
LoMix: Learnable Weighted Multi-Scale Logits Mixing for Medical Image Segmentation

Md Mostafijur Rahman Radu Marculescu
Department of Electrical and Computer Engineering
The University of Texas at Austin
Austin, TX 78703
{mostafijur.rahman, radum}@utexas.edu

Abstract

U-shaped networks output logits at multiple spatial scales, each capturing a different blend of coarse context and fine detail. Yet, training still treats these logits in isolation—either supervising only the final, highest-resolution logits or applying deep supervision with identical loss weights at every scale—without exploring *mixed-scale* combinations. Consequently, the decoder output misses the complementary cues that arise only when coarse and fine predictions are fused. To address this issue, we introduce LoMix (Logits Mixing), a Neural Architecture Search (NAS)-inspired, differentiable plug-and-play module that **generates** new mixed-scale outputs and **learns** how exactly each of them should guide the training process. More precisely, LoMix mixes the multi-scale decoder logits with four lightweight fusion operators: addition, multiplication, concatenation, and attention-based weighted fusion, yielding a rich set of synthetic “mutant” maps. Every original or mutant map is given a softplus loss weight that is co-optimized with network parameters, mimicking a one-step architecture search that automatically discovers the most useful scales, mixtures, and operators. Plugging LoMix into recent U-shaped architectures (i.e., PVT-V2-B2 backbone with EMCAD decoder) on Synapse 8-organ dataset improves DICE by +4.2% over single-output supervision, +2.2% over deep supervision, and +1.5% over equally weighted additive fusion, all with **zero** inference overhead. When training data are scarce (e.g., one or two labeled scans, 5% of the trainset), the advantage grows to +9.23%, underscoring LoMix’s data efficiency. Across four benchmarks and diverse U-shaped networks, LoMix improves DICE by up to +13.5% over single-output supervision, confirming that learnable weighted mixed-scale fusion generalizes broadly while remaining data efficient, fully interpretable, and overhead-free at inference. Our implementation is available at <https://github.com/SLDGroup/LoMix>.

1 Introduction

Precise delineation of organs, tumours, and lesions underpins radiotherapy planning, volumetric assessment, and computer-aided diagnosis. State-of-the-art systems almost invariably adopt U-shaped encoder–decoder architectures such as UNet [25], UNet++ [36], Attn-UNet [19], TransUNet [3], and SwinUNet [2]. These models generate logit maps at multiple decoder resolutions: coarse maps offer global anatomical context, whereas fine maps sharpen boundaries and reveal small pathologies.

Surprisingly, the prevailing training protocols ignore most of this multi-scale richness: they either back-propagate loss only from the final logits (single-output supervision) or apply deep supervision (i.e., auxiliary losses on intermediate network outputs) with an identical loss weight for every scale [36, 8, 20, 23]. This uniform treatment presumes that each resolution is equally informative for every anatomy, an assumption contradicted by clinical practice, where minute, high-contrast structures

(e.g. pancreas, gall-bladder) lean heavily on fine-scale details, whereas large homogeneous organs (e.g. liver) profit chiefly from coarse context. The resulting mismatch leaves scale-specific clues under-exploited, and the deficiency grows when labels are scarce.

Deep supervision is also *isolationist* in nature as it overlooks the synergy that can emerge when coarse- and fine-grain logits are *combined*. Early “mutation” methods attempt to bridge this gap by summing up the decoder logits with equal weights, with MERIT [21] being such a prominent example. However, such static fusion fixes the operator, enforces equal contributions from all scales, and demands manual retuning whenever imaging protocols, organ sizes, or data volumes change. General loss-balancing schemes developed for multi-task learning, such as uncertainty weighting [14] or GradNorm [5], do not capture the structured correlations inside a single-task, multi-scale decoder, and thus leave much of this information content untapped.

To address these limitations, we introduce **LoMix** (Logits Mixing) to convert passive deep supervision into an *active, learnable ensemble* of mixed-scale predictions. During training every pair of decoder stages is fused by four lightweight, differentiable operators: pixel-wise addition, multiplication, concatenation followed by a 1×1 convolution, and attention-based weighted fusion, resulting in a rich family of “mutant” logits that explicitly blend coarse context with fine detail. Each original or mutant map is modulated by a soft-plus weight optimized jointly with the network, so the model performs a Neural Architecture Search (NAS) style selection of the most informative scales and fusion modes *within* the main optimization loop; that is, no extra optimizer, no validation-set grid search is needed. The added parameters are negligible and used only during training, leaving the FLOPs, latency, and memory footprint at test-time unchanged, while providing a data-driven fusion that adapts to organ size, image contrast, and label scarcity.

When integrated into a recent U-shaped network, PVT-V2-B2 backbone [34] with EMCAD decoder [24], LoMix improves the mean DICE score on Synapse 8-organ segmentation by **+4.2%** over single-output supervision, **+2.2%** over uniform deep supervision, and **+1.5%** over equal-weight additive fusion; of note, the gains are even larger on the harder Synapse 13-organ segmentation task. Consistent improvements on ACDC cardiac MRI and BUSI breast-tumour ultrasound across both CNN and transformer backbones confirm robustness of LoMix. Even with only 5% of training scans available, LoMix still delivers a **+9.23%** DICE improvement, underscoring the data efficiency.

In summary, LoMix (i) is the **first framework** that jointly optimizes *which* decoder scales to mix and *how* to mix them for a single-task; (ii) **substitutes manual loss weighting** with an automatic, interpretable, NAS-inspired weighting mechanism; (iii) introduces **zero inference overhead**; and (iv) **consistently improves performance** across datasets, backbones, and annotation budgets. By allowing networks to learn their own multi-scale fusion strategy, LoMix offers a principled and practical advancement that puts forth a strong baseline for data-efficient medical image segmentation.

The remainder of this paper is structured as follows: Section 2 discusses related literature, Section 3 details the LoMix framework, Section 4 explains experimental evaluations, Section 5 presents several critical ablation studies, and Section 6 concludes our findings and suggests future research directions.

2 Related Work

Medical Segmentation Architectures: U-shaped encoder–decoder networks with skip connections (e.g., U-Net) are the de facto architectures in medical image segmentation [25]; they can capture fine details via multi-scale feature maps, but are limited by the locality of convolutional operations. For example, Chen et al. note that standard U-Net struggles to model long-range dependencies, motivating hybrid designs [3]. To address this, recent work has proposed transformer-based backbones for segmentation. TransUNet [3] combines a CNN encoder with a Vision Transformer to learn the global context, while the decoder recovers the spatial details. Similarly, Swin-Unet [2] uses a hierarchical Swin Transformer in both encoder and decoder, demonstrating that pure-transformer U-shaped models outperform purely convolutional ones on multi-organ tasks.

Other variants adopt the Pyramid Vision Transformer (PVT) [34] as the encoder. For instance, CASCADE [20] and G-CASCADE [22] use PVT encoders with novel attention- or graph-based decoders to progressively refine multi-scale features. Polyp-PVT [7] and SSFormer [33] also leverage PVT backbones for polyp segmentation, incorporating hand-crafted fusion modules (e.g., cascaded fusion, camouflage, and locality decoders) to combine features across scales.

More recently, EMCAD [24] introduces an efficient multi-scale convolutional attention decoder that uses depth-wise convolutions and gated attention to fuse multi-resolution features. While these transformer-based models generate rich multi-scale outputs, they typically use fixed fusion or skip-connection schemes and do not learn explicit weights for combining feature maps.

Deep Supervision and Static Multi-Scale Fusion: Deep supervision has been widely adopted to improve training of segmentation models by attaching auxiliary loss functions to intermediate layers. For example, UNet++ [36] employs nested skip pathways: intermediate decoder outputs are each supervised by ground truth to encourage multi-scale consistency. In practice, however, these supervision losses are typically combined with fixed rules (e.g., simple averaging, summation, etc.).

Likewise, multi-scale fusion in many models is static. Common approaches concatenate or sum feature maps from different depths without learning the fusion weights. For example, Polyp-PVT’s [7] design includes fixed modules to merge encoder features across levels, but these components have pre-defined roles and uniform weighting. Such static fusion schemes (even when effective) do not adaptively learn which scales or channels to emphasize, leaving the relative contributions of multi-scale features unchanged during training.

Mutation-Based Training and Its Limitations: Some recent methods aim to exploit multi-scale logits through loss-level ensembling. MERIT [21] is such a notable example: it aggregates original multi-stage logits through an additive MUTATION mixing strategy. This loss aggregation ensembles logits from different scales for final training. However, the MUTATION *mixes uniformly* all decoder outputs (implicitly giving equal weight of 1 to each scale) using only element-wise addition and does not include learnable parameters for fusion. Similarly, EMCAD’s [24] multi-scale decoding produces several parallel predictions, but these are also combined using only additive fusion when computing loss. Hence, while mutation-based training can improve robustness via implicit ensembling, it lacks a trainable mechanism to re-weight or fuse multi-scale outputs using multiple operations dynamically.

Adaptive Weighting and NAS-Inspired Fusion: Learning to balance multi-scale signals has also been studied from the perspective of multi-task learning. Notably, uncertainty-weighted loss functions [14] and GradNorm [5] automatically tune loss weights across tasks. While these methods adaptively adjust the loss terms, they are designed for distinct tasks and do not directly apply to multi-scale outputs of a single task. They also do not explicitly handle feature-level fusion. Some NAS methods have explored learnable combinations of feature maps (e.g. NAS-FPN in detection [9]), but none learns operator-level fusion for multi-scale segmentation outputs. In contrast, to the best of our knowledge, LoMix is the *first approach* to enable fully learnable multi-scale fusion in segmentation: it synthesizes decoder outputs combinatorially (in the style of NAS cell search [18]) and learns softplus-parameterized weights to aggregate losses. This allows LoMix to adaptively emphasize relevant scales and operations during training, going beyond fixed strategies used in prior work.

3 Method

Next, we first review the background on U-shaped architectures and multi-scale outputs, then formalize the problem, and describe each component (see Figure 1).

3.1 Background: U-shaped Networks and Multi-scale Outputs

U-shaped networks [25, 24, 10], employ an encoder-decoder design with skip connections. The encoder path repeatedly downsamples the input to capture hierarchical features (see Figure 1(a)), while the decoder path upsamples these features to reconstruct the segmentation map (see Figure 1(b)). Skip connections between corresponding encoder and decoder layers ensure that fine spatial details lost during downsampling are recovered at the output.

To further leverage multi-scale context, many networks produce outputs at multiple decoder stages. For example, deeply supervised networks generate intermediate segmentation maps ($P_1, P_2, P_3, \dots, P_L$ in Figure 1) at different scales, thus providing richer gradients during training and allowing each decoder stage to capture structures at its respective scale.

In medical imaging, this multi-scale strategy is especially beneficial, since anatomical structures can vary greatly in size. UNet++ [36] is one instantiation of this idea, introducing dense nested skip

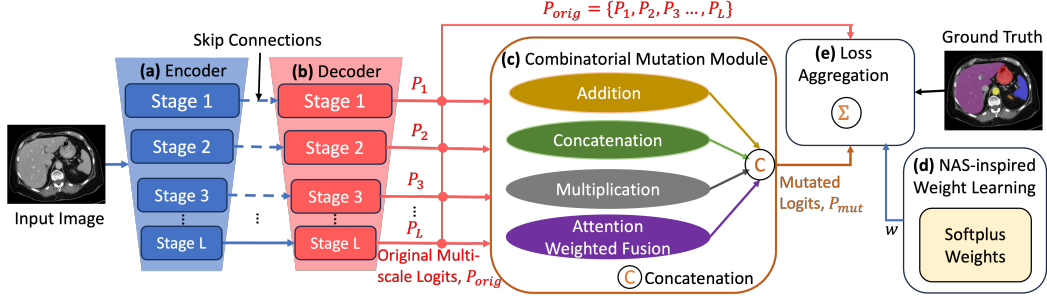


Figure 1: The proposed LoMix supervision strategy during training. (a-b) An input image is processed by a U-shaped network, producing original multi-scale logits P_{orig} from different stage of decoders. (c) The *Combinatorial Mutation Module* synthesizes additional predictions by applying four fusion operators to every non-trivial original prediction subset: **Addition**, **Concatenation**, **Multiplication**, and a learnable **Attention-Weighted Fusion (awf)**. (d) NAS-inspired weight-learning produces differentiable *softplus*-transformed weights w for every original logit in P_{orig} or mutated logit in P_{mut} , optimized jointly with network parameters via back-propagation. (e) All original and mutated logits are supervised by a loss objective weighted by $\{w\}$.

connections and side-output layers that fuse information across scales. Such multi-scale outputs encourage consistency across resolutions and improve medical image segmentation accuracy.

3.2 Problem Definition: Logit Mixing (LoMix)

Let X denote an input image of size $H \times W$ and let $Y \in \{1, \dots, C\}^{H \times W}$ be the ground truth with C classes. A U-shaped network with L decoder stages produces L logit maps $\{Z_\ell(X)\}_{\ell=1}^L$, where each $Z_\ell \in \mathbb{R}^{C \times H_\ell \times W_\ell}$ has spatial resolution $H_\ell \times W_\ell$. Upsampling each Z_ℓ to the full resolution gives logit maps $P_\ell(X) = \sigma(Z_\ell(X)) \in [0, 1]^{C \times H \times W}$, with σ a softmax (for multi-class segmentation) or sigmoid (for binary segmentation), as appropriate. We denote the set of these original logits by $\mathcal{P}_{orig}$. LoMix also synthesizes a collection $\mathcal{P}_{mut}$ of “mutant” maps through fusion operators (addition, multiplication, concatenation, and attention-weighted fusion) in the *Combinatorial Mutation Module* (see Section 3.3 and Figure 1(c)), forming the combined set $\mathcal{P} = \mathcal{P}_{orig} \cup \mathcal{P}_{mut}$.

