PROGRESSIVE TASK-CONDITIONED CORRECTION FOR SEMANTIC SEGMENTATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Hierarchical segmentation decoders repeatedly combine encoder and decoder representations, but their discrepancy provides candidate refinement content rather than a task update by itself. We formulate decoding as *progressive task-conditioned correction* and introduce DiCoSeg. In the encoder, Consistency-Guided Directional Aggregation (CGDA) adapts multi-directional state aggregation while retaining the pretrained equal-path computation as its initialization. In the decoder, each Corrective Fusion Block (CFB) estimates where a correction is supported and its signed task-space direction, transports supported corrections while preserving their support, and uses the resulting Corrective State to modulate discrepancy-derived features. A Persistent Task State carries this corrective context across decoder stages, turning hierarchical decoding into an evolving correction process rather than a sequence of isolated fusions. Across medical-image segmentation and natural-image scene parsing, DiCoSeg consistently improves segmentation quality, while controlled analyses support the intended separation between observable discrepancy and corrective action. DiCoSeg therefore separates two operations often coupled implicitly in hierarchical fusion: identifying representational discrepancy and deciding how that discrepancy should update the task representation.

1 INTRODUCTION

Hierarchical segmentation combines semantic abstraction with spatial detail through encoder–decoder hierarchies (Ronneberger et al., 2015). Recent decoders increasingly adapt multi-scale fusion and skip interaction (Chen et al., 2025; Hwang et al., 2025; Bhurtel & Rawat, 2026; Rahman et al., 2026; Tao et al., 2026), while state-space models provide an efficient basis for long-range dense prediction (Fu et al., 2025). This makes a more specific question central to hierarchical decoding: when encoder and decoder representations differ, how should that difference change the current task representation?

At decoder stage l, the encoder feature E_l and the current decoder feature D_l describe the same spatial content from different representational states. Their discrepancy can reveal content that the decoder has not yet incorporated, but the discrepancy itself does not specify how the segmentation representation should change. Directly turning this difference into an update therefore couples two distinct problems: identifying a representation difference and deciding how that difference should influence the task. Adaptive decoders already show the value of conditioning feature interaction on semantic context, scale, and skip relevance (Chen et al., 2025; Hwang et al., 2025; Bhurtel & Rawat, 2026; Rahman et al., 2026). We make the transition from discrepancy to update explicit.

As illustrated in Fig. 1, we formulate hierarchical decoding as *progressive task-conditioned correction*. Feature discrepancy identifies candidate refinement content; an evolving task state interprets that content in the context of the segmentation task; and the resulting corrective state controls how refinement is written back to the decoder. Because the task state persists across stages, each correction is formed in the context accumulated from coarser decoding stages.

We instantiate this formulation in **DiCoSeg**. The encoder builds on visual selective state-space modeling (Gu & Dao, 2024; Liu et al., 2024b) and introduces **Consistency-Guided Directional Aggregation (CGDA)**, which adapts the contribution of directional states according to their consistency with a directional consensus. Its signed consistency coefficient is zero-initialized so that the encoder

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

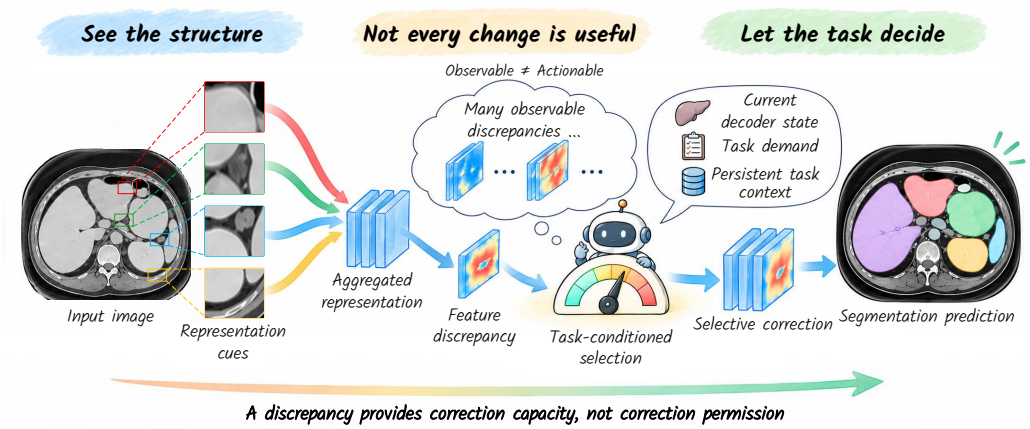


Figure 1: Motivation. Encoder–decoder discrepancy provides candidate refinement content, while the evolving task context determines how that content is converted into a selective correction.

begins from the pretrained equal-path aggregation. The decoder applies a **Corrective Fusion Block (CFB)** at each resolution. CFB estimates correction support and direction, transports supported corrections while preserving their support, maintains a persistent task state across stages, and uses the resulting corrective state to modulate discrepancy-derived features before residual writeback. Thus, CGDA structures directional representation formation, while CFB governs how hierarchical discrepancy becomes a task-conditioned representation update.

