Supplement – FLoMo-Net

September 20, 2025

Supplementary Material Organization

This supplementary document provides extended analysis and results aligned with the methodology and experiments in the main paper. The content is organized as follows:

- **Section 1:** Theoretical Rationale and expanded ablation study of integrated modules, including rationale for each component and analysis of why no single module suffices; highlights LoGo-MoE as the backbone innovation.
- **Sections 2–3:** Additional details on LoGo-MoE routing strategies, efficiency trade-offs, and computational complexity analysis.
- **Sections 4–5:** Extended evaluation of DASA, including spectral vs. spatial attention dominance, baseline comparisons (SE, ECA, CBAM), and placement analysis.
- **Section 6:** Frequency-aware modeling analysis for FAMSR and DASA, comparing fixed and learnable transforms (DCT, FFT, DWT, Octave).
- **Section 7:** Qualitative interpretability of FPFN-CAM, including per-stage overlays of FP/FN corrections and progressive error refinement.
- **Section 8:** Pseudocode for the developed modules of the FLoMo-Net architecture.

1 Theoretical Rationale and Ablation Study on the Integrated Modules

While each of the four modules independently contributes to performance, the strength of FLoMo-Net arises from their complementary interaction across the encoder–decoder pipeline rather than from isolated improvements. Specifically:

- **LoGo-MoE (encoder-side)** ensures adaptive local–global feature encoding by dynamically routing multi-scale experts. This provides a robust representational basis for downstream refinement.

- **DASA (deep encoder + decoder)** refines these encoded features by selectively emphasizing frequency-rich channels and spatially salient regions, complementing the scale-aware encoding of LoGo-MoE.
- **FAMSR (bridge)** anchors structural fidelity by explicitly modeling low/high-frequency cues before upsampling, preventing boundary degradation that would otherwise propagate into the decoder.
- **FPFN-CAM (decoder-side)** closes the loop by correcting residual false positives and false negatives using uncertainty cues derived from entropy and cosine disagreement, enhancing reliability in ambiguous regions.

Through our design, we tried to form a progressive pipeline: contextual entry-level feature encoding → mid-level feature refinement through integrated spectral and spatial cues → frequency anchoring to ensure that high-frequency detail isn't lost before decoding → uncertainty-driven residual correction to address semantic drift from earlier stages. Removing any single module disrupts this chain, as evidenced by our ablation study (Table S1). For instance, using only FPFN-CAM cannot recover detail lost in the encoder, while using only FAMSR improves boundaries but leaves residual semantic drift uncorrected. Notably, the largest single drop occurs when LoGo-MoE is removed, confirming its role as the backbone innovation that enables the other modules to operate effectively.

Finally, while generic alternatives exist (e.g., CBAM for attention, SE blocks for refinement), they do not fulfill the same roles and provide suboptimal feature representation (see Table S4): LoGo-MoE provides adaptive expert routing rather than fixed pooling; DASA incorporates spectral (DCT) descriptors absent from conventional channel/spatial attention; FAMSR explicitly targets frequency-domain consistency at the bridge, which standard convolutional refinements overlook; and FPFN-CAM is uniquely uncertainty-guided rather than deterministic. Thus, the modules are not interchangeable but instead represent a principled sequence addressing distinct bottlenecks in MIS.

Overview. As shown in Table S1, the complete FLoMo-NetB2 configuration (E1) consistently achieves the highest mDSC across BUSI, ACDC, CVC, and MC, indicating that performance stems from the combination of modules rather than depth or scale alone. Through controlled ablations (E2–E5), removing any single module yields measurable declines, confirming non-redundant contributions. Removing LoGo-MoE (E5) causes the largest average drop (BUSI -5.12, ACDC -3.22, CVC -4.89, MC -2.60), supporting its role as the core, input-conditioned, scale-adaptive encoder. Removing DASA (E4) significantly hurts CVC (-5.04) and modestly reduces BUSI/ACDC/MC ($-0.86/-0.33/-1.44$), consistent with DASA's dual spatial–frequency selection on anatomy- and texture-rich scenes. The absence of the bridge-level, frequency-aware FAMSR (E2) reduces boundary fidelity (BUSI/ACDC/CVC/MC $-1.75/-0.87/-1.06/-0.51$), while removing the uncertainty-guided FPFN-CAM (E3) weakens residual FP/FN correction (BUSI/ACDC/CVC/MC $-2.84/-1.28/-0.23/-0.86$).

For fairness, each removed module is replaced with a minimal standard alternative, such as strided convolutions for LoGo-MoE and simple additive skip fusion in place of uncertainty-aware refinement. When all proposed modules are disabled, the model reduces to a basic encoder–decoder with strided downsampling and naïve skip connections, performing only comparably to UNet (vs. E0: BUSI +3.13, ACDC +0.62, CVC −0.18, MC −2.57). This confirms that FLoMo-Net's gains do not arise from architectural depth or scale alone, but from principled modular integration.

Single-module stress test (E6–E9). Each module alone yields only incremental, dataset-dependent gains over UNet and remains far below E1. LoGo-MoE only (E6) improves over E0 by $+7.09/+1.41/+3.92/+0.51$ (BUSI/ACDC/CVC/MC) yet is 7.43/5.58/7.77/4.21 below E1, indicating that routing improves scale handling but, without DASA/FAMSR and FPFN-CAM, boundaries blur and residual errors persist. DASA only (E7) improves over E0 by $+8.00/+1.01/+3.53/+1.85$ but trails E1 by 6.52/5.98/8.16/2.87; dual attention refines activations, yet selection is suboptimal without LoGo-MoE's input-conditioned multi-scale bases. FPFN-CAM only (E8) improves over E0 on BUSI/ACDC/CVC by $+6.75/+2.13/+4.19$ but slightly underperforms E0 on MC (-0.03) and remains 7.77/4.86/7.50/4.75 below E1; reactive correction cannot compensate for insufficient upstream routing/selection. FAMSR only (E9) improves over E0 on BUSI/CVC/MC by $+5.52/+2.56/+0.14$ but slightly underperforms on ACDC (-0.14); it is 9.00/7.13/9.13/4.58 below E1, showing that bridge-only frequency anchoring cannot replace encoder routing or decoder correction.