For each map $P_u \in \mathcal{P}$, we introduce a scalar $\alpha_u \in \mathbb{R}$ and convert it to a positive loss weight $w_u = \text{softplus}(\alpha_u) = \ln(1 + e^{\alpha_u}) > 0$ by a NAS-inspired weight learning (see Section 3.4 and Figure 1(d)). Given a segmentation loss $\mathcal{L}_{seg}(P, Y)$ (e.g. Cross-entropy + DICE), we train both the network parameters Θ and the loss-weight parameters $\{\alpha_u\}$ by minimizing $\min_{\Theta, \{\alpha_u\}} \sum_{P_u \in \mathcal{P}} w_u \mathcal{L}_{seg}(P_u(X), Y)$ (see Section 3.5 and Figure 1(e)). Because only the final decoder output P_L is used at the test time, LoMix adds zero inference overhead while adaptively learning which scales and fused combinations are most informative for robust segmentation.

3.3 Combinatorial Mutation Module (CMM)

In our LoMix framework, a U-shaped network produces L logit maps $P_1, \dots, P_L \in \mathbb{R}^{C \times H \times W}$, each at a progressively finer scale (all upsampled to a common $H \times W$ spatial size). Let $P_i(p) \in \mathbb{R}^C$ denote the logits of C classes at pixel $p \in \{1, \dots, H\} \times \{1, \dots, W\}$ from the i -th decoder output. The *Combinatorial Mutation Module* (CMM) creates additional fused logits by combining subsets of these logits under four operators. Specifically, for every non-trivial subset $S \subseteq \{1, \dots, L\}$ with $|S| \geq 2$, we define *fused logit maps* $P_S^{(op)}(p) \in \mathbb{R}^C$ (for $op \in \{add, mult, cat, awf\}$) as follows:

- **Addition (add):** We combine each subset S of the original logit maps by *element-wise addition* to produce fused map $P_S^{(add)}(p)$ as in Eq. 1:

$$P_S^{(add)}(p) = \sum_{i \in S} P_i(p) \quad (1)$$

Intuitively, Addition fusion aggregates confidence from each subset of logit maps. Addition fusion will highlight regions where either decoder is confident (acting like an OR operation).

- **Multiplication (mult):** We take the *element-wise (Hadamard) product* $P_S^{(mult)}(p)$ of each subset S of the original logit maps as in Eq. 2:

$$P_S^{(mult)}(p) = \prod_{i \in S} P_i(p) \quad (2)$$

Multiplication fusion provides high confidence only where *all* logit maps agree (analogous to an AND operation). This fusion thus focuses on the intersection of the decoders' predictions, which can enhance precision by reinforcing common correct predictions and canceling out disagreements (if either logit map is uncertain, the product lowers confidence).

- **Concatenation (cat):** We concatenate each subset S of the original logit maps *channel-wise* and apply a 1×1 convolution to fuse them to produce fused map $P_S^{(cat)}(p)$ as in Eq. 3:

$$P_S^{(cat)}(p) = W_S [P_i(p)]_{i \in S} \quad (3)$$

where $[P_i(p)]_{i \in S} \in \mathbb{R}^{|S|C}$ is the channel-wise concatenation and $W_S \in \mathbb{R}^{C \times (|S|C)}$ is a 1×1 convolution weight matrix (with output dimension equal to one logit map). This operation allows the network to learn an optimal pixel-wise linear combination of each subset of logits. The 1×1 convolution can be viewed as automatically weighting and combining the two inputs for each output class, potentially learning to trust one decoder more in certain regions and the other decoder elsewhere, based on data.

- **Attention-Weighted Fusion (awf):** We introduce an *attention gating* to adaptively mix each subset S of original logits. We first compute attention scores and normalize using Eq. 4:

$$\tilde{\alpha}_S(p) = W'_S [P_i(p)]_{i \in S}, \quad \alpha_{i,S}(p) = \frac{\exp(\tilde{\alpha}_{i,S}(p))}{\sum_{j \in S} \exp(\tilde{\alpha}_{j,S}(p))} \quad (4)$$

Then, we take the attention-weighted sum $P_S^{(wf)}(p)$ of *each subset of logits* as in Eq. 5:

$$P_S^{(awf)}(p) = \sum_{i \in S} \alpha_{i,S}(p) P_i(p) \quad (5)$$

Attention-Weighted Fusion can learn to favor the logit that is likely to be correct at each pixel of the image (for instance, one pixel might be better at fine details, another at coarse structure, so the attention gate can interpolate accordingly). It generalizes the addition fusion by allowing spatially varying weighting instead of a fixed linear mix at each pixel.

These four fusion operations help us produce *new segmentation predictions* from each subset of original logits without adding significant computation (each is a simple pixel-wise operation or a 1×1 conv). They are complementary: addition and multiplication are fixed arithmetic mixes (one expansive, other selective), while concatenation and attention are learnable mixes (one globally learned weight, another dynamically learned per-pixel weight). By supervising all of them, we expose the network to a wide variety of joint-decoder behaviors. The decoder stages are incentivized to cooperate because an error from one decoder stage can be corrected by another in a fused output, thus leading to an overall more accurate ensemble of logits. In our LoMix framework, we apply all four fusion operations to *every subset* of original logits. Adding all non-empty subsets of the L decoder predictions introduces only $2^L - 1 - L$ extra logit maps (e.g., 11 for $L = 4$, 26 for $L = 5$); the total number of fused (mutant) logits is $4(2^L - 1 - L)$. Combined with the L original logits, the overall count is upper bounded by $L + 4(2^L - 1 - L)$, which remains tractable for typical $L \leq 5$ since U-shaped networks rarely exceed five stages as in [25, 20, 24, 12].

3.4 NAS-Inspired Weight Learning

To enable the network to learn the relative importance of each logit map, we associate a trainable scalar weight with every original decoder output P_i and every fused (mutated) output $P_S^{(op)}$. Concretely, let $\alpha_i \in \mathbb{R}$ be the raw parameter for output P_i and $\alpha_S^{(op)} \in \mathbb{R}$ the parameter for fused output $P_S^{(op)}$. We map these parameters through the Softplus function to obtain strictly positive weights as in Eq. 6:

$$w_i = \text{softplus}(\alpha_i) = \ln(1 + e^{\alpha_i}), \quad w_S^{(op)} = \text{softplus}(\alpha_S^{(op)}) \quad (6)$$

By construction, $w_i > 0$ and $w_S^{(\text{op})} > 0$ for all i, op . These weights $w_i, w_S^{(\text{op})}$ serve as learnable scaling factors on the loss contributions of each corresponding logit map. All parameters α_i and $\alpha_S^{(\text{op})}$ are learned jointly with the network weights via backpropagation on the overall training objective.

These design choices allow the model to automatically learn how much emphasis to place on each original and fused logit during training, in a manner reminiscent of architecture weighting in NAS but applied to losses. **The learned weights are shown in Appendix A.6 of Supplementary Material.**

3.5 Loss Aggregation

Each output logits map is trained with a standard segmentation loss (e.g., Cross-Entropy and DICE losses). For the ground-truth mask Y and the i -th original output P_i , we define its per-output loss as in Eq. 7. Similarly, per-output loss for each fused output $P_S^{(\text{op})}$ is defined in Eq. 8:

$$\mathcal{L}_i = \beta \mathcal{L}_{\text{CE}}(P_i, Y) + \gamma \mathcal{L}_{\text{DICE}}(P_i, Y) \quad (7)$$

$$\mathcal{L}_S^{(\text{op})} = \beta \mathcal{L}_{\text{CE}}(P_S^{(\text{op})}, Y) + \gamma \mathcal{L}_{\text{DICE}}(P_S^{(\text{op})}, Y) \quad (8)$$

Here, $\mathcal{L}_{\text{CE}}$ is the Cross-entropy loss weighted by β and $\mathcal{L}_{\text{DICE}}$ is the DICE loss weighted by γ (here, $\beta + \gamma = 1$ and $\beta, \gamma > 0$). The total training loss is then formed by weighting each output’s loss by the corresponding learned softplus weight and summing as in Eq. 9:

$$\mathcal{L}_{\text{total}} = \sum_{i=1}^L w_i \mathcal{L}_i + \sum_{\text{op}} w_S^{(\text{op})} \mathcal{L}_S^{(\text{op})} \quad (9)$$

Weighting the loss terms (rather than directly combining logits) provides several benefits:

- First, it preserves distinct supervision for each output: each P_i and $P_S^{(\text{op})}$ is individually trained and can receive gradients weighted by its own weights w . If a certain mutated logit $P_S^{(\text{op})}$ proves to be unhelpful or noisy, then the model can drive its $w^{(\text{op})}$ toward zero, effectively ignoring its loss contribution. Conversely, if an output is beneficial, its weight can be increased to emphasize it. This dynamic loss weighting is analogous to multi-task learning schemes where uncertainty or task relevance modulates loss terms.
- Second, it avoids the pitfalls of weighting logits directly: mixing logits into a single prediction would blur their individual contributions and could hinder training of underperforming branches. Instead, our weighted loss formulation (Eq. 9) allows the network to automatically focus on useful logit outputs while minimizing the impact of less informative ones, thus leading to more effective training of the ensemble of original and mutated logits.

4 Experimental Evaluation

We evaluate LoMix on several medical image segmentation datasets. **Datasets, additional results and analyses including qualitative visualization are provided in the Supplementary Material.**

4.1 Implementation details

Our methods are implemented and evaluated using Pytorch 1.11.0, operating on a single NVIDIA RTX A6000 GPU equipped with 48GB of RAM. We use the PVT-EMCAD-B2 as a default model [24] in our experiments with multi-scale kernels $[1 \times 1, 3 \times 3, 5 \times 5]$ and four stages unless otherwise mentioned. We consider all four operators (Addition, Multiplication, Concatenation, Attention-Weighted Fusion) with NAS-inspired learnable softplus weights in our LoMix supervision. Only the last-stage prediction from the decoder is used as the final segmentation output. Model optimization is achieved with AdamW [17] optimizer with learning rate and weight decay set to $1e - 4$.

4.2 Comparison with SOTA Methods

Synapse 8-organ Segmentation: Table 1 reports Synapse 8-organ results averaged over at least three runs, comparing single-output supervision (LL), Deep Supervision (DS) [36], MUTATION [21], and

Table 1: Synapse 8-organ segmentation with Last Layer (LL), Deep Supervision (DS) [36], MUTATION [21], and our LoMiX. DICE scores (%) are reported for Gallbladder (GB), Left kidney (KL), Right kidney (KR), Pancreas (PC), Spleen (SP), and Stomach (SM). $\uparrow$ denotes the higher the better and $\downarrow$ denotes the lower the better. Results are averaged over at least three runs. Two-sided Wilcoxon signed-rank tests [35] indicate that LoMiX significantly outperforms LL and DS at $\alpha = 0.05$. Best results are shown in **bold**.

Methods	Average			Per-organ DICE (%) $\uparrow$							
	DICE (%) $\uparrow$	HD95 $\downarrow$	mIoU (%) $\uparrow$	Aorta	GB	KL	KR	Liver	PC	SP	SM
UNet [25] + LL	70.1	44.7	59.4	84.0	56.7	72.4	62.6	87.0	48.7	81.5	67.9
+ DS	77.8	26.9	68.3	85.4	68.0	81.4	76.2	91.4	56.9	87.6	75.9
+ MUTATION	81.5	26.4	71.8	89.4	70.5	85.4	80.4	94.1	66.3	88.3	77.5
+ LoMiX (Ours)	83.6	24.3	74.6	90.4	75.3	86.3	82.5	94.3	67.9	91.8	80.2
AttUNet [19] + LL	71.7	34.5	61.4	82.6	61.9	76.1	70.4	87.5	46.7	80.7	67.7
+ DS	77.9	29.9	68.1	85.4	67.5	81.4	77.4	91.0	57.2	87.1	76.3
+ MUTATION	82.6	19.9	73.1	88.1	73.8	86.3	80.5	94.1	69.2	90.4	78.7
+ LoMiX (Ours)	83.0	19.4	74.1	90.0	75.5	84.4	81.4	94.5	67.2	91.3	79.7
TransUNet [3] + LL	77.6	26.9	67.3	86.6	60.4	80.5	78.5	94.3	58.5	87.1	75.0
+ DS	82.7	17.3	73.5	86.6	68.5	87.7	84.6	94.4	65.3	90.8	83.5
+ MUTATION	83.0	17.0	73.9	89.3	63.7	86.9	83.0	95.5	69.6	93.1	82.7
+ LoMiX (Ours)	83.6	16.6	74.6	88.9	70.3	89.4	85.2	94.8	67.8	89.4	83.0
UNeXt [31] + LL	70.5	29.2	60.1	81.9	30.6	80.8	75.8	92.3	48.4	84.1	70.3
+ DS	72.6	30.7	61.3	80.2	60.8	76.1	70.0	91.8	48.0	83.3	70.6
+ MUTATION	75.6	28.1	64.4	81.8	61.6	81.1	75.1	92.6	53.1	85.7	73.9
+ LoMiX (Ours)	76.8	22.7	66.2	83.8	60.9	81.1	78.3	92.8	56.2	86.6	74.5
PVT-CASCADE-B2 [20] + LL	80.8	20.5	71.8	85.6	66.6	84.1	81.0	92.9	67.0	90.0	79.5
+ DS	81.1	20.2	70.9	83.0	70.6	82.2	80.4	94.1	64.4	90.1	83.7
+ MUTATION	83.0	17.8	74.3	86.9	67.6	87.1	82.1	94.4	68.7	91.8	85.6
+ LoMiX (Ours)	84.3	16.4	75.4	86.5	72.8	87.4	84.6	95.6	70.1	92.6	84.8
PVT-EMCAD-B2 [24] + LL	80.9	22.9	71.2	87.1	68.0	84.9	81.1	94.6	63.1	89.8	78.9
+ DS	82.9	19.7	73.8	87.4	67.8	87.7	83.7	95.2	65.6	91.5	84.2
+ MUTATION	83.6	15.7	74.7	88.1	68.9	88.1	84.1	95.3	68.5	92.2	83.9
+ LoMiX (Ours)	85.1	14.9	76.4	88.8	73.5	89.1	84.7	95.8	69.7	92.5	86.5

LoMiX across six representative backbones spanning convolutional and transformer families (UNet [25], AttUNet [19], TransUNet [3], UNeXt [31], PVT-CASCADE-B2 [20], and PVT-EMCAD-B2 [24]). LoMiX consistently achieves the highest average DICE and mIoU and the lowest HD95 for all backbones, improving DICE by +3–5% on average over deep supervision (DS) and by up to +13.5% over last-layer (LL) supervision, without any inference time overhead. The gains are especially pronounced on small or challenging organs such as gallbladder and pancreas, indicating that learnable mixed-scale fusion recovers complementary fine detail and coarse context that prior supervision schemes fail to exploit. These results demonstrate that LoMiX is a unified, lightweight, and plug-and-play training module that generalizes across diverse U-shaped decoders and transformer backbones while remaining fully compatible with standard inference pipelines.