Our contributions are summarized as follows:

- We formulate hierarchical segmentation decoding as progressive task-conditioned correction, explicitly separating encoder–decoder discrepancy from the task-conditioned update it induces.
- We introduce Consistency-Guided Directional Aggregation (CGDA), which adapts multi-directional state aggregation through consensus-based consistency while preserving pretrained equal-path aggregation at initialization.
- We develop a Corrective Fusion Block (CFB) that maintains a persistent task state and converts representation discrepancy into selective feature correction through corrective-state estimation and support-preserving transport.

2 Related Work

Hierarchical decoding and adaptive feature fusion. U-shaped segmentation networks established the encoder–decoder hierarchy as a durable way to combine semantic abstraction with spatial detail (Ronneberger et al., 2015). Recent work increasingly treats skip pathways and decoder fusion as adaptive representation interactions. Decoder-centric designs revisit multi-scale restoration and fusion (Wazir & Kim, 2025; Bhurtel & Rawat, 2026); GLM-SFNet combines global and local semantic fusion, MARSeg uses adaptive spatial–channel fusion, and MambaLiteUNet and MHMamba introduce adaptive or gated skip interaction (Chen et al., 2025; Hwang et al., 2025; Rahman et al., 2026; Tao et al., 2026). These methods support the value of conditionally combining hierarchical features. DiCoSeg focuses on the next step in this interaction: the encoder–decoder difference is treated as candidate refinement content rather than as an update by itself.

Adaptive refinement and task-dependent processing. Segmentation refinement is also becoming increasingly dependent on the current prediction or task context. PromptMoE selects refinement experts from the image and coarse mask, SegMoTE performs modality- and task-adaptive expert selection, and IAD uses semantic self-optimization to refine subsequent segmentation behavior (Price et al., 2026; Lu et al., 2026; Hong et al., 2026). These approaches motivate task-dependent feature

transformation. DiCoSeg realizes this principle within hierarchical decoding by maintaining a task state across scales and using it to condition the corrective state that governs feature updates.

State-space models for dense prediction. Mamba introduced input-dependent selective state spaces, and VMamba adapted selective scanning to two-dimensional visual representations (Gu & Dao, 2024; Liu et al., 2024b). Subsequent segmentation models combine state-space representations with local attention, multi-scale decoding, semantic fusion, or complementary spatial–frequency modeling (Fu et al., 2025; Chen et al., 2025; Wu et al., 2025). Recent systems further explore multi-head state-space modeling, adaptive skip fusion, cross-gated interaction, and refinement of information lost during state-space compression (Tao et al., 2026; Rahman et al., 2026; Zhu et al., 2026). Within this family, CGDA targets the aggregation of directional states, while the DiCoSeg decoder addresses how the resulting hierarchical representations are progressively corrected for segmentation.

3 METHOD

3.1 DICOSEG AS PROGRESSIVE TASK-CONDITIONED CORRECTION

We formulate hierarchical decoding as a sequence of task-conditioned corrective updates. At decoder stage l, let D_l denote the current decoder feature after upsampling, E_l the encoder feature at the same resolution, and T_{l-1} the Persistent Task State carried from the preceding coarser stage. Their feature discrepancy provides a representation-level signal for refinement, while the evolving task state provides the context in which that signal is converted into an update. A stage therefore implements the mapping

$$(D_l, E_l, T_{l-1}) \longrightarrow (F_l', T_l), \tag{1}$$

where F_l' is the updated decoder feature and T_l carries the accumulated task context to the next stage.

Figure 2 summarizes DiCoSeg. The encoder uses **Consistency-Guided Directional Aggregation (CGDA)**, and the deepest encoder representation initializes T_0. Each decoder stage follows *upsampling → Corrective Fusion Block (CFB) → VSS refinement*. CFB comprises **Corrective-State Estimation (CSE)**, **Support-Preserving Correction Transport (SPCT)**, and **Selective Feature Correction (SFC)**.