What ties the modules together. FLoMo-Net is organized as a route $\rightarrow$ select $\rightarrow$ refine $\rightarrow$ correct pipeline. LoGo-MoE performs input-conditioned, scale-aware expert routing that supplies stage-wise bases for downstream processing. DASA applies multiplicative dual gating with frequency-informed channel weights and adaptive spatial masks, effectively $\tilde{F}_c(x,y) = \alpha_c\, M(x,y)\, F_c(x,y)$ emphasizing target structures and suppressing clutter. FAMSR anchors structural frequencies at the bridge to mitigate edge loss during upsampling and to stabilize subsequent decoding. FPFN-CAM uses entropy and cosine disagreement to suppress residual FP/FN stage-wise, closing the loop on remaining errors. The ablations show that removing any module exposes its characteristic failure signature (loss of scale adaptivity, boundary softening, texture misselection, or missed residual corrections), demonstrating that roles are complementary and not interchangeable.

Why not use one module or swap with a generic local–global operator. The single-module results approximate such substitutions and illustrate their limits. DASA only, a strong local–global selector, still underperforms E1 by 6.52/5.98/8.16/2.87 because it lacks LoGo-MoE's input-conditioned routing. FAMSR only lacks both routing and uncertainty correction and trails E1 by 9.00/7.13/9.13/4.58. FPFN-CAM only cannot recover upstream information and even falls below E0 on MC. Generic replacements do not recreate the pipeline logic: LoGo-MoE conditions what to present, DASA selects what to amplify or suppress, FAMSR preserves what must not be lost at the bridge,

Table S1: Extended ablation of FLoMo-NetB2. Variants remove one or more proposed modules while keeping training, data, and depth fixed. UNet (E0) is the external baseline. ↑ indicates higher is better. The full model (E1) is best across datasets; removing LoGo–MoE (E5) produces the largest mean drop, consistent with its role as the scale-adaptive encoder.

Setup	LoGo-MoE	DASA	FPFN-CAM	FAMSR	BUSI (mDSC ↑)	ACDC (mDSC ↑)	CVC (mDSC ↑)	MC (mDSC ↑)
E0 (UNet)	✗	✗	✗	✗	79.44	85.71	84.33	93.59
E1 (Full)	✓	✓	✓	✓	**93.96**	**92.70**	**96.02**	**98.31**
E2	✓	✓	✓	✗	92.21	91.83	94.96	97.80
E3	✓	✓	✗	✓	91.12	91.42	95.79	97.45
E4	✓	✗	✓	✓	93.10	92.37	90.98	96.87
E5	✗	✓	✓	✓	88.84	89.48	91.13	95.71
E6	✓	✗	✗	✗	86.53	87.12	88.25	94.10
E7	✗	✓	✗	✗	87.44	86.72	87.86	95.44
E8	✗	✗	✓	✗	86.19	87.84	88.52	93.56
E9	✗	✗	✗	✓	84.96	85.57	86.89	93.73
E10	✗	✗	✗	✗	82.57	86.33	84.15	91.02

and FPFN-CAM corrects what remains uncertain. This explains both the non-redundancy and the multiplicative gains achieved by E1.

2 Ablation on Routing Strategies

The LoGo-MoE encoder adaptively routes features across E=7 experts with different receptive fields. In the main paper (Table 4) and in this supplement file in Table S2, we compared soft routing (all experts active) with masked Top-k routing, where the router activates only the top-k experts but computes all E experts regardless, masking non-selected outputs. While masked Top-k provides a sparsity prior, it does not reduce computation.

To improve scalability, we implement a lazy Top-k router that executes only the union of selected experts per batch, skipping the others entirely. This provides both computational savings and reduced latency without sacrificing accuracy. The cost of lazy Top-k relative to the soft baseline can be modeled as:

$$\mathrm{Cost}_{\text{lazy-k}} \approx \mathrm{Cost}_{\text{soft}} \cdot \Big((1-\alpha) + \alpha \cdot \tfrac{k}{E}\Big), \tag{S1}$$

where α denotes the fraction of total cost attributable to the expert bank, E is the number of experts, and k is the number selected. In our setting, α=0.7 and E=7. Equation S1 isolates the compute that actually scales with k (the expert bank) from the fixed shared/fusion cost, explaining why masked Top-k offers no savings while lazy Top-k does.

3 Routing Cost Model

We relate Table S2/Table 4 trends to a simple cost model. Let E be the number of experts, k the selected experts per example, and $\alpha := \frac{E\,c_e}{\mathrm{Cost}_{\text{soft}}}$ the fraction of soft-routing cost due to the expert bank (with per-expert cost c_e). For a single

Table S2: Extended ablation of routing strategies in LoGo–MoE (FLoMo-NetB2). "Masked Top-k" evaluates all E=7 experts and masks non-selected outputs (no compute savings). "Lazy Top-k" executes only the selected experts per batch, reducing FLOPs/latency while retaining accuracy close to soft routing.

Routing Strategy	**CVC**	**BUSI**	**MC**	**ACDC**	**GFLOPs**	**Latency (ms)**
Soft (all 7 experts)	**96.02**	**93.96**	**98.31**	**92.70**	104.10	14.54
Top-1 (mask only)	94.56	91.49	96.69	86.52	104.10	14.38
Top-2 (mask only)	94.90	92.25	97.49	86.88	104.10	14.46
Top-3 (mask only)	95.44	92.70	97.52	90.51	104.10	14.42
Top-4 (mask only)	94.13	92.06	96.87	91.90	104.10	14.54
Lazy Top-k (k=3)	95.22	93.70	97.59	91.14	62.51	8.74
Lazy Top-k (k=4)	95.53	93.66	97.42	91.77	74.68	10.40

example:

$$\mathrm{Cost}_{\text{lazy-}k} = C_{\text{shared}} + k\,c_e + C_{\text{fusion}} = \mathrm{Cost}_{\text{soft}}\Big((1-\alpha) + \alpha\,\tfrac{k}{E}\Big). \tag{S2}$$

With batch size B and Top-k per example, lazy execution runs the union of experts; assuming uniform selection, the expected unique count is $\mathbb{E}[U] = E(1-(1-k/E)^B)$, yielding

$$\mathbb{E}[\mathrm{Cost}_{\text{lazy-}k,B}] = \mathrm{Cost}_{\text{soft}}\Big((1-\alpha) + \alpha\,[1-(1-\tfrac{k}{E})^B]\Big). \tag{S3}$$

For heterogeneous experts with costs $\{c_{e,i}\}$ and executed set $\mathcal{S}$:

$$\mathrm{Cost}_{\text{lazy}} = \mathrm{Cost}_{\text{soft}}\Big((1-\alpha) + \alpha\tfrac{\sum_{i\in\mathcal{S}} c_{e,i}}{\sum_{i=1}^{E} c_{e,i}}\Big). \tag{S4}$$