ACDC Cardiac Organ Segmentation: Table 2 reports the cardiac organ segmentation on ACDC dataset, averaging over at least three runs. We can see that across a broad set of CNN and transformer models, our LoMiX-enhanced models consistently deliver superior performance. Notably, PVT-EMCAD-B2 + LoMiX attains a new peak average DICE of 92.51%, surpassing its baseline PVT-EMCAD-B2 model (92.12%) and all prior methods. It achieves the highest scores on every structure: RV 91.41%, Myo 89.96%, and LV 96.15%. Even the lightweight PVT-EMCAD-B0 + LoMiX improves over PVT-EMCAD-B0 (91.34% $\rightarrow$ 91.69%), matching or exceeding more complex cascaded and MERIT-based models. These gains confirm that LoMiX’s learnable, multi-scale fusion yields more accurate and robust cardiac contours—particularly evident in the challenging myocardium region—while maintaining identical inference complexity.

4.3 Evaluation with Limited Data

Limited Data Setup: We create subsets using 5% (1 scan), 10% (2 scans), 20% (4 scans), and 40% (7 scans) of the Synapse training set to evaluate performance under constrained supervision.

Figure 2 and Table 3 illustrate how LoMiX consistently outperforms conventional single-head (“Last-Layer”) supervision as the amount of labeled data shrinks. In Table 3 and Figure 2(A), with 40% of

Table 2: Results of cardiac organ segmentation on ACDC dataset. DICE scores (%) are reported for Right ventricle (RV), Myocardium (Myo), and Left ventricle (LV). $\uparrow$ ($\downarrow$) denotes the higher (lower) the better. Results are averaged over at least three runs. Best results are shown in **bold**.

Methods	Avg. DICE (%) $\uparrow$	RV $\uparrow$	Myo $\uparrow$	LV $\uparrow$
UNet [25]	87.55	87.10	80.63	94.92
Attn_UNet [19]	86.75	87.58	79.20	93.47
ViT+CUP [3]	81.45	81.46	70.71	92.18
TransUNet [3]	89.71	86.67	87.27	95.18
SwinUNet [2]	88.07	85.77	84.42	94.03
MT-UNet [32]	90.43	86.64	89.04	95.62
MISSFormer [12]	90.86	89.55	88.04	94.99
PVT-CASCADE [20]	91.46	89.97	88.9	95.50
TransCASCADE [20]	91.63	90.25	89.14	95.50
Rolling_UNet_S [16]	87.59	85.02	83.59	94.17
CMUNeXt [30]	85.19	81.30	82.54	91.74
UNeXt [31]	84.68	81.06	81.22	91.76
Cascaded MERIT [21]	91.85	90.23	89.53	95.80
PVT-GCASCADE [22]	91.95	90.31	89.63	95.91
EGE-UNet [27]	80.68	76.6	75.21	90.23
PVT-EMCAD-B0 [24]	91.34	89.37	88.99	95.65
PVT-EMCAD-B2 [24]	92.12	90.65	89.68	96.02
PVT-EMCAD-B0 + LoMix (Ours)	91.69 $\pm$ 0.51	90.33	88.99	95.75
PVT-EMCAD-B2 + LoMix (Ours)	92.51 $\pm$ 0.47	91.41	89.96	96.15

Table 3: Performance of LoMix on Synapse 8-organ segmentation under limited data. Gallbladder (GB), Left kidney (KL), Right kidney (KR), Pancreas (PC), Spleen (SP), and Stomach (SM). $\uparrow$ higher is better, $\downarrow$ lower is better. Each row is averaged over five runs. The best entries for each limited data setting are shown in **bold**.

Data Fraction	Scheme	Average			Per-organ DICE (%) $\uparrow$							
		DICE $\uparrow$	HD95 $\downarrow$	mIoU $\uparrow$	Aorta	GB	KL	KR	Liver	PC	SP	SM
40% (7 scans)	Last Layer	71.43	35.71	61.27	83.93	60.05	74.44	71.75	92.72	53.88	78.45	56.21
	LoMix (Ours)	76.49	27.14	66.67	87.27	66.27	82.81	74.60	95.05	59.29	82.61	64.05
20% (4 scans)	Last Layer	64.40	36.50	54.74	80.92	35.71	74.17	69.74	90.13	39.05	74.04	51.39
	LoMix (Ours)	69.21	29.86	59.60	84.11	41.59	77.86	72.24	94.47	47.98	80.83	54.60
10% (2 scans)	Last Layer	55.32	46.18	45.69	70.37	27.73	66.82	60.38	83.64	31.79	69.92	31.93
	LoMix (Ours)	64.53	38.58	54.55	75.81	37.29	74.01	70.26	92.33	45.33	82.50	38.73
5% (1 scan)	Last Layer	37.22	72.21	28.83	49.14	23.54	41.24	41.17	73.77	10.13	46.42	12.36
	LoMix (Ours)	46.45	56.94	37.36	57.58	26.92	50.30	49.48	80.40	16.44	65.72	24.75

the Synapse training set, LoMix improves the mean DICE score +5.1%; at 20% data the gain is +4.8%, and when supervision drops to only 10% and 5% the improvement margins surge to $> +9\%$. The radar plot on the right (Figure 2(B)) shows that these improvements are not confined to a single organ: LoMix raises DICE scores for every organ, with the largest boosts on small, hard-to-segment classes such as gallbladder (GB), pancreas (PC), and stomach (SM), confirming that dynamically weighting complementary scales is especially beneficial where context/details trade-offs are hardest. Hence, LoMix delivers uniform, per-organ gains and turns the U-shaped PVT-EMCAD-B2 architecture into a far more data-efficient and across-organ robust learner without any added inference cost.

5 Ablation Study

This section describes three critical ablation studies. **More ablation results and analyses are provided in the Supplementary Material.**

5.1 Fusion Operation Ablation

Figure 3 shows that DICE scores increase systematically as we expand the pool of fusion operators. In Figure 3(A), starting from a single operator (AWF, leftmost bar), adding Multiply or Add provides modest gains, while injecting the Concat operation provides the sharpest jump in mean DICE.

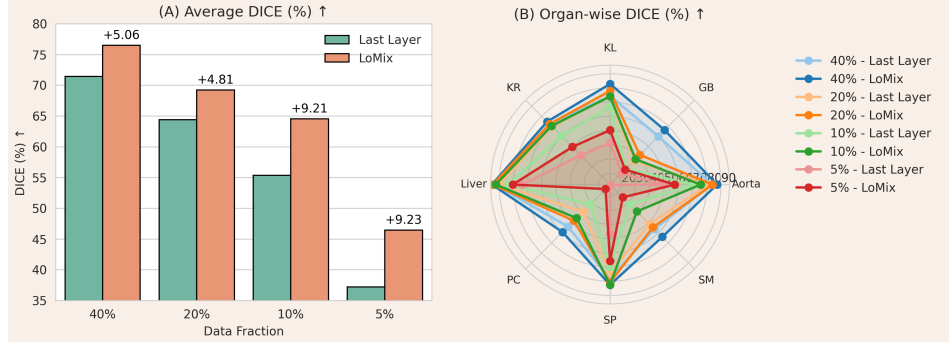


Figure 2: Limited data evaluation on Synapse 8-organ segmentation with PVT-EMCAD-B2. LoMix uses NAS-inspired softplus weighting. ↑ denotes higher is better. Results are averaged over five runs. Detailed results are provided in Table 3.

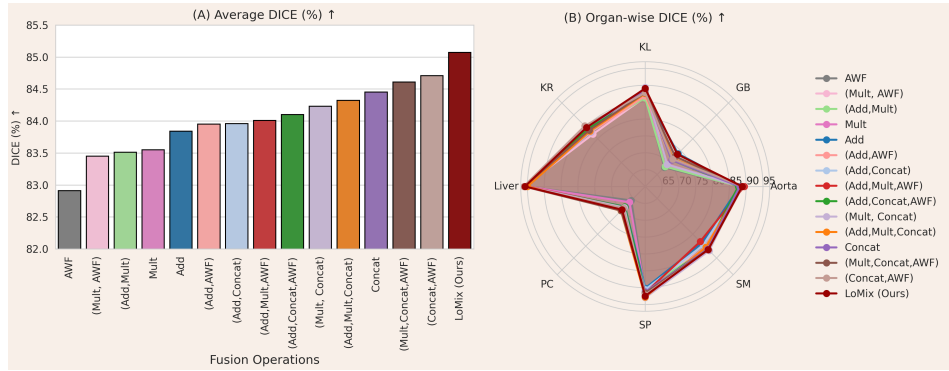


Figure 3: Comparison of different fusion operation combinations using NAS-inspired Softplus weights and PVT-EMCAD-B2 model for Synapse 8-organ segmentation. ↑ indicates higher is better.

Each additional operator gives the softplus search greater freedom to discover complementary scale interactions, and the trend is strictly monotonic: the full LoMix variant that activates all four operators tops the chart at 85.07% DICE, outperforming the best three-operator setting by 0.46%. The radar plot (Figure 3(B)) shows that these improvements are broader, LoMix encloses smaller polygons for every organ, with the largest margins on hard, low-contrast structures (GB, PC, SM) while still improving saturated classes such as liver and aorta toward their performance ceiling. In short, this ablation confirms that diverse operator choice, coupled with NAS-inspired weight learning, is critical: each extra operation opens a new pathway for the model to align coarse and fine cues, and the resultant mixture consistently translates into superior, organ-robust segmentation.

5.2 Effect of NAS-Inspired Weight Learning

Figure 4 contrasts fixed vs. learnable NAS-inspired softplus loss weighting across six supervision types. In Figure 4(A), deep supervision, single-operator fusions (addition, multiplication, concatenation, attention-weighted fusion (AWF)), and LoMix benefit from learning weights online instead of keeping them equal. The absolute gain ranges from +0.21% DICE (Add) to a pronounced +0.7% for Multiply, improving LoMix to 85.07% mean DICE score without changing architecture or inference cost. The radar plot (Figure 4(B)) shows that the learned variant never hurts any organ and delivers the largest jumps on the most scale-sensitive structures: gallbladder (GB) and left kidney (KL), while improving already strong classes (liver, spleen (SP)) toward the performance ceiling. Together, the results confirm that the learnable softplus weighting is a universal add-on: it tightens every supervision strategy but realizes its full potential when paired with LoMix’s rich operator pool.

5.3 Effect of LoMix on Backbone Architectures

Figure 5 demonstrates that LoMix is architecture-agnostic: whether the backbone is transformer-based (PVT-v2 [34]) or purely convolutional (ResNet [11]), replacing conventional supervision with

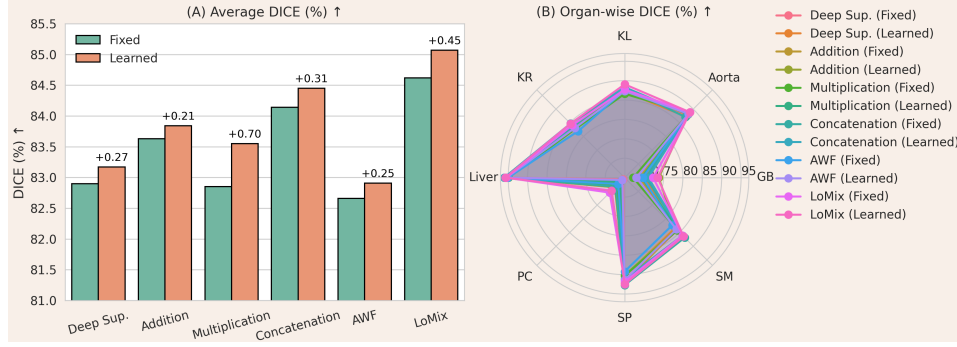


Figure 4: Effect of NAS-inspired Softplus weight learning on Synapse 8-organ segmentation with PVT-EMCAD-B2. Fixed = equal loss weights, Learned = NAS-inspired softplus weights.