3.2 CONSISTENCY-GUIDED DIRECTIONAL AGGREGATION

Visual state-space blocks aggregate states produced by multiple scan directions. CGDA asks whether the direction-specific deviation from the multi-directional consensus is consistent with the representation residual left by that consensus. Let O be the local observation before selective scanning and $\{Y_k\}_{k=1}^{N_{\text{dir}}}$ the aligned directional states, with $N_{\text{dir}} = 4$ in our implementation. Their directional consensus is

$$\bar{Y} = \frac{1}{N_{\text{dir}}} \sum_{k=1}^{N_{\text{dir}}} Y_k. \tag{2}$$

We compare the normalized residual between the local observation and the consensus with the deviation contributed by each direction. Writing

$$r = \text{LN}(O) - \text{LN}(\bar{Y}), \qquad \delta_k = \text{LN}(Y_k) - \text{LN}(\bar{Y}), \tag{3}$$

the directional consistency score is

$$c_k = \frac{2\langle r, \delta_k \rangle}{\|r\|_2^2 + \|\delta_k\|_2^2 + \epsilon}. \tag{4}$$

CGDA converts these scores into directional weights through a learnable signed consistency coefficient,

$$\alpha = \tanh(a), \qquad w_k = N_{\text{dir}}\ \text{softmax}_k(\alpha c_k), \qquad Y = \sum_{k=1}^{N_{\text{dir}}} w_k Y_k. \tag{5}$$

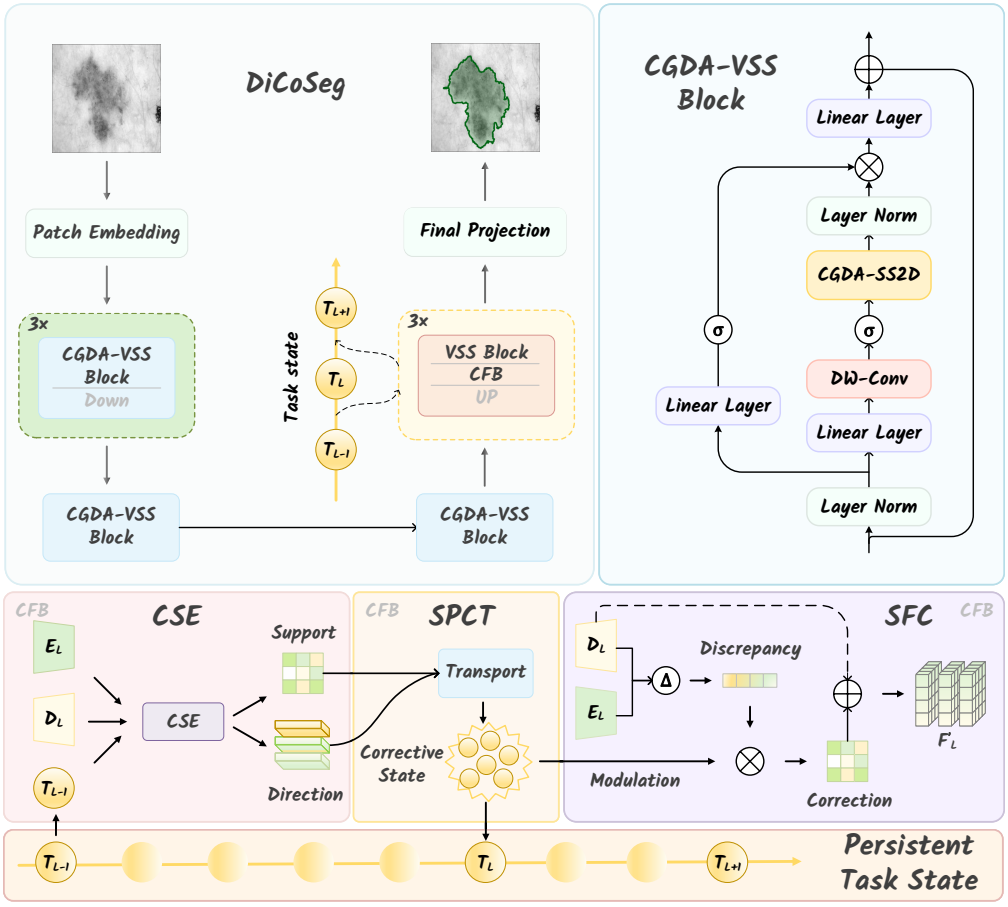


Figure 2: DiCoSeg architecture. CGDA acts in encoder VSS blocks; each decoder stage applies CFB (CSE → SPCT → SFC) before VSS refinement, while the task state is updated across stages.