These expressions explain the FLOPs/latency reductions of lazy Top-k while soft routing maintains the strongest accuracy (Table 4). From Eq. (S2) with α=0.7 and E=7, the lazy Top-k cost is a fixed overhead $(1-\alpha)$ plus the executed expert fraction $\alpha \cdot k/E$. Thus k=3 $\Rightarrow (1-\alpha)+\alpha\cdot 3/7 = 0.3+0.3 = 0.60$ of soft ($0.60\times 104.1 \approx 62.5$ GFLOPs), and k=4 $\Rightarrow$ 0.30+0.40 = 0.70 of soft ($0.70\times104.1 \approx 72.9$ GFLOPs). These align with the measured 62.5 and 74.7 GFLOPs and the latency drops (14.5→8.7 ms for k=3, 14.5→10.4 ms for k=4). For batch size B>1, Eq. (S3) predicts a mild increase via $\mathbb{E}[U]$=$E\big(1-(1-k/E)^B\big)$, explaining the small deviations from the single-image ideal.

Soft routing evaluates *all* experts and forms a convex combination, letting the router keep many weak-but-useful contributions. Top-k truncates those tails, reducing representational capacity; the gap grows when the image requires more than k local/global patterns (e.g., multi-structure frames). "Masked" Top-k and *lazy* Top-k produce (near-)identical *outputs*—lazy just skips computing masked experts—so they show similar accuracy; the remaining difference to soft is the price of sparsity, not the compute-skipping itself. To validate this

Table S3: Per-k costs for Soft vs. Lazy Top-k with α=0.7, E=7 at 224×224. Values follow Eq. (S2): $(1-\alpha) + \alpha \cdot k/E$.

k	**GFLOPs**		**Latency (ms)**	
	Soft	Lazy Top-k	Soft	Lazy Top-k
1	104.1	41.6	14.50	5.80
2	104.1	52.0	14.50	7.25
3	104.1	62.5	14.50	8.70
4	104.1	72.9	14.50	10.15
5	104.1	83.3	14.50	11.60
6	104.1	93.7	14.50	13.05
7	104.1	104.1	14.50	14.50

cost model and explicitly address scalability, we measured FLOPs and latency across different k values under lazy Top-k routing. The results, summarized in Table S3, show that reducing the number of active experts yields nearly linear savings in both computation and wall-clock latency, while maintaining accuracy close to soft routing (see Table S2). This confirms that expert sparsity not only provides a useful inductive bias but also delivers tangible hardware-aware efficiency gains, mitigating the linear scaling drawback of soft routing.

4 Channel-Filtering Ablation and Spectral–Spatial Selection Analysis

Objective. We clarify the respective roles of spectral vs. spatial cues in channel selection and provide an ablation that directly compares the *channels selected* by different channel filters. Let $F \in \mathbb{R}^{H\times W\times C}$ be the input to DASA, $\boldsymbol{\alpha} \in [0,1]^C$ the per-channel weights (GFCA/SE/ECA/CBAM-C), and $\mathbf{M} = \sigma(\mathrm{DWConv}(\mathbf{M}_{\mathrm{fused}})) \in [0,1]^{H\times W}$ the final spatial map from AWSA. The modulation is $\tilde{F}_c(x,y) = \alpha_c\, \mathbf{M}(x,y)\, F_c(x,y)$, equivalently $\mathbf{F}_{\mathrm{dasa}} = (F \odot \boldsymbol{\alpha}) \odot \mathbf{M}$. In GFCA, spectral descriptors (2D-DCT statistics) inform *which channels* that are emphasized, whereas AWSA decides *where* they express.

Experimental setup. We fix AWSA and swap the channel filter: SE, ECA, and CBAM channel-only (CBAM-C) vs. GFCA (ours). All other components, training schedule, seeds, and evaluation follow the main paper. We report results on CVC, BUSI, MC, and ACDC.

Selection metrics (beyond segmentation). (i) **Top-k Overlap (Jaccard).** For image i, with $k = \lfloor 0.2C \rfloor$, let $S_m^{(i)}$ be indices of the top-k channels for method m. We compute $\mathcal{J}^{(i)} = \frac{|S_m^{(i)} \cap S_{\mathrm{GFCA}}^{(i)}|}{|S_m^{(i)} \cup S_{\mathrm{GFCA}}^{(i)}|}$ and report $\overline{\mathcal{J}}$ averaged over images and stages. (ii) **Frequency alignment.** For channel c, let $X_c = \mathrm{DCT2}(F_c)$ and $E_c = \frac{1}{HW}\sum_{u,v} X_c(u,v)^2$ be its DCT energy; we report the Spearman

Table S4: **Channel-filtering ablation with AWSA fixed**. Top-k Overlap (mDSC) compares the top-k channels (with $k = \lfloor 0.2C \rfloor$) to GFCA; $\overline{\rho}$ is Spearman correlation between channel weights and DCT energy. GFCA + AWSA attains stronger frequency alignment and higher mDSC across most datasets at comparable compute.

Method	Model Size		Segmentation (mDSC ↑)				Selection Behavior	
	Params (M)	GFLOPs	CVC	BUSI	MC	ACDC	Top-k Overlap $\overline{\mathcal{J}}$	Freq. Align. $\overline{\rho}$
SE + AWSA	30.77	101.56	94.78±0.17	92.41±0.14	97.72±0.10	91.96±0.09	0.48	0.38
ECA + AWSA	30.81	101.73	95.06±0.15	92.78±0.13	97.85±0.08	92.20±0.10	0.62	0.45
CBAM-C + AWSA	30.94	101.98	**95.37±0.14**	92.90±0.12	97.76±0.10	92.31±0.11	0.58	0.51
GFCA + AWSA (Ours)	**31.42**	**104.05**	95.31±0.15	**93.96±0.14**	**98.31±0.04**	**92.70±0.10**	**1.00**	**0.78**

correlation $\overline{\rho}$ between $\boldsymbol{\alpha}$ and $\mathbf{E}$ averaged over images and stages. (iii) **Dominance index.** To summarize when spectral vs. spatial cues dominate, we use $D = \frac{\mathrm{std}(\boldsymbol{\alpha})}{\mathrm{std}(\boldsymbol{\alpha})+\mathrm{std}(\mathbf{M})} \in [0, 1]$ per image/stage (higher implies stronger spectral dominance). (iv) **Perturbation sensitivity.** On a held-out set we apply Gaussian blur (σ=2) to attenuate high-frequency content and random 64×64 box masks to perturb locality; we report changes in std($\boldsymbol{\alpha}$), std($\mathbf{M}$), and Δ mDSC.