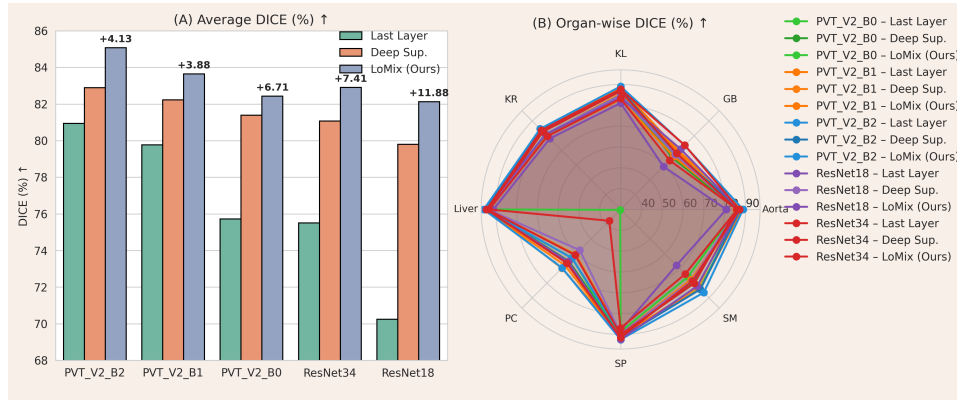


Figure 5: Comparison of different supervision schemes on Synapse 8-organ segmentation across five backbones. LoMix uses NAS-inspired softplus weighting. Sup.: Supervision.

our LoMix consistently improves performance. In Figure 5(A), mean DICE score increases with supervision strength: Last Layer < Deep Sup. < LoMix for every network. The absolute gain delivered by LoMix over single-head supervision is generalizable: +7.41-11.88% for ResNet variants and +3.88-6.71% on PVT-v2 variants which shows that our LoMix supervision unlocks benefits that standard decoders leave untapped. Figure 5(B) confirms that the improvements are broader: LoMix dominates other supervisions on all eight organs, with the largest margin gains again on scale-sensitive organs such as gallbladder (GB) and pancreas (PC). Crucially, these gains come at zero inference cost, thus underscoring LoMix’s practicality as a universal booster for existing segmentation backbones.

6 Conclusion

This work introduced LoMix, a plug-and-play, NAS-inspired module that unlocks the untapped value of multi-scale decoder logits by (i) generating a rich family of mixed-scale predictions through four differentiable fusion operators and (ii) learning, via softplus gating, how strongly each real or fused map should guide training. Extensive experiments on four medical-image tasks demonstrate that LoMix consistently outperforms both single-output supervision and classical deep supervision baselines while incurring *no* extra inference cost. Because LoMix exposes its learned weights as explicit scalars, the fusion it learns is interpretable, transferable across backbones, and easy to fine-tune, offering practitioners a principled yet practical path to high quality segmentation without architectural redesign. Our current implementation of LoMix is limited to 2D medical segmentation tasks. Future work will extend the approach to dense prediction tasks beyond segmentation.

Acknowledgments and Disclosure of Funding

This work is supported in part by the NSF grant CCF-2531882, and in part by the iMAGiNE Consortium (<https://imagine.utexas.edu/>).

References

- [1] Al-Dhabyani, W., Gomaa, M., Khaled, H., Fahmy, A.: Dataset of breast ultrasound images. *Data in Brief* **28**, 104863 (2020)
- [2] Cao, H., Wang, Y., Chen, J., Jiang, D., Zhang, X., Tian, Q., Wang, M.: Swin-unet: Unet-like pure transformer for medical image segmentation. In: *Proceedings of the European Conference on Computer Vision Workshops*. pp. 205–218. Springer (2022)
- [3] Chen, J., Mei, J., Li, X., Lu, Y., Yu, Q., Wei, Q., Luo, X., Xie, Y., Adeli, E., Wang, Y., et al.: Transunet: Rethinking the u-net architecture design for medical image segmentation through the lens of transformers. *Medical Image Analysis* p. 103280 (2024)
- [4] Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **40**(4), 834–848 (2017)
- [5] Chen, Z., Badrinarayanan, V., Lee, C.Y., Rabinovich, A.: Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In: *International Conference on Machine Learning*. pp. 794–803. PMLR (2018)
- [6] Codella, N., Rotemberg, V., Tschandl, P., Celebi, M.E., Dusza, S., Gutman, D., Helba, B., Kalloo, A., Liopyris, K., Marchetti, M., et al.: Skin lesion analysis toward melanoma detection 2018: A challenge hosted by the international skin imaging collaboration (isic). *arXiv preprint arXiv:1902.03368* (2019)
- [7] Dong, B., Wang, W., Fan, D.P., Li, J., Fu, H., Shao, L.: Polyp-pvt: Polyp segmentation with pyramid vision transformers. *CAAI Artificial Intelligence Research* **2** (2023)
- [8] Fan, D.P., Ji, G.P., Zhou, T., Chen, G., Fu, H., Shen, J., Shao, L.: Pranet: Parallel reverse attention network for polyp segmentation. In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. pp. 263–273. Springer (2020)
- [9] Ghiasi, G., Lin, T.Y., Le, Q.V.: Nas-fpn: Learning scalable feature pyramid architecture for object detection. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. pp. 7036–7045 (2019)
- [10] Hassan, R., Mondal, M.R.H., Ahamed, S.I., Mostafa, F., Rahman, M.M.: An efficient dual-line decoder network with multi-scale convolutional attention for multi-organ segmentation. *Biomedical Signal Processing and Control* **112**, 108611 (2026)
- [11] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. pp. 770–778 (2016)
- [12] Huang, X., Deng, Z., Li, D., Yuan, X., Fu, Y.: Missformer: An effective transformer for 2d medical image segmentation. *IEEE Transactions on Medical Imaging* **42**(5), 1484–1494 (2023)
- [13] Jha, D., Smedsrud, P.H., Riegler, M.A., Halvorsen, P., De Lange, T., Johansen, D., Johansen, H.D.: Kvasir-seg: A segmented polyp dataset. In: *International Conference on Multimedia Modeling*. pp. 451–462. Springer (2019)
- [14] Kendall, A., Gal, Y., Cipolla, R.: Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. pp. 7482–7491 (2018)
- [15] Kim, T., Lee, H., Kim, D.: Uacanet: Uncertainty augmented context attention for polyp segmentation. In: *ACM International Conference on Multimedia*. pp. 2167–2175 (2021)

- [16] Liu, Y., Zhu, H., Liu, M., Yu, H., Chen, Z., Gao, J.: Rolling-unet: Revitalizing mlp’s ability to efficiently extract long-distance dependencies for medical image segmentation. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 38, pp. 3819–3827 (2024)
- [17] Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101 (2017)
- [18] Nekrasov, V., Chen, H., Shen, C., Reid, I.: Fast neural architecture search of compact semantic segmentation models via auxiliary cells. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 9126–9135 (2019)
- [19] Oktay, O., Schlemper, J., Folgoc, L.L., Lee, M., Heinrich, M., Misawa, K., Mori, K., McDonagh, S., Hammerla, N.Y., Kainz, B., et al.: Attention u-net: Learning where to look for the pancreas. In: Medical Imaging with Deep Learning (2018)
- [20] Rahman, M.M., Marculescu, R.: Medical image segmentation via cascaded attention decoding. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 6222–6231 (2023)
- [21] Rahman, M.M., Marculescu, R.: Multi-scale hierarchical vision transformer with cascaded attention decoding for medical image segmentation. In: Medical Imaging with Deep Learning (2023)
- [22] Rahman, M.M., Marculescu, R.: G-cascade: Efficient cascaded graph convolutional decoding for 2d medical image segmentation. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 7728–7737 (2024)
- [23] Rahman, M.M., Marculescu, R.: Mk-unet: Multi-kernel lightweight cnn for medical image segmentation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops. pp. 1042–1051 (2025)
- [24] Rahman, M.M., Munir, M., Marculescu, R.: Emcad: Efficient multi-scale convolutional attention decoding for medical image segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11769–11779 (2024)
- [25] Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical Image Computing and Computer-Assisted Intervention. pp. 234–241. Springer (2015)
- [26] Roy, S., Koehler, G., Ulrich, C., Baumgartner, M., Petersen, J., Isensee, F., Jaeger, P.F., Maier-Hein, K.H.: Mednext: transformer-driven scaling of convnets for medical image segmentation. In: International Conference on Medical Image Computing and Computer-Assisted Intervention. pp. 405–415. Springer (2023)
- [27] Ruan, J., Xie, M., Gao, J., Liu, T., Fu, Y.: Ege-unet: an efficient group enhanced unet for skin lesion segmentation. In: International Conference on Medical Image Computing and Computer-Assisted Intervention. pp. 481–490. Springer (2023)
- [28] Silva, J., Histace, A., Romain, O., Dray, X., Granado, B.: Toward embedded detection of polyps in wce images for early diagnosis of colorectal cancer. *Int. J. Comput. Assist. Radiol. Surg.* **9**(2), 283–293 (2014)
- [29] Tajbakhsh, N., Gurudu, S.R., Liang, J.: Automated polyp detection in colonoscopy videos using shape and context information. *IEEE Transactions on Medical Imaging* **35**(2), 630–644 (2015)
- [30] Tang, F., Ding, J., Quan, Q., Wang, L., Ning, C., Zhou, S.K.: Cmunext: An efficient medical image segmentation network based on large kernel and skip fusion. In: IEEE International Symposium on Biomedical Imaging. pp. 1–5. IEEE (2024)
- [31] Valanarasu, J.M.J., Patel, V.M.: Unext: Mlp-based rapid medical image segmentation network. In: International Conference on Medical Image Computing and Computer-Assisted Intervention. pp. 23–33. Springer (2022)

- [32] Wang, H., Xie, S., Lin, L., Iwamoto, Y., Han, X.H., Chen, Y.W., Tong, R.: Mixed transformer u-net for medical image segmentation. In: IEEE International Conference on Acoustics, Speech and Signal Processing. pp. 2390–2394. IEEE (2022)
- [33] Wang, J., Huang, Q., Tang, F., Meng, J., Su, J., Song, S.: Stepwise feature fusion: Local guides global. arXiv preprint arXiv:2203.03635 (2022)
- [34] Wang, W., Xie, E., Li, X., Fan, D.P., Song, K., Liang, D., Lu, T., Luo, P., Shao, L.: Pvt v2: Improved baselines with pyramid vision transformer. *Computational Visual Media* **8**(3), 415–424 (2022)
- [35] Wilcoxon, F.: Individual comparisons by ranking methods. In: Breakthroughs in statistics: Methodology and distribution, pp. 196–202. Springer (1992)
- [36] Zhou, Z., Rahman Siddiquee, M.M., Tajbakhsh, N., Liang, J.: Unet++: A nested u-net architecture for medical image segmentation. In: International Workshop on Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support, pp. 3–11. Springer (2018)

NeurIPS Paper Checklist

1. Claims

Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

Answer: [\[Yes\]](#)

Justification: The methodology and experimental results sections justify the claims made in the abstract and introduction. The scope of the contributions is also clear in the abstract and introduction.

Guidelines:

- The answer NA means that the abstract and introduction do not include the claims made in the paper.
- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. Limitations

Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [\[Yes\]](#)

Justification: The limitation is discussed in the conclusion as the current evaluations of our proposed LoMix supervision are limited to 2D medical segmentation tasks. In the future, we will extend experiments on 3D segmentation tasks.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. Theory assumptions and proofs

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [\[Yes\]](#)

Justification: The paper explains the theory and ideas in the methods section. The paper explains assumptions made in the methods as well.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. Experimental result reproducibility

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [\[Yes\]](#)

Justification: The paper explains the implementation details setup in the experimental evaluation section. The paper will also publish the source code and weights upon acceptance.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general, releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
 - (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
 - (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
 - (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
 - (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. Open access to data and code

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: The data is publicly available online as it uses BUSI, Synapse Multi-organ (BTCV), and ACDC datasets. The citations for these datasets are included. The code will be publicly available to reproduce the results.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so “No” is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. Experimental setting/details

Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: Yes, all of the experimental setups are explained and the same setup was used to evaluate all of the models.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. Experiment statistical significance

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: Including error bars would take too much space. The standard deviation range of 1-3% in different datasets is mentioned.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. Experiments compute resources

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: The paper states that a single Nvidia RTX A6000 GPU with 48GB of RAM was used to run the experiments.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. Code of ethics

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics <https://neurips.cc/public/EthicsGuidelines>?

Answer: [Yes]

Justification: Yes, the conducted research preserves anonymity and follows all of NeurIPS Code of Ethics.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. Broader impacts

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: The proposed LoMix is an unparalleled solution for improved high-fidelity medical diagnostics. The paper discusses potential societal impacts on the medical imaging community in the abstract, introduction, and conclusion. There is no known negative societal impact of the proposed research.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. Safeguards

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: The paper poses no such risks as it focuses on fundamental research on improving medical image segmentation.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. Licenses for existing assets

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: All open source datasets and models used are properly cited and follow the license and terms of those datasets and models.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.

- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [NA]

Justification: No new assets are introduced in the paper.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: The paper does not involve crowdsourcing or any research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: The paper does not involve crowdsourcing or any research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: The paper uses the LLM only for writing, editing, or formatting purposes.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (<https://neurips.cc/Conferences/2025/LLM>) for what should or should not be described.

A Appendix / Supplementary Material

A.1 Training Algorithm

Algorithm 1 outlines the training procedure for our U-shaped segmentation network with the LoMix module. We train end-to-end with AdamW [17], updating both the usual network parameters and the loss weight parameters α . Pseudocode is given in Algorithm 1 below.

During training, the network learns to minimize the total loss by improving all logits. As training progresses, the $\text{softplus}(\alpha)$ weights will adjust – for example, if the attention-fused output consistently yields lower error than others, its corresponding α_{ij}^{awf} may increase, giving it more influence.

In the end, we have multiple trained decoder branches and fusion modules. For inference, one can either use output of the single best-performing decoder stage or combine the decoder outputs via one of the fusion strategies (e.g., attention fusion or a simple average) to produce the final segmentation.

Thanks to our LoMix supervised training, all these outputs are optimized to be accurate and complementary. The result is a segmentation model that embodies an ensemble of experts, trained jointly in a principled manner to maximize the segmentation quality.