The inner product and squared norms in Eq. equation 4 are evaluated over channels independently at each spatial location, yielding spatially varying c_k and w_k. The coefficient α is signed: positive values favor directions with larger consistency scores, whereas negative values reverse that ordering. We initialize $a = 0$, which gives $\alpha = 0$ and $w_k = 1$, so the block begins exactly from the pretrained equal-path aggregation. Training then learns both the magnitude and sign of each block's response to the consistency ordering. The selective-scan transition itself is unchanged; CGDA modifies only how the directional states are aggregated. Learned block-wise coefficients are reported in Appendix A.5.

3.3 Task-Conditioned Corrective Decoding

The decoder maintains a Persistent Task State alongside the feature hierarchy. Let X_{deep} be the deepest encoder representation. A task-space projection initializes the K-channel state logits

$$T_0 = \psi(X_{\text{deep}}). \tag{6}$$

Before stage l, the state carried from the coarser stage is bilinearly resized to the spatial resolution of D_l and E_l,

$$\widetilde{T}_{l-1} = \mathcal{R}_l(T_{l-1}), \tag{7}$$

where $\mathcal{R}_l$ is the identity when the resolutions already match. CSE and SPCT use probabilities obtained from $\widetilde{T}_{l-1}$ by sigmoid (binary) or softmax (multiclass), while the persistent state itself is updated in logit space.

Corrective-State Estimation. A corrective update has two complementary attributes: its spatial relevance and its direction in task space. CSE estimates them jointly from the current decoder feature, encoder feature, and task state,

$$S_l = \sigma\Big(h_s(D_l, E_l, \widetilde{T}_{l-1})\Big), \qquad Q_l = \tanh\Big(h_q(D_l, E_l, \widetilde{T}_{l-1})\Big), \tag{8}$$

where S_l is the *correction support* and Q_l is the *correction direction*. S_l is a class-agnostic spatial field indicating where a task-relevant correction is supported, whereas Q_l specifies the signed update in task space. Their product $S_l \odot Q_l$ is the local supported correction proposed at stage l.

Support-Preserving Correction Transport. A correction direction is meaningful under the support from which it was inferred. We therefore transport the support-weighted correction and the support itself through the same spatial propagation operator $\mathcal{P}_l$, conditioned on the current task and encoder context. The transported direction is recovered as

$$Q_l^T = \frac{\mathcal{P}_l(S_l \odot Q_l)}{\mathcal{P}_l(S_l) + \epsilon}. \tag{9}$$

The resulting *corrective state* combines local and transported directions while retaining the receiver-side support,

$$C_l = S_l \odot \left[(1 - \lambda_l)Q_l + \lambda_l Q_l^T\right], \tag{10}$$

where $\lambda_l \in (0, 1)$ is learned. In practice, $\mathcal{P}_l$ is a structure-aware normalized scan. Task probabilities and encoder geometry determine spatial conductance, while bidirectional horizontal and vertical recurrences propagate the joint field $[S_l \odot Q_l, S_l]$. A lightweight local geometry refinement further aligns propagation with task-probability structure. The operator acts once on the concatenated correction–support field, preserving the support normalization in Eq. equation 9; the full recurrence is given in Appendix A.5. This construction lets spatial transport extend an established correction without separating it from the support that gives the correction its task meaning.

The same corrective state updates the aligned Persistent Task State,

$$T_l = \widetilde{T}_{l-1} + \eta_l C_l, \tag{11}$$

with learned step size $\eta_l \in (0, 1)$. The resulting T_l is carried to the next decoder stage and resized there by equation 7. Thus each stage receives the corrective context accumulated at coarser resolutions.

Selective Feature Correction. The encoder–decoder discrepancy supplies the feature content available for refinement. We define

$$X_l = \mathrm{LN}(E_l) - \mathrm{LN}(D_l), \qquad R_l = \phi_l(X_l), \tag{12}$$

where R_l is the discrepancy-derived feature produced by a local transformation ϕ_l. The corrective state is projected into a feature-space task-conditioned modulation,

$$M_l = \tanh(W_l C_l). \tag{13}$$

Their interaction yields the realized correction

$$\Delta F_l = R_l \odot M_l. \tag{14}$$

Finally, the correction is written onto the ordinary encoder–decoder fusion,

$$F_l' = D_l + E_l + \gamma_l \Delta F_l, \tag{15}$$

where γ_l is a learned residual scale. Thus, feature discrepancy supplies refinement content, while the corrective state determines how that content is expressed as an update.