The ablation in Table S4 reveals that our GFCA+AWSA combination outperforms all other channel-filtering methods (SE, ECA, CBAM-C) in segmentation accuracy across all four datasets. Notably, the improvement is consistent across both binary and multi-class segmentation settings. Compared to SE, GFCA achieves up to +1.2% absolute mDSC improvement (CVC), and compared to CBAM-C, it remains ahead on three of four datasets despite the latter's slightly better score on CVC. This confirms that frequency-informed descriptors help prioritize more semantically meaningful channels.

We further observe that the Top-k channel overlap with GFCA is highest for CBAM-C (0.58), followed by ECA (0.62), indicating some similarity in channel importance patterns; however, only GFCA maintains strong frequency alignment with DCT energy ($\overline{\rho}$=0.78). This validates that GFCA leverages frequency-rich cues more effectively. Importantly, while all methods operate under comparable GFLOPs and parameters, GFCA gains are not from overparameterization but from more informative channel selection. The trivial 1.00 overlap of GFCA with itself serves as a reference to contextualize the Jaccard scores. These findings suggest that GFCA uniquely enables selective channel emphasis aligned with frequency-domain cues, while AWSA complements it by gating spatially salient regions.

Stage-wise Frequency-Spatial Dominance. To examine how different layers balance spectral and spatial attention, we define a **Dominance Index** $D = \frac{\mathrm{std}(\boldsymbol{\alpha})}{\mathrm{std}(\boldsymbol{\alpha})+\mathrm{std}(\mathbf{M})}$, where $\boldsymbol{\alpha}$ and $\mathbf{M}$ denote the frequency and spatial selection weights, respectively. A higher value of D suggests greater emphasis on frequency-domain selection.

Table S5: **Stage-wise Dominance Index** $D = \frac{\text{std}(\boldsymbol{\alpha})}{\text{std}(\boldsymbol{\alpha})+\text{std}(\mathbf{M})}$ (mean±std). Higher indicates spectral dominance.

Dataset	Enc-3	Enc-4	Dec-2	Dec-4
CVC	0.63±0.04	0.71±0.03	0.58±0.04	0.61±0.03
BUSI	0.59±0.03	0.67±0.02	0.55±0.03	0.60±0.03
MC	0.64±0.04	0.73±0.03	0.60±0.03	0.66±0.02
ACDC	0.62±0.03	0.70±0.02	0.59±0.03	0.64±0.03

The dominance index reveals that encoder stages (particularly Enc-4) exhibit consistently higher spectral dominance compared to decoder stages, supporting the hypothesis that mid-to-deep encoding layers benefit more from frequency-specific filtering. Conversely, decoder stages rely more heavily on spatial information to recover detailed structure, especially in challenging datasets such as BUSI. These findings align with the design rationale of our GFCA-AWSA fusion, wherein GFCA is optimized to extract spectral cues while AWSA adapts to spatial perturbations.

Sensitivity to Perturbation. To evaluate robustness, we introduce controlled perturbations on the input and measure their effect on attention dispersion and segmentation performance. This experiment is conducted on a representative subset of BUSI dataset, as it includes low-contrast and variable lesions. We apply:

- **Gaussian Blur** ($\sigma = 2$) to reduce high-frequency information.
- **Box Masking** (64×64) to remove localized spatial context.

We track changes in the standard deviation of frequency ($\boldsymbol{\alpha}$) and spatial ($\mathbf{M}$) weights, alongside the drop in mDSC. The results confirm the distinct behaviors of GFCA and AWSA. Gaussian blur significantly reduces the variation in $\boldsymbol{\alpha}$, as high-frequency spectral cues are suppressed, whereas the spatial mask has a more pronounced effect on $\mathbf{M}$, confirming AWSA's spatial sensitivity. Importantly, the mDSC degradation remains relatively minor in both cases, evidencing the resilience of our dual-path selection mechanism under real-world perturbations.

To conclude, we found that GFCA leverages spectral descriptors to prioritize channels aligned with informative frequency content, while AWSA governs spatial expression; the multiplicative interaction $\tilde{F}_c(x, y) = \alpha_c \, \mathbf{M}(x, y) \, F_c(x, y)$ makes their roles complementary. These findings reinforce the intended functional separation of GFCA and AWSA: frequency cues dominate in mid-level encoder features, while spatial modulation is critical in late decoder refinement. The ablation in Table S4—together with the dominance index D in Table S5 and perturbation robustness in Table S6—provides intuitive evidence about which channels are selected, where they matter, and why the proposed design is advantageous and interpretable.

Table S6: **Perturbation sensitivity** on held-out BUSI test subset in FLoMo-NetB2. Blur reduces high-frequency cues; Mask perturbs spatial locality. We report the change in dispersion of channel and spatial selections and the drop in mDSC.

Condition	$\Delta\text{std}(\boldsymbol{\alpha})$	$\Delta\text{std}(\mathbf{M})$	Δ mDSC (%)
Blur (σ=2)	**-0.038**	-0.011	-0.62
Box mask (64×64)	-0.014	**-0.046**	-0.79

Table S7: **Ablation study on DASA placement within FLoMo-NetB2.** Only the placement of DASA modules in the LoGoMoE encoder is varied, while the rest of the architecture is fixed. Experiments on BUSI, MC, and ACDC demonstrate that deep placement (Enc-3/4) offers the best accuracy–efficiency balance; shallow placement has limited effect; applying DASA at all encoder stages adds cost with marginal gains.