A.2 Advantages of LoMix

Our LoMix-enhanced segmentation has several notable advantages:

- **Interpretability:** The learned weights $\text{softplus}(\alpha_i)$ and $\text{softplus}(\alpha_S^{\text{op}})$ provide insights into the importance of each decoder output and fusion type. For instance, if the addition-fusion outputs receive a high weight, it means that the simple aggregated prediction is consistently effective; if a particular logits’s weight drops close to zero, then the system concludes that that particular logit is not helpful. Examining these weights can thus reveal which combinations of predictions the model finds most useful, thus offering a peek into the ensemble strategy learned by the network.
- **Adaptability:** Because the loss weights are learned, the framework adapts to different datasets and scenarios. The model can allocate more weight to certain predictions if the data benefit from that fusion. For example, on data where one decoder stage consistently outperforms the others, the learning process can put more emphasis to that decoder stage (and possibly its fused outputs) to optimize performance. Conversely, if all decoders are needed (say each captures a different class or scale), then the weights can remain distributed.

Algorithm 1 Training with Logits Mixing (LoMix)

Input: Labeled image dataset $\mathcal{D} = \{(I_n, Y_n)\}_{n=1}^N$; number of decoder stages L ; network parameters Θ (encoder, decoders); fusion-weight scalars $\{\alpha_u\} \leftarrow 0$ for every original or mutant logit P_u

Output: Optimized model Θ and fusion-weight scalars $\{\alpha_u\}$

```

1: Initialize optimizer  $\mathcal{O}$  for  $(\Theta, \{\alpha_u\})$ 
2: while training not converged do
3:   Sample mini-batch  $\{(I_b, Y_b)\}_{b=1}^B \subset \mathcal{D}$ 
4:    $\{P_\ell\}_{\ell=1}^L \leftarrow f_\Theta(I_b)$  ▷ upsampled stage logits
5:    $\mathcal{P}_{\text{orig}} \leftarrow \{P_1, \dots, P_L\}, \mathcal{P}_{\text{mut}} \leftarrow \emptyset$ 
6:   for all non-empty subsets  $S \subseteq \{1, \dots, L\}$  with  $|S| \geq 2$  do
7:     Generate fused logits  $P_S^{(\text{add})}, P_S^{(\text{mult})}, P_S^{(\text{cat})}, P_S^{(\text{awf})}$  via Eqs. 1–5
8:      $\mathcal{P}_{\text{mut}} \leftarrow \mathcal{P}_{\text{mut}} \cup \{P_S^{(\text{add})}, P_S^{(\text{mult})}, P_S^{(\text{cat})}, P_S^{(\text{awf})}\}$ 
9:    $\mathcal{L}_{\text{total}} \leftarrow 0$ 
10:  for all  $P_u \in \mathcal{P}_{\text{orig}} \cup \mathcal{P}_{\text{mut}}$  do
11:     $w_u \leftarrow \text{softplus}(\alpha_u)$  ▷ NAS-inspired, unconstrained
12:     $\mathcal{L}_{\text{total}} \leftarrow \mathcal{L}_{\text{total}} + w_u \mathcal{L}_{\text{seg}}(P_u, Y_b)$ 
13:  Update  $(\Theta, \{\alpha_u\}) \leftarrow \mathcal{O}(\nabla_{\Theta, \alpha} \mathcal{L}_{\text{total}})$ 
14: return  $\Theta, \{\alpha_u\}$ 

```

This adaptability alleviates the need for manual hyperparameter tuning of multi-output losses for each new application.

- **Training Stability and Regularization:** Supervising multiple predictions (original and fused) acts as an implicit regularizer and stabilizer. Indeed, it is less likely that the network will overfit or get stuck in a poor local minimum because each decoder stage is guided by its own loss and by the fused losses that tie all decoder stages together. If one decoder begins to make mistakes, the others (and their combinations) still provide correct feedback, preventing the entire model from drifting off. Additionally, the learned weighting further stabilizes training by reducing the impact of any particularly noisy loss term: if a fused output is extremely erroneous at the start, its weight can adjust downward, preventing it from exploding the gradient. Overall, we observed that LoMix yields faster convergence and more robust training than a single-output or manually deep-supervised counterpart, thanks to these effects.

A.3 Why Softplus in NAS-inspired Weight Learning?

We choose Softplus (see Eq. 6) rather than alternatives like Softmax or explicit normalization for several reasons:

- **No Sum Constraint.** Softmax would constrain the weights to sum to one, coupling them and forcing a distribution over outputs. This would prevent the model from independently suppressing a noisy output (as reducing one weight necessitates increasing others). In contrast, Softplus outputs are independent and unbounded, so each weight can shrink toward zero (effectively ignoring that logit) or grow arbitrarily without affecting the sum of other weights.
- **Strict Positivity.** Softplus guarantees $\alpha > 0$ smoothly, unlike ReLU which could produce exact zeros or sigmoid which would bound weights in $(0, 1)$. Positive weights ensure each loss term contributes non-negatively.
- **Smooth Gradients.** Softplus is smooth and has non-vanishing derivatives for all inputs, which stabilizes learning of the weight parameters. A hard normalization or clipping could yield zero gradients for some ranges.

A.4 Datasets

We evaluate the LoMix’s efficacy across seven datasets covering six segmentation tasks. Our two multi-class segmentation datasets are Synapse Multi-organs¹ and ACDC cardiac organs². The Synapse multi-organ dataset is used for abdominal organ segmentation and includes 30 abdominal CT scans with 3,779 axial slices of 512×512 pixels. Following the TransUNet [3], 18 scans (2,212 slices) are used for training and 12 for validation/testing. We segment eight organs: aorta, gallbladder, left kidney, right kidney, liver, pancreas, spleen, and stomach. For cardiac organ segmentation, the ACDC dataset contains 100 cardiac MRI scans segmented into three sub-organs: right ventricle (RV), myocardium (MYO), and left ventricle (LV). We follow the TransUNet protocol using 70 cases (1,930 slices) for training, 10 for validation, and 20 for testing. Our binary breast cancer segmentation dataset, BUSI [1] contains 647 images: 437 benign and 210 malignant. Our skin lesion segmentation dataset. Our three polyp segmentation datasets are Kvasir [13] (1,000 images), ClinicDB [3] (612 images), CVC-ColonDB [29] (379 images), and ETIS-LaribPolypDB [28] (196 images). Furthermore, we use ISIC2018 [6] (2,594 images) for skin lesion segmentation. We use 80% of the data for training, 10% for validation, and 10% for testing in BUSI, Kvasir, CVC-ColonDB, ETIS-LaribPolypDB, and ISIC2018 datasets.

A.5 Dataset Specific Implementation Details

For multi-class segmentation in Synapse Multi-organs and ACDC datasets, we use an input size of 224×224 , and optimize the combined Cross-entropy ($\beta=0.3$) + DICE ($\gamma=0.7$) loss. We train models for 300 and 400 epochs with a batch size of 6 and 12 for Synapse and ACDC datasets, respectively.

¹<https://www.synapse.org/#!/Synapse:syn3193805/wiki/217789>

²<https://www.creatis.insa-lyon.fr/Challenge/acdc/>

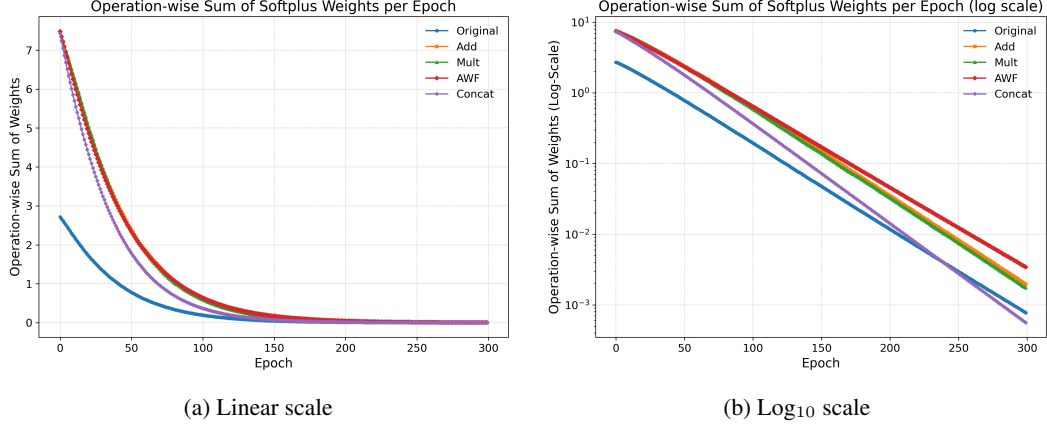


Figure S.1: Operation-wise sum of softplus loss weights during training. Each curve aggregates all logits produced by the same fusion family (Original, Add, Mul, WF, Concat) in the PVT-EMCAD-B2 + LoMix run. (a) softplus weight values in linear scale, (b) the same values in log-scale.

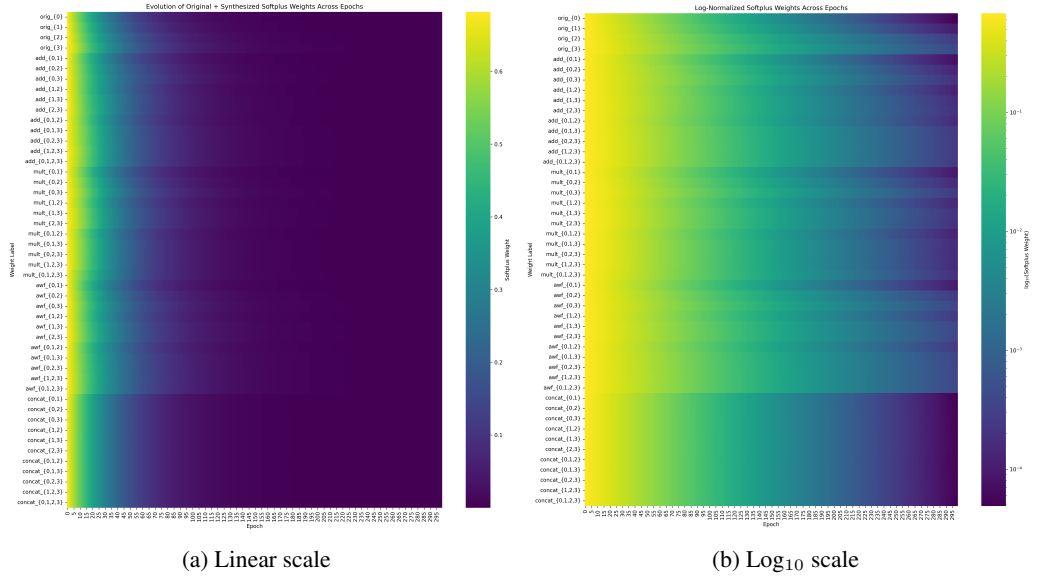


Figure S.2: Evolution of softplus weights over training epochs in Synapse 8-organ segmentation. (a) softplus values in linear-scale, (b) the same data with a logarithmic color normalization, revealing the relative ordering of very small weights and confirming that a few predictions dominate the loss while many others are softly suppressed.

The image dimensions are set to 256×256 pixels for the BUSI and ISIC2018 datasets, while the image dimensions are set to 352×352 pixels for the polyp datasets (Kvasir, CVC-ColonDB, ETIS-LaribPolypDB), respectively. We utilize a multi-scale training approach, with scales of $\{0.75, 1.0, 1.25\}$ and no augmentation. We use a hybrid weighted BinaryCrossEntropy (BCE) with a weighted Intersection over Union (IoU) loss (1:1) and train models for 200 epochs with batches of 16 for BUSI, ISIC2018, Kvasir, CVC-ColonDB, and ETIS-LaribPolypDB. We employ random rotation and flipping as data augmentation methods in all of our experiments except the BUSI dataset. We save the best model based on validation DICE score in all datasets and report DICE score on testsets except the Synapse Multi-organ and ACDC datasets, while the predictions from all four stages are summed together to produce the final segmentation map in the BUSI, ISIC 2018, Kvasir, CVC-ColonDB, and ETIS-LaribPolypDB datasets.

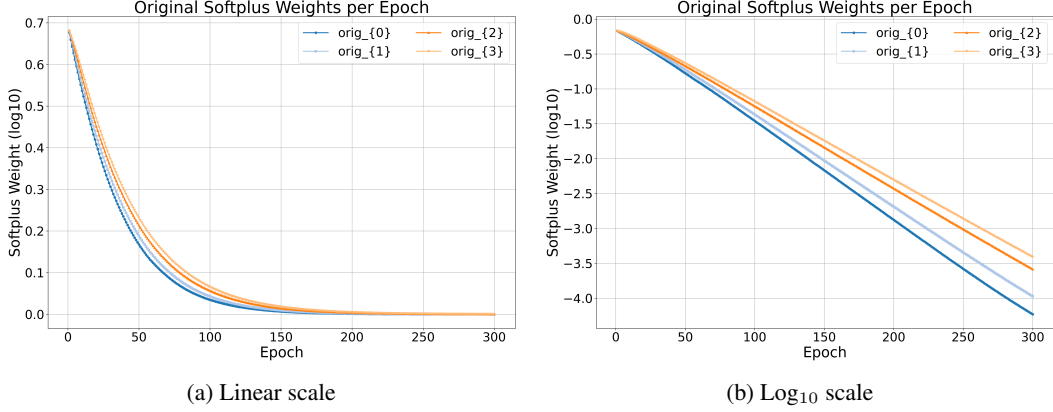


Figure S.3: Evolution of softplus loss weights for the four original logits. Both panels track the same data over 300 training epochs; (a) the left panel plot the softplus values in linear scale; (b) the right plot rescales the y-axis logarithmically to expose tiny values.