3.4 Learning the Corrective Process

The corrective supervision follows directly from the semantics of the decoder. At stage l, let Y_l be the task target represented at the stage resolution and let $P_l = \pi(\widetilde{T}_{l-1})$ be the task probability

induced by the spatially aligned incoming task state, where π is the task-appropriate probability mapping. The desired movement in task space is

$$\Delta_l^\star = Y_l - P_l. \tag{16}$$

Its magnitude determines where a correction is supported, while its signed values provide the target direction of that correction. The updated task state is simultaneously encouraged to remain aligned with the task target. We collect these stage-wise constraints into a single *corrective supervision* term,

$$\mathcal{L}_{\text{corr}} = \frac{1}{L}\sum_{l=1}^{L} \mathcal{D}_{\text{corr}}\big(S_l, Q_l, C_l, T_l; \Delta_l^\star, Y_l\big), \tag{17}$$

where $\mathcal{D}_{\text{corr}}$ supervises correction support, correction direction, neutral behavior in already-aligned regions, and task-state alignment. The corresponding target construction and component definitions are detailed in the appendix.

The core learning objective is therefore written compactly as

$$\mathcal{L}_{\text{core}} = \mathcal{L}_{\text{seg}} + \mathcal{L}_{\text{corr}}, \tag{18}$$

where the stage-wise weighting inside $\mathcal{L}_{\text{corr}}$ is absorbed into $\mathcal{D}_{\text{corr}}$. Exact target construction, component weights, and training-stabilization terms are given in the appendix.

4 Experiments

4.1 Experimental Setup

Datasets and metrics. We evaluate DiCoSeg on four medical image segmentation benchmarks and one natural-image scene parsing benchmark. **Synapse** uses abdominal CT volumes from the Multi-Atlas Labeling Beyond the Cranial Vault challenge (Landman et al., 2015); we report mean Dice similarity coefficient (DSC), 95th-percentile Hausdorff distance (HD95), and DSC for eight abdominal organs. **ACDC** contains cardiac MRI scans with annotations of the right ventricle, myocardium, and left ventricle (Bernard et al., 2018); we report mean DSC and HD95. **ISIC 2017** and **ISIC 2018** are dermoscopic lesion-segmentation benchmarks from the International Skin Imaging Collaboration (Codella et al., 2018; 2019); we report mIoU and DSC in the main text, with accuracy, specificity, and sensitivity provided in the appendix. **ADE20K** provides dense annotations over 150 object and stuff categories for scene parsing (Zhou et al., 2017); we report single-scale mIoU.

Evaluation protocol. The same DiCoSeg mechanism is used across the medical datasets. All medical results reported as mean $\pm$ standard deviation are averaged over three independent runs. Within each benchmark, reproduced methods use the same data split and preprocessing, while their published optimization recipe is followed when available. Synapse ablations use the same training and evaluation protocol across variants. Complete training and evaluation code is included with the supplementary material.

For ADE20K, we integrate the DiCoSeg mechanisms into VMamba-T under the same UPerNet segmentation setting and train at 512×512 resolution. FLOPs follow the VMamba reporting convention at 512×2048; the compact medical-model complexity analysis is reported at 224×224. Dataset-specific splits, optimization schedules, augmentation, corrective-target construction, and implementation details are provided in the appendix.

4.2 Main Results

We compare with established U-shaped and Transformer-based models (Ronneberger et al., 2015; Chen et al., 2024; Zhang et al., 2021) and recent state-space segmenters (Ruan et al., 2025; Liu et al., 2024a; Xu, 2024; Zhao et al., 2025; Zhu et al., 2025; Liang et al., 2026; Geng et al., 2026; Kang, 2026; Rahman et al., 2026).

Table 1: Synapse results. DSC is reported in percentage points; lower HD95 is better.

Method	DSC ↑	HD95 ↓	Aorta	Gall.	Kid.-L	Kid.-R	Liver	Pancr.	Spleen	Stom.
U-Net	77.04±0.17	35.55±1.92	84.98	64.17	80.54	76.39	92.81	55.15	88.03	74.24
TransUNet	77.68±0.26	29.76±1.96	85.15	65.47	79.87	77.15	93.59	56.22	89.02	74.98
VM-UNet	81.16±0.23	20.90±2.77	89.70	67.23	84.25	80.99	95.26	60.90	91.35	79.60
HC-Mamba	81.56±0.26	25.27±2.35	89.97	68.09	84.87	81.49	**97.30**	59.10	93.60	78.02
DMM-UNet	81.10±0.18	24.26±2.81	89.39	66.69	85.33	82.15	97.27	58.54	91.50	77.92
Swin-UMamba	81.62±0.19	20.54±1.21	90.34	69.05	85.44	80.50	96.73	59.33	92.43	79.14
CCViM	82.72±0.18	19.18±1.94	90.27	69.75	85.59	83.86	96.97	60.79	**94.12**	80.41
NexMamba	81.29±0.25	22.17±2.61	89.33	67.09	86.24	81.83	96.07	58.30	92.66	78.80
BDFNet	81.21±0.21	23.23±1.75	88.82	68.95	84.41	81.26	96.40	59.19	92.30	78.32
PMRNet	82.98±0.16	20.38±1.12	**90.90**	71.51	86.89	83.78	97.25	59.16	93.14	81.17
DiCoSeg	**83.77±0.18**	**14.73±1.54**	88.46	**71.77**	**88.46**	**85.06**	95.67	**64.91**	90.62	**85.16**