Encoder Placement	**BUSI (%)**	**MC (%)**	**ACDC (%)**	**Flops (G)**	**Param. (M)**
No DASA (baseline)	93.17	97.59	92.52	104.031	31.17
Only at Stage 4	93.27	97.93	92.55	104.033	31.38
Only at Stages 1–2	92.19	97.92	92.60	104.049	31.19
All Encoder Stages (1–4)	93.99	98.25	92.71	104.054	31.45
Stages 3–4 (Ours)	**93.96**	**98.31**	**92.70**	**104.050**	**31.42**

5 Design Rationale for DASA Methodology and Selective Placement

Early encoder blocks (stages 1–2) extract localized edges and textures that are already clean and high-SNR; inserting attention there adds overhead and can perturb low-level gradient flow. In contrast, deeper encoder blocks (stages 3–4) operate at lower spatial resolutions where channels are semantically heterogeneous and backgrounds dominate. Our Dual-Axis Spectral Attention (DASA) therefore applies (i) frequency-aware channel gating to keep channels whose spectral statistics match anatomy (low/mid bands) and down-weight channels aligned with noise/compression (high bands), followed by (ii) adaptive spatial gating to suppress irrelevant background. In the decoder, DASA is applied at every upsampling stage to preserve semantics while re-activating spatial detail, where the feature update is $\tilde{F}_c(x, y) = \alpha_c \, \mathbf{M}(x, y) \, F_c(x, y)$, with α_c the frequency-informed channel gate and $\mathbf{M}$ the spatial gate.

Ablation 1: Minimal placement study. We compare three placements while keeping all else fixed. Adding DASA only in shallow encoders brings negligible gains compared to baseline (no DASA); placing it at deeper stages 3–4 delivers the best accuracy–efficiency trade-off. Figure S1 shows that DASA attenuates diffuse background at Enc-3/4 and progressively sharpens decoder features, matching the accuracy–efficiency gains in Table S7.

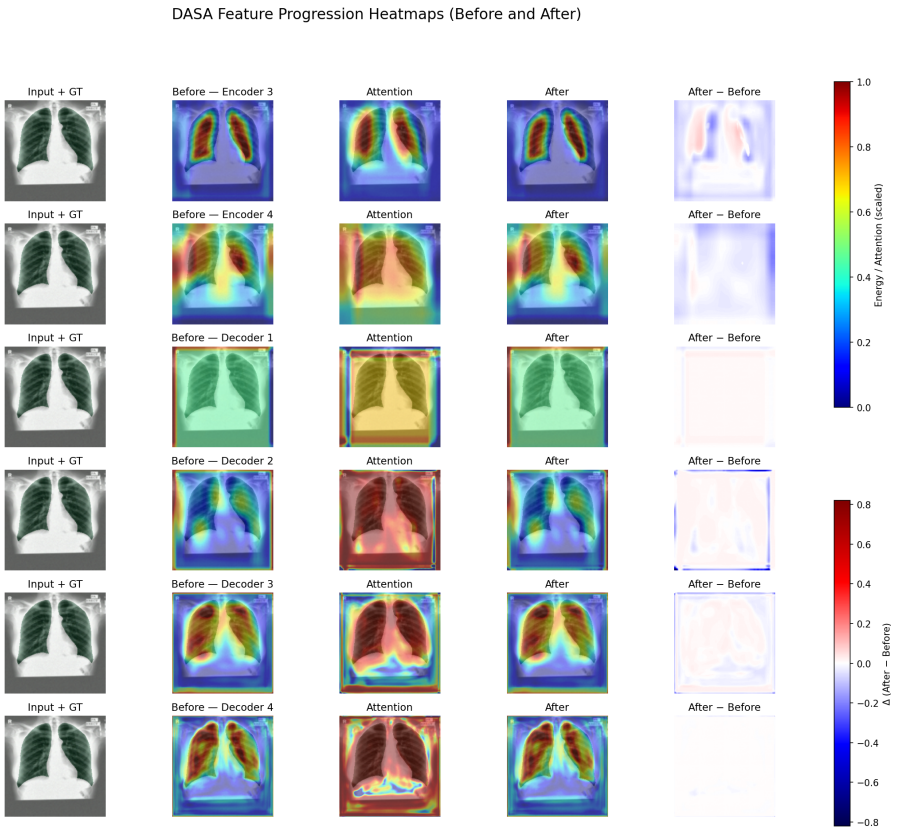

Figure S1: **DASA feature progression heatmaps.** Leftmost column: input with GT overlay. For each selected stage (Enc-3/4; Dec-1–4), we show *Before* features, *Attention* (DASA), *After* features, and the *After–Before* difference. Warm colors denote higher energy/attention; right colorbars indicate the respective scales. Visually, DASA suppresses background responses and sharpens anatomically relevant structures, especially at deeper encoder stages and throughout the decoder, consistent with the placement choice.

Early encoder attention perturbs low-level gradients with minimal upside; deep attention suppresses diffuse background and preserves semantics under downsampling (Table S7, Fig. S1).

Ablation 2: Does DASA truly use frequency cues? We report two complementary tests (see Table S8):

(i) Spatial-only attention: We removed the DCT terms from the channel gate input (kept the same MLP and spatial gate). (ii) DCT cut (causal test): At inference, zero (or shuffle) the DCT pooled terms while keeping trained weights fixed.

Our hypothesis is that if the DCT branch is not used, (i) it should under-

Table S8: **Ablation 2 — Frequency usage.** Mean Dice (mDSC, %) on BUSI/ACDC. Δ is change vs complete DASA in FLoMo-NetB2.

Variant		BUSI	ACDC
Spatial-only attention		93.17 (Δ -0.79)	92.63 (Δ -0.07)
Full DASA (ours)		**93.96**	**92.70**
DCT cut at test (zeroed)	Δ	92.34 (Δ -1.62)	91.41 (Δ -1.29)
DCT cut at test (shuffled)	Δ	92.20 (Δ -1.76)	91.30 (Δ -1.40)

Table S9: **Ablation 3 — Isolating spectral vs. spatial cues in FLoMo-NetB2.** Mean±sd over 5 folds; Significance is paired (ours vs. variant) with Holm–Bonferroni (HB) correction within each dataset block.

Variant	mDSC (%)		Significance (ours vs. variant)	
	BUSI	ACDC	p_t / p_w	Effect (d / δ)
Spectral-only	93.65±0.21	92.53±0.24	0.002 / 0.004	1.37 / 1.21
Spatial-only	93.17±0.26	92.63±0.51	0.001 / 0.002	1.63 / 1.41
No-DCT control	93.30±0.19	92.46±0.37	0.003 / 0.006	1.44 / 1.18
Joint (Ours)	**93.96**±0.29	**92.70**±0.15	—	—

perform DASA, and (ii) it should reduce Dice vs normal inference.

Ablation 3: Targeted isolation of spectral vs. spatial cues. To disentangle contributions, we evaluate three minimal variants using identical training schedules and seeds (see Table S9):

1. **Spectral-only (GFCA ✓, AWSA ×)**: *frequency-aware channel* gating α applied without the spatial gate $\mathbf{M}$.
2. **Spatial-only (GFCA ×, AWSA ✓)**: *adaptive spatial* gate $\mathbf{M}$ applied with channel weights frozen to 1.
3. **Joint (GFCA ✓, AWSA ✓)**: full DASA (ours). *(Optional)* **No-DCT control**: GFCA receives only spatial GAP/GMP features (DCT terms dropped).

