while we include these contributions on account of their technical significance and the pace of the field, their reported results, architectural rankings, and efficiency claims may be revised or retracted following peer evaluation, and readers should weight preprint-derived conclusions accordingly. Second, access restrictions on certain publisher platforms introduced unavoidable gaps during bibliographic verification, despite the multi-source retrieval protocol defined in Section 2; some potentially relevant conference-specific proceedings and journal supplements may therefore be underrepresented in the synthesized evidence base. Third, and most consequentially for clinical translation practice, the overwhelming majority of speed claims in the reviewed literature are hardware-specific and cannot be assumed portable across deployment environments: a latency figure reported on an A100 GPU in a high-performance computing cluster provides strictly limited predictive value for deployment on a Quadro RTX clinical workstation, an ARM-based edge imaging device, or a shared hospital server environment, and cross-institutional efficiency comparisons must therefore be treated with appropriate caution unless hardware-matched re-profiling has been explicitly performed.

6. Forward-Looking Roadmap

The most immediately actionable priority for the field is the standardization of measurement protocols and the routine publication of deployment manifests alongside released model checkpoints, enabling independent researchers and clinical engineers to reproduce and contextually compare latency figures without reconstructing evaluation infrastructure from first principles-an obstacle whose practical impact on cross-study comparability is equivalent to that caused by non-reproducible accuracy results. At the intermediate research horizon, the most impactful open direction is genuine algorithm-system co-design: jointly optimizing model architecture decisions, quantization schedules, and compiler backend kernel libraries for fixed, explicitly specified clinical hardware classes, rather than designing models for abstract benchmarking environments whose hardware profiles diverge systematically from the constraints encountered in real clinical deployment. Over the longer term, achieving real-time brain tumor segmentation as a genuinely deployed clinical capability-rather than a performance target demonstrated exclusively on curated benchmark splits-requires constructing robustness to missing modalities directly into production inference pipelines and subjecting the resulting end-to-end systems to external multi-center prospective validation, where the statistical heterogeneity of acquisition protocols, scanner manufacturers, field strengths, and institutional preprocessing conventions constitutes a substantially more demanding and ecologically valid test than the controlled retrospective split-validation paradigm that currently dominates the segmentation literature.

7. Conclusion

The latest progress in brain tumor segmentation

acceleration cannot be adequately accounted for by analyses of isolated model-level changes, because the strongest and most reproducible gains arise when linear-time global modeling, precision-aware compression, compiler/runtime optimization, and modality-robust inference design are developed as mutually constrained components of the same deployment system. Across the evidence reviewed here, the central methodological lesson is that acceleration claims become scientifically meaningful only when they are reported with complete measurement context and coupled to explicit quality-retention evidence; absent this reporting discipline, improvements in nominal throughput or FLOPs remain weak proxies for clinical utility. Consequently, the next phase of impactful research should prioritize hardware-matched reproducibility, end-to-end latency accounting, and prospective robustness validation under missing-modality conditions, thereby shifting the field from non-reproducible benchmark-centric speed claims toward verifiable real-time neuro-oncology workflows that can be trusted in heterogeneous clinical infrastructures.

References

[1] M. Martucci et al., Magnetic resonance imaging of primary adult brain tumors: State of the art and future perspectives, Biomedicines 11 (2) (2023) 364. doi: 10.3390 / biomedicines11020364.

[2] B. Jiang et al., Deep learning for brain tumor segmentation in multimodal MRI images: A review of methods and advances, Image and Vision Computing 156 (2025). doi:10.1016/j.imavis.2025.105463.

[3] H. Xue et al., Multi-modal tumor segmentation methods based on deep learning: a narrative review, Quantitative Imaging in Medicine and Surgery 14 (1) (2024) 1122-1140. doi:10.21037/qims-23-818.

[4] F. J. Dorfner et al., A review of deep learning for brain tumor analysis in MRI, npj Precision Oncology 9 (1) (2025). doi:10.1038/s41698-024-00789-2.

[5] N. Netshamutshedzi et al., A systematic review of hybrid machine learning models for brain tumour segmentation and detection, Frontiers in Artificial Intelligence 8 (2025). doi:10.3389/frai.2025.1615550.

[6] T. Chen et al., TVM: An automated end-to-end optimizing compiler for deep learning, OSDI 2018.