On Synapse, DiCoSeg improves both aggregate overlap and boundary quality, reaching 83.77 ± 0.18 DSC and 14.73 ± 1.54 HD95. The organ-wise results show that the gain is not uniform: DiCoSeg is strongest on the gallbladder, both kidneys, pancreas, and stomach, with the largest margins on the pancreas and stomach.

Table 2: ACDC and ISIC results. Values are percentages except HD95.

	ACDC		ISIC17		ISIC18	
Method	DSC ↑	HD95 ↓	mIoU ↑	DSC ↑	mIoU ↑	DSC ↑
U-Net	89.99±0.15	1.35±0.03	76.82±0.23	86.89±0.15	77.93±0.18	87.60±0.11
TransFuse	90.74±0.17	1.46±0.07	79.28±0.26	88.44±0.16	80.41±0.24	89.14±0.14
VM-UNet	90.74±0.17	1.24±0.08	80.05±0.22	88.92±0.14	81.11±0.21	89.56±0.13
HC-Mamba	90.37±0.22	1.31±0.09	79.18±0.23	88.38±0.15	80.66±0.22	89.29±0.14
DMM-UNet	90.30±0.12	1.44±0.15	80.32±0.19	89.09±0.12	80.55±0.28	89.22±0.17
Swin-UMamba	91.39±0.19	1.28±0.04	80.92±0.23	89.45±0.14	81.38±0.26	89.73±0.16
CCViM	91.19±0.20	1.35±0.05	81.14±0.24	89.59±0.15	81.70±0.22	89.92±0.13
NexMamba	90.46±0.14	1.40±0.07	79.84±0.24	88.79±0.15	80.77±0.26	89.36±0.16
BDFNet	90.32±0.14	1.41±0.08	80.29±0.22	89.07±0.13	80.96±0.16	89.48±0.10
MambaLiteUNet	–	–	81.02±0.18	89.51±0.11	81.54±0.19	89.83±0.12
PMRNet	91.52±0.17	1.31±0.06	81.27±0.24	89.67±0.14	81.57±0.14	89.85±0.09
DiCoSeg	**92.32±0.14**	**1.17±0.08**	**81.85±0.22**	**90.02±0.13**	**82.35±0.19**	**90.32±0.11**

The pattern persists across cardiac MRI and dermoscopic lesion segmentation. On ACDC, DiCoSeg reaches 90.87/90.78/95.33 DSC on the right ventricle, myocardium, and left ventricle, improving all three structures. On ISIC17/18, sensitivity reaches 88.83/90.45 while specificity remains 98.28/96.83, complementing the mIoU and DSC gains in Table 2.

Table 3: ADE20K with VMamba-T/UPerNet. FLOPs are measured at 512×2048.

Method	mIoU (SS) ↑	Params	FLOPs
VMamba-T (Liu et al., 2024b)	47.9	62M	949G
VMamba-T + DiCoSeg	**50.7**	63M	954G

Under the aligned VMamba-T/UPerNet setting, integrating DiCoSeg raises single-scale mIoU from 47.9 to 50.7, while parameters increase from 62M to 63M and FLOPs from 949G to 954G.

4.3 CONTROLLED ABLATIONS ON SYNAPSE

Table 4: Synapse ablations under the same protocol. DiCoSeg reference: 83.96 DSC / 15.45 HD95.