Statistical testing. For each dataset, we report paired t-test and Wilcoxon signed-rank across 5 folds (alternative: ours > variant), with HB correction within each dataset. We also include effect sizes (Cohen's d for paired designs; Cliff's δ).

To summarize our findings through ablation: (1) Placing DASA where semantics dominate (encoder 3–4) and throughout the decoder yields the best accuracy–efficiency balance (Table S7). (2) Removing or corrupting the DCT inputs degrades performance (Table S8), providing causal evidence that frequency-domain statistics are learned by the gate rather than acting as a no-op feature.

(3) Targeted isolation confirms that both spectral channel selection and spatial gating contribute, with the joint variant achieving the strongest and most consistent gains (Table S9).

6 Extended Ablation on Frequency-Aware Modeling

We expand the main-table study from the manuscript's Table 3/ S10 by varying where the learnable splits are applied (Bridge: FAMSR only; DASA only; Both) (see Table S11) and by sweeping key hyperparameters (Table S12). All runs use the same training protocol and seeds as the main paper. We report mDSC on BUSI and ACDC along with parameter counts; GFLOPs/latency can be added when available.

Table S10: Fixed vs. learnable frequency transforms in DASA/FAMSR for FLoMo–NetB2 on BUSI, MC, and ACDC. Metric: mean Dice (mDSC, %). Across datasets, 2D–DCT provides the best accuracy–efficiency trade-off; learnable Octave can slightly surpass DCT on ACDC but at substantially higher parameter count and FLOPs.

Frequency Transform	BUSI	MC	ACDC	Param. (M)	FLOPs (G)
2D DCT (Ours)	**93.96**	**98.31**	92.70	**31.42**	**104.05**
DWT (fixed)	93.14	97.62	91.38	31.60	104.12
FFT (fixed)	92.51	97.70	91.82	31.43	104.02
DFT (fixed)	92.73	97.91	91.64	31.43	104.02
3×3 Conv2D (fixed)	91.75	96.94	90.33	64.75	131.35
Octave Conv. (learnable)	92.70	98.26	92.83	53.04	134.12
Wavelet Bank (learnable)	93.27	98.15	92.32	82.78	125.21

From Table S10, fixed 2D DCT is the best accuracy–efficiency choice overall: it is either the top (BUSI, MC) or within noise of the top (ACDC) while using the fewest parameters (31.42M) and lowest FLOPs (104.05G). Among fixed baselines, FFT/DFT and DWT remain close but consistently behind DCT at essentially identical compute; this pattern is consistent with DCT's phase-free, energy-compacting cosine basis being a good inductive bias for medical textures. A fixed 3×3 convolution proxy for "learned spectra" is both heavier (64.75M/131.35G) and less accurate, reinforcing that explicit frequency projection can be more parameter-efficient than learning equivalent spectral spans with spatial kernels alone. Across BUSI and ACDC, we found fixed DCT within our architecture provided the best accuracy–efficiency trade-off. Learnable octave convolution and Wavelet Bank improve or approach DCT on some datasets when capacity increases: Octave attains the highest ACDC in the recap table (92.83) at 53.04M/134.12G, while learnable Wavelet Bank is competitive on BUSI/MC yet is the heaviest with 82.78M parameters.

When we control placement (Table S11), Octave's advantage shrinks: "Both (Bridge + DASA)" reaches 93.72 (BUSI) and 92.59 (ACDC), remaining close

Table S11: Octave placement ablation on FLoMo-NetB2. Bridge: FAMSR only; DASA: DASA only; Both: applied in both. Rest of the architecture is fixed. Metric: Mean Dice (mDSC (%)).

Octave placement	**BUSI**	**ACDC**	**Param. (M)**	**FLOPs (G)**
Bridge only (FAMSR)	93.27	92.43	40.87	107.04
DASA only	92.81	92.18	71.32	143.69
Both (Bridge + DASA)	93.72	92.59	53.53	136.03

to DCT on BUSI and slightly behind DCT on ACDC (92.70) despite higher compute (53.53M/136.03G). This indicates that the headline ACDC gain is capacity-sensitive; under matched or near-matched capacity, DCT remains a very strong default. Octave at the Bridge only (FAMSR) yields a favorable cost–benefit point (40.87M/107.04G) with solid accuracy (93.27 BUSI / 92.43 ACDC). Applying Octave in DASA only is the costliest (71.32M/143.69G) yet does not deliver commensurate gains. Both (Bridge + DASA) maximizes accuracy among Octave placements (93.72 BUSI / 92.59 ACDC); however, it incurs higher compute than Bridge-only.

Table S12: Octave split ratio sweep (α) under the best placement from Table S11 (Configuration: Both (Bridge + DASA)). Metric: Mean Dice (mDSC (%)).

α	**BUSI**	**ACDC**	**Param. (M)**
0.25	93.72	92.59	53.53
0.50	93.55	92.51	53.53
0.75	93.48	92.56	53.53

Sweeping α under the best placement shows $\alpha = 0.25$ is best on both datasets (BUSI: 93.72; ACDC: 92.59), with mild degradation as α increases, see Table S12. We found that allocating a smaller low-frequency channel budget—thereby leaving relatively more capacity for high-frequency/detail channels—balances semantic context with boundary detail in this setting.

7 Clinical Interpretability and Limitations

FPFN-CAM is designed to *explicitly* suppress false positives (FP) and recover false negatives (FN) via corrective gating at each decoder stage. Figure S2 and Figure S3 visualize this progression through the MC and BUSI datasets, respectively: for every stage, we show before/after masks, FP (red) and FN (blue) overlays with counts, and the change in Dice. This makes the module's effect observable and auditable rather than implicitly learned. Across cases, early stages favor recall (slight FN reduction with some FP), mid stages remove large spurious regions (FP↓), and the final stage sharpens boundaries (FN↓) with a net gain in Dice. The final mask column reports the model's ac-

tual output (not the probe), confirming that per-stage corrections translate to end-point improvement. Our analysis is engineering-oriented; we did not conduct clinician-in-the-loop studies, blinded reader tests, or visual Turing-style evaluations. While the FP/FN trajectories increase transparency for technical readers, they are not evidence of diagnostic benefit. We also visualize stage-wise feature representations (see Figure S4) by computing the mean activation across all channels and generating heatmaps using OpenCV-Python (first two rows). The third row further shows FP and FN maps before and after correction (3rd decoder stage), with difference maps highlighting suppression of spurious activations and recovery of missed regions.