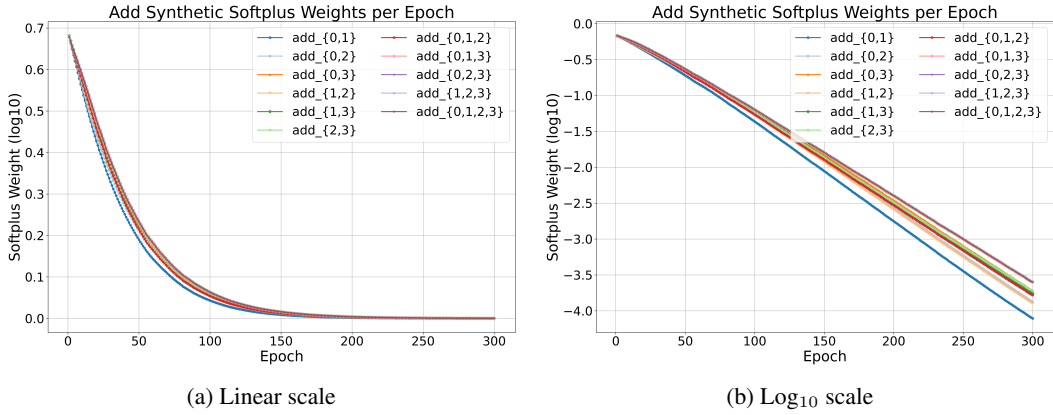


Figure S.4: Per-subset softplus weights for the *addition* operation during training. Curves correspond to every additive mixture of decoder logits. Both the (a) linear- and (b) log-scaled plots reveal how the NAS-inspired optimization reallocates loss weight across epochs.

A.6 Weight Dynamics and Interpretability

Figure S.1 shows that LoMix reallocates supervision away from the four original logits toward the far larger pool of fused logits. By epoch 50, the original logits hold $< 10\%$ of the total weight, while the learnable fusions, especially attention-weighted fusion (AWF) and multiplication (Mult) remain dominant for the next 250 epochs. Addition and concatenation also preserve non-trivial weight, illustrating that every operator family contributes useful gradients. The log-scale panel highlights near-perfect exponential decay for all groups, with parallel slopes indicating that LoMix balances them proportionally rather than suppressing any single operator outright. Together, these curves confirm that the NAS-inspired optimization discovers a nuanced, multi-operator loss distribution in which learnable fusions drive most of the training signal, yet fixed arithmetic fusions and even the original heads are still retained to provide complementary guidance.

Figure S.2 visualizes the softplus loss weights that LoMix learns for *all* 48 supervised logits (4 originals + 44 fusions) over 300 epochs. In the linear-scale map (Figure S.2a), the vast majority of weights collapse to closer to 0 after 50 epochs, while a small cluster—mainly the finest (last layer) original logit and a few mutants created by **AWF**, **Add**, **Mul**, and **Concat**—retain higher weights. The log-scale view (Figure S.2b) makes the hierarchy clearer: the finest grain and attention-weighted fusion (awf) maps dominates over other original or synthetic maps. Thus, the differentiable search automatically reduces 48 supervision maps down to a compact, high-impact subset while discarding redundant logits.

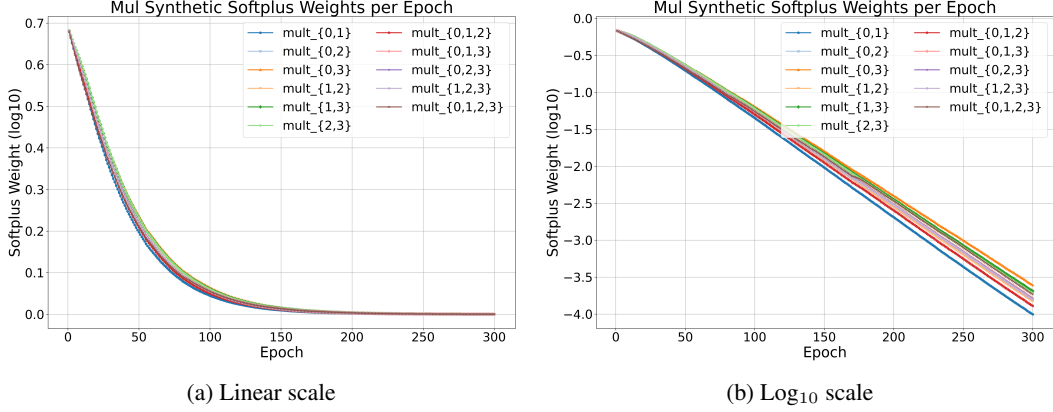


Figure S.5: Per-subset softplus weights for the *multiplication* operation during training. Curves correspond to every multiplication mixture of decoder logits. Both the (a) linear- and (b) log-scaled plots reveal how the NAS-inspired optimization reallocates loss weight across epochs.

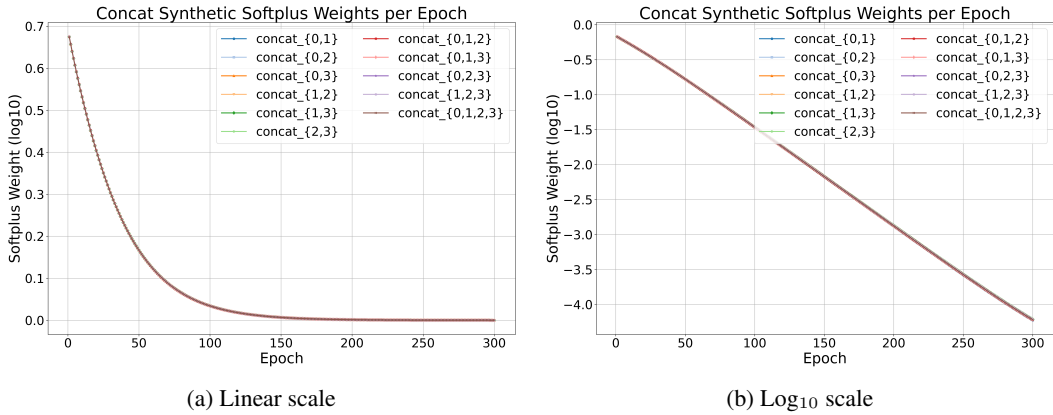


Figure S.6: Per-subset softplus weights for the *concatenation* operation during training. Curves correspond to every concatenation mixture of decoder logits. Both the (a) linear- and (b) log-scaled plots reveal how the NAS-inspired optimization reallocates loss weight across epochs.

Original. Figure S.3 shows how LoMix optimizes the weights to four original decoder outputs during training. Both linear-scale (Figure S.3a) and log-scale (Figure S.3b) reveal that the fine-scale heads (*orig₂* / *orig₃*) retaining more influence than the coarser ones during all training epochs. This trend mirrors the full heatmap in Figure S.2: the NAS-inspired optimization quickly reallocates the supervision to a handful of high-utility fused logits, allowing LoMix to concentrate the gradient signal where it is most beneficial.

Addition (Add). Figure S.4 shows that all additive subsets decay smoothly, with the full 4-logit sum (*add_{0,1,2,3}*) holding the largest weight throughout training. This indicates that the optimizer values a coarse, resolution-agnostic blending of logits for global consistency even late into training.

Multiplication (Mult). Figure S.5 demonstrates that multiplicative subsets start at the same magnitude as Add, leaving the 3- and 4-branch products dominant after 150 epochs. This pattern suggests that the optimizer relies on multiplicative interactions primarily when multiple scales jointly agree, using them as a selective gating rather than an expansive fusion.

Concatenation (Concat). All Concat curves in Figure S.6 lie almost on top of each other, revealing that concatenation quickly learns to down-weight raw channel stacks. The uniform and steady decay implies that concatenation contributes mainly in the early epochs to speed convergence, with its influence reducing as finer operators take over.

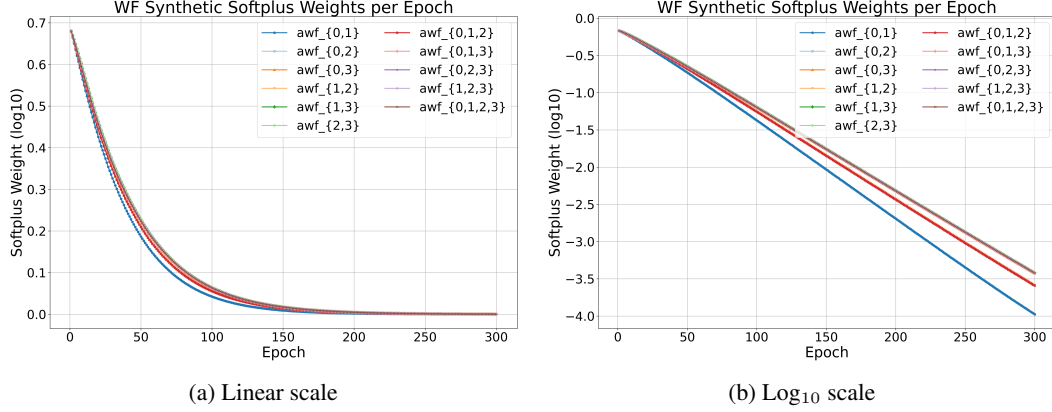


Figure S.7: Per-subset softplus weights for the *attention-weighted fusion* (AWF) operation during training. Curves correspond to every AWF mixture of decoder logits. Both the (a) linear- and (b) log-scaled plots reveal how the NAS-inspired optimization reallocates loss weight across epochs.

Table S1: Results of Synapse 8-organ segmentation. DICE scores (%) are reported for individual organs. Results of UNet, AttnUNet, PolypPVT, SSFormerPVT, TransUNet, and SwinUNet are taken from [24]. Gallbladder (GB), Left kidney (KL), Right kidney (KR), Pancreas (PC), Spleen (SP), and Stomach (SM). $\uparrow$ ($\downarrow$) denotes the higher (lower) the better. ‘—’ means missing data from the source. Results are averaged over five runs. Best results are shown in **bold**.

Methods	Average			Per-organ DICE (%) $\uparrow$							
	DICE (%) $\uparrow$	HD95 $\downarrow$	mIoU (%) $\uparrow$	Aorta	GB	KL	KR	Liver	PC	SP	SM
UNet [25]	70.11	44.69	59.39	84.00	56.70	72.41	62.64	86.98	48.73	81.48	67.96
AttnUNet [19]	71.70	34.47	61.38	82.61	61.94	76.07	70.42	87.54	46.70	80.67	67.66
UNet++ [36]	80.01	28.08	69.91	89.15	70.99	83.37	79.21	94.00	61.23	86.38	75.79
DeepLabv3Plus-R50 [4]	79.37	23.43	69.52	83.84	63.56	85.31	82.29	94.04	59.22	88.61	78.12
SSFormer [33]	78.01	25.72	67.23	82.78	63.74	80.72	78.11	93.53	61.53	87.07	76.61
PolypPVT [7]	78.08	25.61	67.43	82.34	66.14	81.21	73.78	94.37	59.34	88.05	79.4
TransUNet [3]	77.61	26.90	67.32	86.56	60.43	80.54	78.53	94.33	58.47	87.06	75.00
SwinUNet [2]	77.58	27.32	66.88	81.76	65.95	82.32	79.22	93.73	53.81	88.04	75.79
MT-UNet [32]	78.59	26.59	—	87.92	64.99	81.47	77.29	93.06	59.46	87.75	76.81
MISSFormer [12]	81.96	18.20	—	86.99	68.65	85.21	82.00	94.41	65.67	91.92	80.81
PVT-CASCADE [20]	81.06	20.23	70.88	83.01	70.59	82.23	80.37	94.08	64.43	90.10	83.69
TransCASCADE [20]	82.68	17.34	73.48	86.63	68.48	87.66	84.56	94.43	65.33	90.79	83.52
Rolling-UNet-S [16]	74.84	40.07	63.89	85.43	65.45	77.18	71.28	92.50	49.15	85.77	72.00
CMUNet-S [30]	75.20	28.16	64.37	83.96	61.34	77.01	78.04	91.53	51.50	85.36	72.85
EGE-UNet [27]	62.28	51.22	48.83	70.43	51.35	68.28	59.32	86.70	42.25	67.06	52.84
PVT-GCASCAD [22]	83.28	15.83	73.91	86.50	71.71	87.07	83.77	95.31	66.72	90.84	83.58
UNetXt [31]	72.60	30.68	61.30	80.20	60.82	76.13	69.96	91.80	48.04	83.27	70.64
PVT-EMCAD-B0 [24]	81.97	17.39	72.64	87.21	66.62	87.48	83.96	94.57	62.00	92.66	81.22
PVT-EMCAD-B2 [24]	83.63	15.68	74.65	88.14	68.87	88.08	84.10	95.26	68.51	92.17	83.92
PVT-EMCAD-B0 + LoMix (Ours)	82.60 $\pm$ 0.9	16.80	73.44	87.41	68.92	86.67	83.77	95.41	62.92	92.70	83.02
PVT-EMCAD-B2 + LoMix (Ours)	85.07 $\pm$ 1.3	14.85	76.41	88.84	73.51	89.07	84.71	95.76	69.74	92.47	86.47

Attention-Weighted Fusion (AWF). Figure S.7 shows that AWF subsets preserve comparatively higher weights for longer—especially the full 4-logit fusion—remaining above other operators. This persistence shows the optimizer’s strong preference for spatially adaptive weighting, thus confirming AWF’s key role in exploiting complementary decoder resolutions throughout training.

A.7 Learned Loss Weights Comparison of the Best Epoch

At convergence (Figure S.8), the NAS-inspired optimizer concentrates most of the loss weight on attention-weighted fusion (AWF) combinations, followed by a smaller but still meaningful allocation to multiplicative and additive mixes, while concatenation paths receive almost negligible weight. Notably, among the original logits only the fine grain stages (2, 3) receive higher weights, confirming that the network relies primarily on multi-scale mixtures rather than any single head. This distribution echoes our ablation study: the model learns to emphasize operators that can reconcile complementary spatial cues (AWF, Mult, Add), while down-weighting logits that add parameters without clear synergy (Concat), thereby producing the highest DICE performance without manual tuning.