(a) Representation and corrective state

Variant	DSC ↑	HD95 ↓
VSS Baseline	82.03	21.56
CGDA Only	83.13	20.89
CFB Only	82.93	21.21
w/o Correction Support	83.45	17.58
w/o Persistent Task State	83.76	18.36
DiCoSeg	**83.96**	**15.45**

(b) Realization, transport, and learning

Variant	DSC ↑	HD95 ↓
Discrepancy-Only Correction	83.65	20.07
w/o Correction Transport	83.30	17.27
Naive Correction Transport	83.47	18.53
w/o Corrective Supervision	83.38	17.09
Matched Deep Supervision	82.68	19.63
DiCoSeg	**83.96**	**15.45**

CGDA Only reaches 83.13/20.89 DSC/HD95 and CFB Only reaches 82.93/21.21, compared with 82.03/21.56 for the VSS Baseline. With both components active, DiCoSeg reaches 83.96/15.45. The joint setting reduces HD95 by 6.11 relative to the baseline, compared with 0.67 for CGDA alone and 0.35 for CFB alone; the largest geometric improvement is obtained when both are enabled.

The decoder controls clarify how discrepancy becomes correction. Removing Correction Support or the Persistent Task State raises HD95 to 17.58 and 18.36. More strikingly, Discrepancy-Only Correction retains 83.65 DSC but increases HD95 to 20.07. Thus, discrepancy-derived content can preserve substantial regional overlap even when task-conditioned modulation is removed, while geometric quality degrades sharply. Together, these controls separate candidate refinement content from the task-conditioned corrective state that realizes it.

Transport and supervision provide two further controls on this process. Removing transport gives 83.30/17.27, whereas directly transporting the correction direction gives 83.47/18.53; SPCT reaches 83.96/15.45 when the supported correction and its support are transported together, distinguishing propagation from support-preserving transport. With Corrective Supervision, DiCoSeg reaches 83.96/15.45, compared with 82.68/19.63 under Matched Deep Supervision, showing an advantage over ordinary decoder-scale prediction supervision with the same total auxiliary weight. For CGDA, fixing $\alpha = 1$ throughout training gives 83.25/16.98, compared with 83.96/15.45 for the signed coefficient learned from the zero start. In the learned model, six of eight CGDA blocks have positive α and two have negative α, with mean $|\alpha| = 0.0282$. Over the Synapse test set, the realized weights stay close to equal-path aggregation, with mean $|w_k - 1| = 0.00862$ and mean path spread 0.02307. Thus, CGDA behaves as a small, block-specific adjustment around the pretrained directional prior rather than a hard routing mechanism.

4.4 CORRECTIVE BEHAVIOR

Figure 3 visualizes representative Synapse slices together with the intermediate quantities used by the corrective decoder. Feature Discrepancy is spatially widespread, while Correction Support and the resulting Corrective State are more localized; the Realized Correction changes again after task-conditioned modulation in SFC. This qualitative separation is consistent with the stage-wise statistics reported below.

The internal statistics show sparse Correction Support. Its mean stays near 0.05 and only about 3% of locations exceed 0.5, while Feature Discrepancy remains close to unit mean magnitude. The Corrective State stays near 4×10^{-3}, leaving a clear magnitude separation between observed discrepancy and the state carried forward. Across stages, the Discrepancy-Derived Feature decreases from 0.959 to 0.336, while Task-Conditioned Modulation is largest at the finest stage (0.0801). The trend suggests larger discrepancy-derived content at coarse scales and stronger modulation of a smaller signal at the finest scale.

At 224×224, DiCoSeg increases the VSS baseline from 18.516M/4.491G to 18.782M/4.611G parameters/FLOPs, corresponding to 1.44% more parameters and 2.67% more FLOPs. On a single NVIDIA A800 with batch size 1, mean inference latency is 37.78 ms/image for DiCoSeg versus 8.77 ms/image for the VSS baseline, while peak allocated memory is 210.70 versus 192.27 MiB. The full corrective decoder therefore incurs a larger wall-clock cost than the static FLOP increase suggests. Full efficiency statistics are reported in Appendix B.