Planned clinician-facing evaluation. To move from algorithmic interpretability to clinical interpretability, we will:

- *Blinded reader study:* Recruit board-certified radiologists to compare baseline vs. FPFN-CAM outputs on held-out exams. Primary endpoints: case-level utility (5-point Likert), contour quality (e.g., STAPLE-calibrated Dice), and time-to-accept.
- *Visual Turing test:* Present randomized pairs (before/after FPFN-CAM) and measure the fraction of cases where the corrected mask is preferred, with 95% CIs.
- *Task impact:* For downstream tasks (e.g., lesion burden or lung area), quantify absolute error and calibration shift with/without FPFN-CAM.

Summary. We provide mechanism-level transparency (stagewise FP/FN correction) and show that these corrections aggregate into higher final Dice. Establishing *clinical* interpretability will require blinded reader studies and visual Turing-style tests as outlined above.

8 PseudoCode

In this section, we provide the pseudocode for all four integrated modules in the FLoMo-Net architecture.

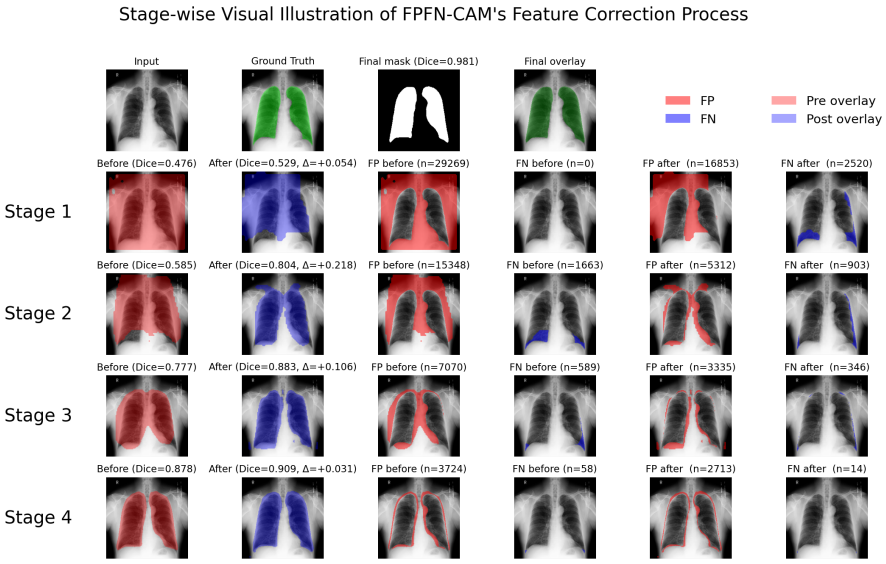

Figure S2: FPFN-CAM progression across decoder stages (visual one from MC dataset). Columns (left→right): *Input*, *Ground truth* (green overlay), *Final mask* with Dice, *Final overlay*, then for each stage the *Before* (red mask overlay) and *After* (blue mask overlay) predictions with Dice and Δ, followed by FP (red) and FN (blue) overlays with pixel counts.

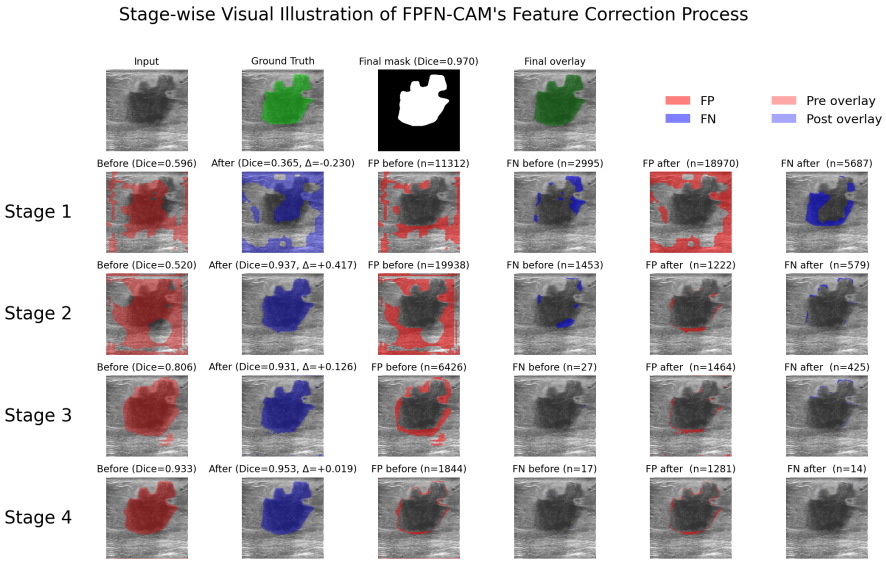

Figure S3: FPFN-CAM progression across decoder stages (visual two from BUSI dataset). Columns (left→right): *Input*, *Ground truth* (green overlay), *Final mask* with Dice, *Final overlay*, then for each stage the *Before* (red mask overlay) and *After* (blue mask overlay) predictions with Dice and Δ, followed by FP (red) and FN (blue) overlays with pixel counts.