[7] O. Shafi et al., Demystifying TensorRT on NVIDIA edge devices, IISWC 2021. doi:10.1109/IISWC53511.2021.00030.

[8] J. Ansel et al., PyTorch 2: Faster machine learning through dynamic Python bytecode transformation and graph compilation, ASPLOS 2024. doi:10.1145/3620665.3640366.

[9] B. Kerfoot et al., The 2024 Brain Tumor Segmentation (BraTS) Challenge: Glioma segmentation on post-treatment MRI, arXiv:2405.18368 (2024).

[10] S. Markidis et al., NVIDIA Tensor Core programmability, performance and precision, IPDPSW 2018. doi:10.1109/IPDPSW.2018.00091.

[11] J. Choquette et al., NVIDIA A100 Tensor Core GPU: Performance and innovation, IEEE Micro 41 (2) (2021) 29-35. doi:10.1109/MM.2021.3061394.

[12] M. J. Page et al., The PRISMA 2020 statement: An updated guideline for reporting systematic reviews, BMJ 372 (2021) n71. doi:10.1136/bmj.n71.

[13] C. Guo et al., AMGFormer: Adaptive multi-granular transformer for brain tumor segmentation with missing modalities (2026). arXiv:2601.19349.

[14] Y. Ding et al., RFNet: Region-aware fusion network for incomplete multi-modal brain tumor segmentation, ICCV 2021. doi:10.1109/ICCV48922.2021.00394.

[15] W. Duan et al., MidFusNet: Mid-dense fusion network for multi-modal brain MRI segmentation, BrainLes 2023. doi:10.1007/978-3-031-33842-7_9.

[16] H. Zhang et al., Incomplete multi-modal brain tumor segmentation via learnable sorting state space, CVPR 2025.

[17] Y. Li et al., CCSD: Cross-modal compositional self-distillation for robust brain tumor segmentation with missing modalities, arXiv:2511.14599 (2025).

[18] B. H. Menze et al., The multimodal brain tumor image segmentation benchmark (BraTS), IEEE Transactions on Medical Imaging 34 (10) (2015) 1993-2024. doi:10.1109/TMI.2014.2377694.

[19] S. Bakas et al., Advancing TCGA glioma MRI collections with expert labels and radiomics, Scientific Data 4 (1) (2017). doi:10.1038/sdata.2017.117.

[20] O. Ronneberger et al., U-Net: Convolutional networks for biomedical image segmentation, MICCAI 2015. doi:10.1007/978-3-319-24574-4_28.

[21] O. Oktay et al., Attention U-Net: Learning where to look for the pancreas (2018). arXiv:1804.03999.

[22] F. Isensee et al., nnU-Net: a self-configuring method for deep learning-based biomedical image segmentation, Nature Methods 18 (2) (2021) 203-211. doi:10.1038/s41592-020-01008-z.

[23] F. Isensee et al., nnU-Net revisited: A call for rigorous validation in 3D medical image segmentation, MICCAI 2024. doi:10.1007/978-3-031-72114-4_47.

[24] Z. Huang et al., STU-Net: Scalable and transferable medical image segmentation models empowered by large-scale supervised pre-training (2023). arXiv:2304.06716.

[25] L. Wei et al., Advances in neural network quantization: A comprehensive review, Applied Sciences 14 (17) (2024). doi:10.3390/app14177445.

[26] A. Dequino et al., Optimizing bfloat16 deployment of tiny transformers on ultra-low power edge SoCs, Journal of Low Power Electronics and Applications 15 (1) (2025). doi:10.3390/jlpea15010008.

[27] T. Liang et al., Pruning and quantization for deep neural network acceleration: A survey, Neurocomputing 461 (2021) 370-403. doi:10.1016/j.neucom.2021.07.045.

[28] M. Sahiner et al., MedPTQ: A practical pipeline for real post-training quantization in 3D medical image segmentation (2025).

[29] T. Gale et al., The state of sparsity in deep neural networks, arXiv:1902.09574 (2019).

[30] A. Vaswani et al., Attention is all you need, NeurIPS 2017.

[31] J. Chen et al., TransUNet: Rethinking the U-Net architecture design for medical image segmentation through the lens of transformers, Medical Image Analysis 97 (2024) 103280. doi:10.1016/j.media.2024.103280.