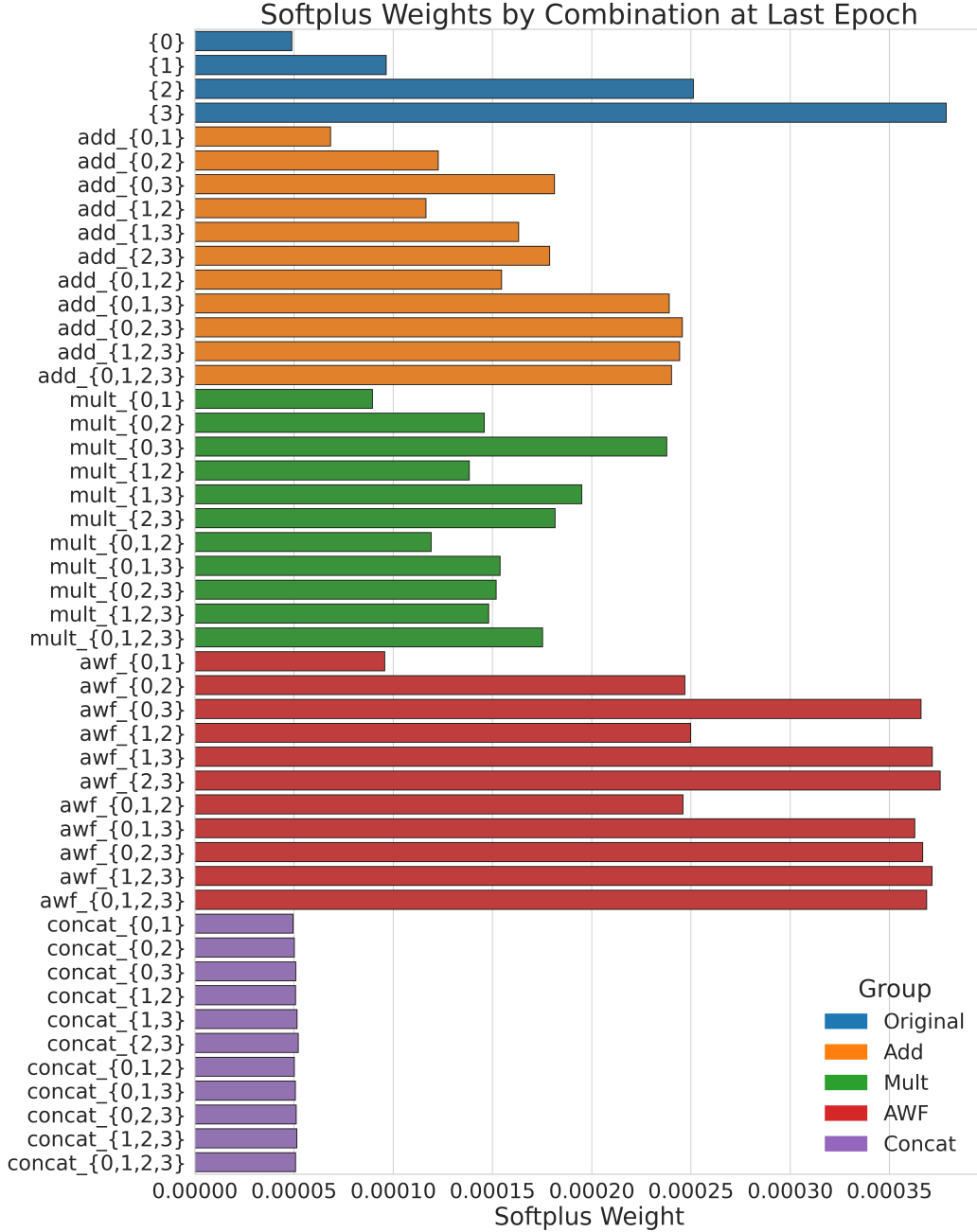


Figure S.8: Learned softplus weights of the **best** LoMix epoch. Each bar corresponds to the final softplus weight assigned to one logit combination, grouped by fusion operator (Original decoders, Addition, Multiplication, AWF, and Concatenation). Longer bars indicate that a combination is more strongly trusted by the learned loss during training.

A.8 Results of Synapse 8-organ Segmentation

Table S1 shows that LoMix achieves clearly superior multi-organ segmentation on Synapse compared to prior CNN- and transformer-based methods. In particular, our LoMix variants attain the highest average DICE, lowest HD95, and highest mIoU of any approach. For example, the best LoMix model (85.07%) outperforms PVT-EMCAD-B2 (83.63%), while also exceeding TransCASCADE (82.68%) and MISSFormer (81.96%). The gains are especially large on small, difficult organs: e.g. LoMix

Table S2: Results of Synapse 13-organ segmentation. DICE scores (%) are reported for individual organs. Gallbladder (GB), Left kidney (KL), Right kidney (KR), Pancreas (PC), Spleen (SP), Esophagus (Eso), Stomach (SM), Inferior Vena Cava (IVC), Portal and Splenic Veins (Veins), Left Adrenal Glands (LAG), and Right Adrenal Glands (RAG). Results are averaged over five runs. Best results per method are shown in bold. LoMix achieves the best average DICE and lowest HD95.

Methods	Average			Per-organ DICE (%) [†]												
	DICE [†]	HD95 [↓]	mIoU [†]	SP	KR	KL	GB	Eso	Liver	SM	Aorta	IVC	Veins	PC	RAG	LAG
Last Layer	67.11	13.96	58.06	90.54	82.79	86.59	66.82	71.33	95.46	80.49	87.51	80.37	64.54	66.00	0.00	0.00
Deep Sup.	70.42	15.20	60.06	90.47	81.83	85.71	67.87	70.40	95.68	82.10	86.84	77.29	65.87	65.41	0.00	46.03
PVT-EMCAD-B2 [24]	76.21	15.56	64.64	91.34	83.40	86.78	68.99	72.49	95.35	84.86	87.42	79.37	67.74	66.86	52.58	53.50
PVT-EMCAD-B2 + LoMix (Ours)	76.90	12.42	65.49	90.24	84.06	87.54	71.08	74.40	95.74	85.68	87.05	80.65	68.31	69.63	51.34	54.07

Table S3: Results of breast cancer and skin lesion segmentation. We reproduce the results of SOTA methods using their publicly available implementations with our 80:10:10 train-val-test splits. The mean DICE scores (%) of testset over five runs are reported. #FLOPs of all the methods are reported for 256×256 inputs. Best results are shown in bold.

Methods	#Params	#FLOPs	BUSI	ISIC2018
UNet [25]	34.53M	65.53G	74.04	86.67
UNet++ [36]	9.16M	34.65G	74.76	87.46
AttnUNet [19]	34.88M	66.64G	74.48	87.05
DeepLabv3+ [4]	39.76M	14.92G	76.81	88.64
PraNet [8]	32.55M	6.93G	75.14	88.46
UACANet [15]	69.16M	31.51G	76.96	88.72
SSFormer-L [33]	66.22M	17.28G	78.76	90.25
PolypPVT [7]	25.11M	5.30G	79.35	90.36
TransUNet [3]	105.32M	38.52G	78.01	89.04
SwinUNet [2]	27.17M	6.20G	77.38	88.66
UNeXt [31]	1.47M	0.57G	74.71	87.78
CMUNeXt [30]	3.15M	7.37G	77.34	87.51
Rolling-UNet-S [16]	1.78M	2.10G	76.38	87.35
PVT-CASCADE-B2 [20]	34.12M	7.62G	79.21	90.41
PVT-EMCAD-B0 [24]	3.92M	0.84G	79.80	90.70
PVT-EMCAD-B2 [24]	26.76M	5.60G	80.25	90.96
PVT-EMCAD-B0 + LoMix (Ours)	3.92M	0.84G	80.47 $\pm$ 1.04	90.77 $\pm$ 0.63
PVT-EMCAD-B2 + LoMix (Ours)	26.76M	5.60G	81.32$\pm$1.21	91.18$\pm$0.72

significantly improves pancreas and gallbladder DICE versus previous models. Moreover, integrating LoMix into lightweight PVTv2 backbones yields consistent boosts: the PVT-EMCAD-B0 + LoMix (82.60%) exceeds the DICE of PVT-B0-EMCAD (81.97%), and similarly the PVT-EMCAD-B2 + LoMix (85.07%) outperforms the PVT-EMCAD-B2 baseline (83.63%). Crucially, LoMix is applied only during training (no extra inference cost), yet consistently pushes the SOTA across all key metrics. The reason behind the performance gain is that LoMix’s adaptive multi-scale logit fusion produces more accurate and robust abdominal organ segmentation, particularly for small, challenging structures such as the pancreas and gallbladder.

A.9 Results of Synapse 13-organ Segmentation

Table S2 shows that integrating LoMix into the PVT-EMCAD-B2 network provides the strongest 13-organ performance reported to date on Synapse. LoMix improves the mean DICE to 76.90%: a gain of +9.8% over single-head supervision and +6.5% over uniform deep supervision, while simultaneously reducing HD95 from 15.56 to 12.42. More importantly, improvements are *not* limited to one or two easy structures. Indeed, LoMix achieves the best DICE in 10 of 13 organs, including difficult small-volume classes such as the gallbladder (+4.3%), esophagus (+3.1%), and portal and splenic veins (+3.8%). It also revives the previously “dead” adrenal-gland predictions, pushing DICE from 0 to 51–54%. Larger, context-driven organs such as liver and spleen see further improvement, and aorta performance remains on par with the best prior result. These gains confirm that LoMix’s learnable, mixed-scale supervision improves both boundary-sensitive and context-dependent structures, thus delivering a uniformly stronger and more anatomically faithful segmentation without introducing any inference-time overhead.

Table S4: Results of polyp segmentation. We reproduce the results of SOTA methods using their publicly available implementations with our 80:10:10 train-val-test splits. The mean DICE scores (%) of testset over five runs are reported. Best results are shown in bold.

Methods	#Params	Kvasir	CVC-ColonDB	ETIS-LaribPolypDB
UNet [25]	34.53M	82.87	83.95	76.85
UNet++ [36]	9.16M	83.36	87.88	77.40
AttnUNet [19]	34.88M	83.49	86.46	76.84
DeepLabv3+ [4]	39.76M	89.06	91.92	90.73
PraNet [8]	32.55M	84.82	89.16	83.84
UACANet [15]	69.16M	90.17	91.02	89.77
SSFormer-L [33]	66.22M	91.47	92.11	90.16
PolypPVT [7]	25.11M	91.56	91.53	89.93
TransUNet [3]	105.32M	91.08	91.63	87.79
SwinUNet [2]	27.17M	89.59	89.27	85.10
UNeXt [31]	1.47M	77.88	83.84	74.03
CMUNeXt [30]	3.15M	78.41	83.25	76.12
Rolling-UNet-S [16]	1.78M	75.93	82.48	73.26
PVT-CASCADE-B2 [20]	34.12M	92.05	91.60	91.03
PVT-EMCAD-B0 [24]	3.92M	91.95	91.71	91.65
PVT-EMCAD-B2 [24]	26.76M	92.75	92.31	92.29
PVT-EMCAD-B0 + LoMix (Ours)	3.92M	92.34±0.96	93.31±0.86	92.74±0.79
PVT-EMCAD-B2 + LoMix (Ours)	26.76M	93.45±0.87	93.98±0.68	93.10±0.96

Table S5: Effect of input resolution on Synapse 8-organ segmentation (↑ higher is better, ↓ lower is better). Each row is averaged over five runs. The best results are shown in **bold**.

Resolution	Average			Per-organ DICE (%)							
	DICE ↑	HD95 ↓	mIoU ↑	Aorta	GB	KL	KR	Liver	PC	SP	SM
224 × 224	85.07	14.85	76.41	88.84	73.51	89.07	84.71	95.76	69.74	92.47	86.47
256 × 256	85.45	12.18	77.05	89.07	72.90	89.26	84.88	95.68	72.65	92.48	86.71
512 × 512	87.25	14.49	79.52	91.58	78.05	88.96	85.73	96.33	76.96	92.41	87.98

A.10 Results of Breast Cancer and Skin Lesion Segmentation

Table S3 shows the evaluation of ultrasound breast-tumour (BUSI) and dermoscopic skin lesion (ISIC2018) benchmarks and again demonstrates that LoMix can improve the DICE score of efficient networks without increasing their computations. When integrated onto the PVT-EMCAD-B2 network, LoMix improves DICE scores by +1.07% on BUSI and +0.22% on ISIC 2018, surpassing heavyweight designs such as DeepLabv3+ and SSFormer-L. In all cases, the gains do not require architectural changes at the test time, confirming that LoMix’s learnable mixed-scale supervision translates into tangible DICE score improvements, even for small, noise-prone medical datasets, without compromising the compactness or efficiency of the underlying model.

A.11 Results of Polyp Segmentation

Table S4 benchmarks LoMix on three challenging colon-polyp datasets against different CNN, transformer, and lightweight hybrid methods. LoMix improves mean DICE of the PVT-EMCAD-B0 network by +0.4-1.6% on Kvasir, CVC-ColonDB, and ETIS-LaribPolypDB, thus outperforming substantially larger models such as DeepLabv3+ and SSFormer-L. Coupling LoMix with the PVT-EMCAD-B2 establishes a new SOTA on all three datasets, thus surpassing the best baseline of PVT-EMCAD-B2 by up to +1.7% while matching its inference parameter count and runtime. Gains are achieved without modifying the network at inference, confirming that LoMix’s learnable mixed-scale supervision improves polyp delineation accuracy and data efficiency without sacrificing the compactness of the underlying architecture.

A.12 Effect of Input Resolution on Synapse 8-organ Segmentation

Table S5 shows that LoMix capitalizes on every pixel it is given. When the input image is enlarged from 224×224 the mean DICE rises from 85.07% to 85.45% while HD95 reduces by 2.7, indicating

Table S6: Comparison of different fusion operation combinations using NAS-inspired Softplus weights and PVT-EMCAD-B2 model for Synapse 8-organ segmentation. Gallbladder (GB), Left kidney (KL), Right kidney (KR), Pancreas (PC), Spleen (SP), and Stomach (SM). $\uparrow$ ($\downarrow$) indicates higher (lower) is better. LoMix achieves the highest average DICE.