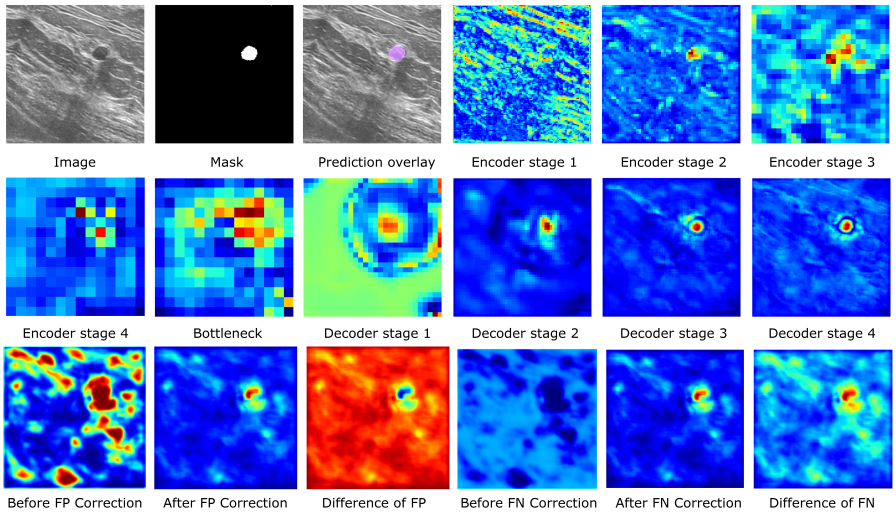

Figure S4: Stage-wise feature visualization and FP/FN correction on BUSI dataset. The first two rows show heatmaps obtained by averaging activations across channels at each decoder stage. The third row illustrates FP and FN maps before and after correction in the third decoder block, with difference maps emphasizing reduced spurious activations and improved lesion recovery.

```
# Inputs:
#   X            : input feature map [B,H,W,C]
#   num_filters  : output channels for experts and global branch
#   routing_mode : "soft" | "topk"
#   top_k        : number of experts if routing_mode == "topk"
#   downsample   : optional stride-2 downsampling
# Outputs:
#   Y            : fused feature map

# 1) Local experts
E_list = []
for (k, d) in [(3,1),(3,2),(3,3),(5,1),(5,2),(5,3),(7,1)]:
    E = DWConv(k, dilation=d)(X)
    E = PWConv(num_filters, act=ReLU)(E)
    E_list.append(E)
E_stack = Stack(axis=1)(E_list)        # [B,N,H,W,C]

# 2) Router
logits = Dense(N)(GAP(X))              # [B,N]
if routing_mode == "topk":
    idx = TopKIndices(logits, top_k)   # [B,top_k]
    mask = OneHot(idx, N).sum(axis=1)  # [B,N]
    logits = logits - 1e9*(1-mask)
scores = Softmax(logits)               # [B,N]

# Weighted sum of experts
scores_b = Reshape([N,1,1,1])(scores)
F_local  = Sum(axis=1)(scores_b * E_stack)

# 3) Global branch
G = GAP2D(X)                            # [B,1,1,C]
G = PWConv(num_filters, act=ReLU)(G)
G = Tile(G, target=(H,W))               # [B,H,W,C]

# 4) Switch gating
gate = Sigmoid( BN( Conv3x3(num_filters)(F_local - G) ) )
Y = gate*F_local + (1-gate)*G

# 5) Refinement
Y = BN( DWConv3x3(act=GELU)(Y) )
if downsample:
    Y = Conv3x3(num_filters, stride=2, act=GELU)(Y)

return Y
```

Listing 1: Pseudo-code: Local–Global Mixture of Experts (LoGo-MoE)

```
# Inputs:
#   F: feature map [B,H,W,C]
# Outputs:
#   F_out: refined feature map [B,H,W,C]

# Channel gate (alpha) using pooled spatial and spectral descriptors
alpha = MLP( LN([
    GAP(F),                 # [B,C]
    GMP(F),                 # [B,C]
    GAP(DCT(F)),            # [B,C]
    GMP(DCT(F)),            # [B,C]
    GAP(AvgPool(AvgPool(F)))# [B,C]
]))                           # alpha in [0,1]^C after final sigmoid

# Spatial gate (M)
M = sigmoid( DWConv( gamma1 * AvgPool(F) + gamma2 * MaxPool(F) ) )
     # [B,H,W,1]

# Apply dual-axis gating
F_out = (alpha (broadcast to [B,H,W,C])) * (M (broadcast to [B,H,W,C])) * F
```

Listing 2: Pseudo-code: Dual-Attention Selective Aggregator Module (DASA)

```
# Input:
#   F: feature map [B,H,W,C]
# Output:
#   F_out: refined feature map [B,H,W,C]

# 1) Frequency paths via 2D-DCT
FDCT  = DCT2(F)
Flow  = PW( DW(FDCT) )
Fmean = mean_spatial(FDCT)                  # broadcast to [B,H,W,C]
Fhigh = PW( DW( abs(FDCT - Fmean) ) )
Ffreq = Flow + Fhigh

# 2) Multi-scale dilated context with learned weights
w = softmax( GAP(F) )                       # [B,3] for 3 dilations
d4  = DW_dilate(F, rate=4)
d8  = DW_dilate(F, rate=8)
d12 = DW_dilate(F, rate=12)
Fdil = w[:,0]*d4 + w[:,1]*d8 + w[:,2]*d12   # implicit broadcast over spatial dims

# 3) Channel calibration and spatial gate
Fse  = SE(Fdil)
Msp  = sigmoid( DW( GAP(Fse) + GMP(Fse) ) ) # [B,H,W,1]

# 4) Fuse frequency and spatially-calibrated context
F_out = PW( DW( Ffreq + Fse * Msp ) )
```

Listing 3: Pseudo-code: Frequency-Aware Multi-Scale Refinement Module (FAMSR)

```
# Inputs:
#   E: encoder features [B,He,We,C]
#   D: decoder features [B,Hd,Wd,C]
# Output:
#   D_ref: corrected decoder features [B,Hd,Wd,C]

# 1) Projections and alignment
E_proj = Conv1x1(E)                          # [B,He,We,C]
D_resz = resize_to(D, spatial_of=E_proj)     # [B,He,We,C]
D_proj = Conv1x1(D_resz)                     # [B,He,We,C]

# 2) Cross interaction and local fusion
gate_E = sigmoid(Conv1x1(E_proj))            # [B,He,We,C]
F      = ReLU(Conv3x3(E_proj + D_proj * gate_E))

# 3) Confidence/uncertainty proxies
S = Softmax(Conv1x1(DWConv3x3(F)))           # pseudo-logits->
    softmax across channels
H = -sum_over_channels( S * log(S + eps) )  # entropy, shape [B,He,
    We,1]
C = 1 - cosine( normalize(E_proj), normalize(D_proj) )   # [B,He,We
    ,1]

# 4) Uncertainty fusion and FP/FN mask
U     = sigmoid(Conv( concat(H, C) ))        # [B,He,We,1]
Mfpfn = sigmoid(Conv( concat(H, C) ))        # [B,He,We,1]

# 5) Residual correction path
R = ReLU(Conv3x3( F * Mfpfn ))               # [B,He,We,C]

# 6) Output (suppress uncertain, add correction, keep decoder bias)
D_ref = GELU( Conv1x1( (1 - U) * F + R + D_proj ) )
```

Listing 4: Pseudo-code: False Positive/Negative Corrective Attention Module (FPFN-CAM)