[32] S. Roy et al., MedNeXt: Transformer-driven scaling of ConvNets for medical image segmentation, MICCAI 2023.

[33] X. Jiang et al., Vision transformer promotes cancer diagnosis: A comprehensive review, Expert Systems with Applications 252 (2024). doi:10.1016/j.eswa.2024.124113.

[34] J. Sun, MedFusion-TransNet: multi-modal fusion via transformer for enhanced medical image segmentation, Frontiers in Medicine 12 (2025). doi:10.3389/fmed.2025.1557449.

[35] T. Moreau et al., A hardware-software blueprint for flexible deep learning specialization, IEEE Micro 39 (5) (2019) 8-16. doi:10.1109/MM.2019.2928962.

[36] NVIDIA Corporation, TensorRT Developer Guide, NVIDIA Documentation (2026).

[37] A. Gu, T. Dao, Mamba: Linear-time sequence modeling with selective state spaces, COLM 2024 (preprint 2023). arXiv:2312.00752.

[38] T. Dao, A. Gu, Transformers are SSMs: Generalized models and efficient algorithms through structured state space duality, ICML 2024. arXiv:2405.21060.

[39] J. Ruan et al., VM-UNet: Vision Mamba UNet for medical image segmentation, ACM Transactions on Multimedia Computing, Communications, and Applications (2025). doi:10.1145/3767748.

[40] R. Zhou et al., Cascade residual multiscale convolution and Mamba-structured U-Net for advanced brain tumor segmentation, Entropy 26 (5) (2024). doi:10.3390/e26050385.

[41] H. Zheng et al., Gated differential linear attention: A linear-time decoder for high-fidelity medical segmentation (2026). arXiv:2603.02727.

[42] N. P. Jouppi et al., A domain-specific architecture for deep neural networks, Communications of the ACM 61 (9) (2018) 50-59. doi:10.1145/3154484.

[43] P. Yang et al., Progressive distillation with optimal transport for federated incomplete multi-modal learning of brain tumor segmentation, IEEE Transactions on Medical Imaging, early access (2025).

[44] M. Havaei et al., HeMIS: Hetero-modal image segmentation, MICCAI 2016.

[45] A. Sathe et al., Optimizing and deploying transformer INT8 inference with ONNX Runtime-TensorRT on NVIDIA GPUs, Microsoft Open Source Blog (2022).

[46] B. Jacob et al., Quantization and training of neural networks for efficient integer-arithmetic-only inference, CVPR 2018.

[47] E. Frantar et al., GPTQ: Accurate post-training quantization for generative pre-trained transformers, ICLR 2023.

[48] S. Han et al., Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding, ICLR 2016.

[49] G. Hinton et al., Distilling the knowledge in a neural network, arXiv:1503.02531 (2015).

[50] S. Gou et al., Knowledge distillation: A survey, International Journal of Computer Vision 129 (2021) 1789-1819.

[51] Y. Weng et al., NAS-Unet: Neural architecture search for medical image segmentation, IEEE Access 7 (2019) 44247-44257. doi:10.1109/ACCESS.2019.2908991.

[52] H. Cai et al., Once-for-all: Train one network and specialize it for efficient deployment, ICLR 2020.

[53] H. Pham et al., Efficient neural architecture search via parameter sharing, ICML 2018.

[54] A. Paszke et al., PyTorch: An imperative style, high-performance deep learning library, NeurIPS 2019.

[55] M. Looks et al., Deep learning with dynamic computation graphs, ICLR 2017.

[56] PyTorch Team, torch.compile end-to-end tutorial and performance tuning guide, PyTorch Documentation (2024).

[57] NVIDIA Corporation, NVIDIA H100 Tensor Core GPU Architecture Whitepaper (2022).

[58] T. Rohlfing et al., The SRI24 multichannel atlas of normal adult human brain structure, Human Brain Mapping 31 (5) (2010) 798-819. doi:10.1002/hbm.20906.

[59] M. J. Cardoso et al., MONAI: An open-source framework for deep learning in healthcare, arXiv:2211.02701 (2022).

[60] F. Perez-Garcia et al., TorchIO: A Python library for efficient loading, preprocessing, augmentation and patch-based sampling of medical images in deep learning, Computer Methods and Programs in Biomedicine 208 (2021) 106236. doi:10.1016/j.cmpb.2021.106236.