Operation	DICE $\uparrow$	HD95 $\downarrow$	mIoU $\uparrow$	Aorta	GB	KL	KR	Liver	PC	SP	SM
AWF	82.91	19.05	73.88	88.30	68.04	87.30	83.13	95.42	65.85	91.39	83.84
(Mult, AWF)	83.45	18.84	74.44	88.00	68.41	86.50	81.96	95.95	69.94	92.43	84.38
(Add,Mult)	83.51	22.99	74.63	88.12	68.35	86.25	83.54	95.24	70.09	90.75	85.74
Mult	83.55	17.47	74.33	88.20	69.34	88.02	83.84	95.06	66.59	91.27	86.09
Add	83.84	20.15	74.84	88.04	73.78	87.77	83.37	95.35	68.47	90.18	83.79
(Add,AWF)	83.95	19.77	75.08	88.38	71.09	86.82	82.82	95.52	69.24	92.36	85.37
(Add,Concat)	83.96	15.74	75.07	88.46	72.06	87.53	84.12	95.24	69.18	90.54	84.52
(Add,Mult,AWF)	84.01	21.62	74.92	89.44	71.40	88.78	84.24	94.98	68.56	91.56	83.11
(Add,Concat,AWF)	84.10	17.42	75.23	87.65	71.09	88.24	84.10	95.88	68.55	91.45	85.86
(Mult, Concat)	84.23	16.25	75.49	88.46	69.21	89.35	82.82	95.31	68.81	92.97	86.89
(Add,Mult,Concat)	84.32	19.14	75.35	88.50	71.79	87.59	83.18	95.02	70.18	92.78	85.48
Concat	84.45	19.33	75.53	88.12	71.12	88.32	84.73	96.05	69.80	91.22	86.23
(Mult,Concat,AWF)	84.61	20.24	75.75	88.83	72.60	87.82	83.58	95.90	70.14	91.56	86.45
(Concat,AWF)	84.71	20.39	76.15	88.94	71.54	87.83	85.40	96.12	69.18	92.11	86.56
LoMix (Ours)	85.07	14.85	76.41	88.84	73.51	89.07	84.71	95.76	69.74	92.47	86.47

crisper boundaries at only a modest memory cost. Doubling the image again to 512×512 unlocks a further leap to **87.25%** DICE score, setting new highs on six of eight organs: gallbladder gains +4.54%, pancreas +7.22%, stomach +1.51%, and even large structures such as aorta and liver surpass 91.5% and 96.3% DICE, respectively. The pattern confirms that LoMix’s learnable multi-scale fusion continues to integrate fine detail without over-fitting, thus scaling with resolution.

A.13 Detailed Results of Fusion Operation Ablation

This section extends Section 5.1 by listing the full numeric results in Table S6. When the search space is restricted to a single operator—e.g., only the attention-weighted fusion (AWF) or only element-wise multiplication—mean DICE remains below 83.6%. Two-operator mixtures raise performance into the mid-84% range, and every additional operator yields a further monotonic gain because the softplus search can explore a richer set of cross-scale interactions. The best three-operator recipe (Concat + Multiply + AWF) reaches 84.71% DICE, but the full LoMix variant, which activates *all four* operators, pushes the average to **85.07%**. Organ-wise scores show that only the complete LoMix combination provides balanced gains across the entire anatomy set, confirming our claim that operator diversity—coupled with NAS-style weight learning—is essential for fully exploiting complementary coarse-to-fine cues.

A.14 Detailed Results of NAS-Inspired Weight Learning

This section extends Section 5.2 by presenting the full numbers in Table S7. Across all five supervision settings, replacing uniform loss weights with our NAS-inspired *softplus* weights provides a clear net win. For plain deep supervision the learned variant improves mean DICE from 82.90% to 83.17% and, more importantly, reduces HD95 by > 4 point, indicating sharper boundaries. The benefit is modest, but consistent for single-operator fusions—+0.21% DICE for Add, +0.70% for Multiply, +0.39% for Concat, and +0.25% for AWF—while simultaneously lowering or matching HD95 in every case. Crucially, when the full operator pool is active, the learned weights unlock LoMix’s headroom: DICE improves from 84.62% to a new high of **85.07%** and HD95 falls by over 3 point, with eight-of-eight organs improving or remaining steady, thus confirming that adaptive re-weighting helps the network emphasize whichever resolutions (and fusions) are most informative for each anatomy. Because weights are pruned scalars after training, these improvements come at *zero* inference cost.

A.15 Detailed Results of Different Supervision Across Backbones

This section augments Section 5.3 with the full numbers in Table S8. Across all five backbones—including three transformer variants (PVT_V2_B0/B1/B2) and two purely-convolutional networks (ResNet18/34)—our LoMix shows the supervision hierarchy: *Last-Layer* < *Deep Supervi-*

Table S7: Effect of NAS-inspired Softplus weight learning on Synapse 8-organ segmentation with PVT-EMCAD-B2. Fixed = uniform loss weights of 1, Learned = our NAS-inspired softplus weights. Gallbladder (GB), Left kidney (KL), Right kidney (KR), Pancreas (PC), Spleen (SP), and Stomach (SM). $\uparrow$ ($\downarrow$) denotes higher (lower) is better. Results are averaged over five runs. The best setting for each fusion is shown in bold.

Fusion	Scheme	Average			Per-organ DICE (%) $\uparrow$							
		DICE $\uparrow$	HD95 $\downarrow$	mIoU $\uparrow$	Aorta	GB	KL	KR	Liver	PC	SP	SM
Deep Sup.	Fixed	82.90	19.70	73.84	87.43	67.80	87.66	83.75	95.18	65.63	91.53	84.19
	Learned	83.17	15.35	74.23	87.29	70.13	87.17	83.74	95.52	67.66	90.45	83.40
Add	Fixed	83.63	15.68	74.65	88.14	68.87	88.08	84.10	95.26	68.51	92.17	83.92
	Learned	83.84	17.15	74.84	88.04	73.78	87.77	83.37	95.35	68.47	90.18	83.79
Multiply	Fixed	82.85	19.16	73.37	88.61	67.19	86.62	82.85	94.93	68.09	90.27	84.21
	Learned	83.55	17.47	74.33	88.2	69.34	88.02	83.84	95.06	66.59	91.27	86.09
Concat	Fixed	84.14	17.77	75.15	87.11	70.51	88.25	83.70	95.71	68.27	92.70	86.86
	Learned	84.45	19.33	75.53	88.12	71.12	88.32	84.73	96.05	69.80	91.22	86.23
AWF	Fixed	82.66	20.56	73.44	88.21	69.69	87.48	81.96	95.05	67.36	89.20	82.34
	Learned	82.91	19.05	73.88	88.30	68.04	87.30	83.13	95.42	65.85	91.39	83.84
LoMix	Fixed	84.62	18.08	75.72	88.82	72.35	87.72	84.10	95.56	70.46	91.87	86.08
	Learned	85.07	14.85	76.41	88.84	73.51	89.07	84.71	95.76	69.74	92.47	86.47

Table S8: Comparison of different supervision schemes on Synapse 8-organ segmentation across five backbones. LoMix uses NAS-inspired softplus weighting. Gallbladder (GB), Left kidney (KL), Right kidney (KR), Pancreas (PC), Spleen (SP), and Stomach (SM). Results are averaged over multiple runs. Best result per model is **bolded**.

Model	Scheme	Average			Per-organ DICE (%) $\uparrow$							
		DICE $\uparrow$	HD95 $\downarrow$	mIoU $\uparrow$	Aorta	GB	KL	KR	Liver	PC	SP	SM
PVT_V2_B0	Last Layer	75.72	19.95	66.69	85.78	66.16	84.98	80.10	94.08	30.29	88.70	75.71
	Deep Sup.	81.38	21.88	71.66	86.01	65.18	87.31	83.03	94.23	65.14	90.63	79.52
	LoMix (Ours)	82.43	17.32	73.10	87.42	68.10	86.84	83.18	95.35	62.96	91.84	83.76
PVT_V2_B1	Last Layer	79.77	27.30	69.85	87.15	65.95	82.84	79.83	94.26	60.54	89.77	77.80
	Deep Sup.	82.23	25.46	72.60	85.54	69.00	85.46	81.02	95.42	67.90	90.86	82.68
	LoMix (Ours)	83.64	21.05	74.52	88.20	69.57	88.22	83.64	95.04	69.90	90.58	84.00
PVT_V2_B2	Last Layer	80.94	22.89	71.18	87.14	68.00	84.87	81.10	94.63	63.08	89.82	78.86
	Deep Sup.	82.90	19.70	73.84	87.43	67.80	87.66	83.75	95.18	65.63	91.53	84.19
	LoMix (Ours)	85.07	14.85	76.41	88.84	73.51	89.07	84.71	95.76	69.74	92.47	86.47
ResNet18	Last Layer	70.25	27.26	60.24	80.79	59.12	81.06	78.13	91.13	15.68	88.10	67.94
	Deep Sup.	79.80	20.63	69.74	85.95	67.13	84.65	81.06	93.72	57.72	89.72	78.49
	LoMix (Ours)	82.12	20.37	72.54	87.03	70.89	83.60	80.96	94.57	65.47	92.42	82.04
ResNet34	Last Layer	75.50	33.65	65.57	86.05	63.30	83.17	79.74	93.20	37.70	86.96	73.87
	Deep Sup.	81.07	18.04	71.38	85.78	68.13	86.78	82.75	94.06	60.93	90.10	80.06
	LoMix (Ours)	82.91	15.31	73.61	87.08	73.61	87.67	83.77	94.36	66.54	91.55	78.73

sion < **LoMix**. Switching from single-head training to LoMix improves the mean DICE by +6.71% on the lightweight PVT_V2_B0, by +4.13% on the large PVT_V2_B2, and by a striking +11.87% on ResNet18, while simultaneously reducing HD95 by 2.63–18.34. Organ-wise scores echo this trend: LoMix delivers the best DICE for most classes on every backbone, with the sharpest jumps on the most scale-sensitive structures (GB, PC, SM) yet still improving saturated organs such as aorta and liver towards the performance ceiling. These consistent architecture-agnostic gains confirm its value as a plug-and-play supervision method for both CNN and transformer networks.

A.16 Cross-dataset evaluation on polyp segmentation

Clinical deployment demands robustness across scanners and sites. Our evaluation has already addressed overfitting risk by spanning multiple, heterogeneous public datasets and modalities (i.e., abdominal CT, cardiac MRI, breast ultrasound, dermoscopy, colonoscopy), each collected with different protocols and devices. Yet LoMiX improves on every dataset without manual tuning, because it actually behaves like a regularizer. In fact, LoMiX implicitly ensembles diverse multi-scale logits only during training, thus reducing the chance of overfitting to dataset-specific biases.

Table S9: Cross-dataset/hospital generalization of LoMiX. Using the PVT-EMCAD-B2 network, all models are trained for 200 epochs on the Kvasir polyp-segmentation training set (900 images); the epoch with the best DICE (%) on the Kvasir validation split (100 images) is saved. Then we evaluated the generalizability on three external test sets (CVC-ClinicDB, CVC-ColonDB, ETIS-LaribPolypDB), without further tuning.

Methods	CVC-ClinicDB	CVC-ColonDB	ETIS-LaribPolypDB
Last Layer	80.59	75.31	71.64
Deep Supervision	81.87	76.16	75.84
MUTATION	81.88	76.39	75.97
LoMiX (Ours)	83.08	77.70	77.01

Table S10: MedNeXt’s [26] performance on 3D Synapse 8-organ segmentation with Last Layer (LL), Deep Supervision (DS), MUTATION, and LoMiX. DICE scores (%) reported for Gallbladder (GB), Left kidney (KL), Right kidney (KR), Pancreas (PC), Spleen (SP), and Stomach (SM).

Methods	Avg. DICE	Avg. HD95	Aorta	GB	LK	RK	Liver	PC	SP	SM
MedNeXt-M_K3 + LL	86.22	6.62	91.96	75.59	87.11	85.14	96.81	78.31	91.26	83.61
MedNeXt-M_K3 + DS	86.63	8.31	91.61	78.91	90.40	85.94	94.72	74.95	92.10	84.43
MedNeXt-M_K3 + MUTATION	86.84	6.04	91.72	79.94	90.97	86.62	96.58	76.65	90.55	81.69
MedNeXt-M_K3 + LoMiX (Ours)	87.19	4.84	91.81	79.87	90.54	86.65	96.68	76.95	90.63	84.37

To make this explicit, we include a cross-dataset experiment (e.g., train on one dataset/hospital, and test on another) as shown in Table S9. LoMiX produces the highest DICE on every dataset which confirm its superior ability to generalize across hospitals and acquisition devices.

A.17 3D Feasibility and Results of Different Supervision

LoMiX is dimension-agnostic: it operates on C-channel class logit maps at the loss level. Extending LoMiX to 3D is straightforward: can be done simply replacing the 2D convolutions and bilinear upsampling with 3D convolutions and trilinear upsampling, everything else remains unchanged. The fusion/weighting logic remains identical, and the cost still scales with the (small) number of decoder stages (rarely exceeding five stages). When GPU memory is low, either in 2D or 3D, standard optimizations such as gradient checkpointing or caching logits on CPU further reduce compute/memory requirements without altering the algorithm, thus keeping LoMiX practical on modest hardware (< 5 GB extra GPU memory required for backpropagation to process a $96 \times 96 \times 96$ volume with a four-stage network and 9 output classes).

To show the feasibility of LoMiX with 3D networks, the new preliminary results of 3D MedNeXt [26] with LoMiX are reported in Table S10. Our results (in bold) demonstrate that LoMiX achieves the best average DICE and HD95 scores among all supervisions.