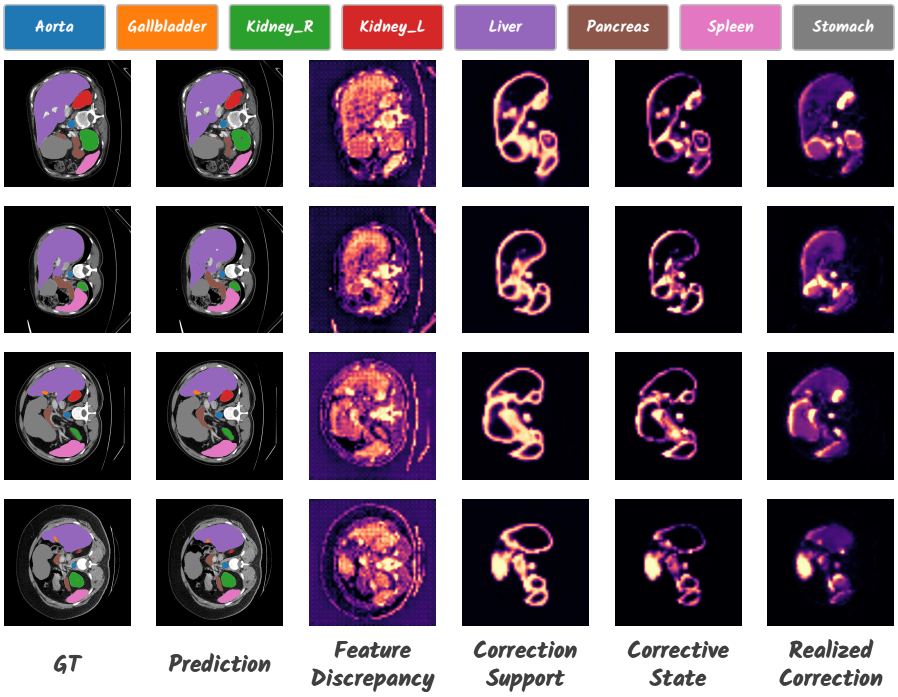


Figure 3: Qualitative corrective behavior on Synapse. Feature Discrepancy, Correction Support, Corrective State, and Realized Correction correspond to X_l, S_l, C_l, and ΔF_l, respectively.

Table 5: Stage-wise Synapse diagnostics. Entries are mean absolute magnitudes; Active denotes $S_l > 0.5$.

Stage	$\text{mean}(S_l)$	Active	$\lvert X_l \rvert$	$\lvert C_l \rvert$	$\lvert R_l \rvert$	$\lvert M_l \rvert$	$\lvert \Delta F_l \rvert$
1	0.0509	3.33%	1.028	0.00492	0.959	0.0469	0.0544
2	0.0471	2.92%	0.949	0.00389	0.522	0.0414	0.0258
3	0.0476	2.96%	0.936	0.00406	0.336	0.0801	0.0306

Case-level Synapse results provide a complementary view of this behavior. DiCoSeg improves DSC on 9 of 12 volumes and HD95 on 10 of 12, with both metrics improving on 8. A small number of mixed or degraded cases remain, including a volume with higher DSC but worse HD95; complete case-level results are reported in Appendix D.

5 Conclusion

DiCoSeg formulates hierarchical decoding as progressive task-conditioned correction, separating representation discrepancy from the state that governs decoder updates. CGDA provides pretrained-compatible directional encoding, while CFB combines support-preserving transport, persistent task-state updates, and selective feature correction. Across medical benchmarks and ADE20K, DiCoSeg improves segmentation quality, while the Synapse ablations and diagnostics support treating representational difference and task-conditioned update as distinct problems. More broadly, effective correction need not rely on increasingly elaborate signals: candidate refinement content and task-dependent realization should remain separate, favoring explicit task conditioning with a direct, pretrained-compatible correction path. Future work should study how to strengthen task selectivity without unnecessarily increasing optimization complexity or weakening transfer across domains.

Reproducibility Statement

The main paper specifies the model formulation and controlled evaluations, while the appendix provides implementation details, training protocols, loss construction, ablation definitions, and additional diagnostics. An anonymized implementation of DiCoSeg is submitted as supplementary code with the paper.

AI Assistance Statement

Generative AI tools were used to aid language polishing and literature retrieval and discovery. The authors reviewed and verified the resulting manuscript.

References

Olivier Bernard, Alain Lalande, Clement Zotti, Frederick Cervenansky, Xin Yang, Pheng-Ann Heng, Irem Cetin, Karim Lekadir, Oscar Camara, Miguel Angel Gonzalez Ballester, Gerard Sanroma, Sandy Napel, Steffen Petersen, Georgios Tziritas, Elias Grinias, Mahendra Khened, Varghese Alex Kollerathu, Ganapathy Krishnamurthi, Marc-Michel Rohe, Xavier Pennec, Maxime Sermesant, Fabian Isensee, Paul Jager, Klaus H. Maier-Hein, Peter M. Full, Ivo Wolf, Sandy Engelhardt, Christian F. Baumgartner, Lisa M. Koch, Jelmer M. Wolterink, Ivana Isgum, Yeonggul Jang, Yoonmi Hong, Jay Patravali, Shubham Jain, Olivier Humbert, and Pierre-Marc Jodoin. Deep learning techniques for automatic mri cardiac multi-structures segmentation and diagnosis: Is the problem solved? *IEEE Transactions on Medical Imaging*, 37(11):2514–2525, 2018. doi: 10.1109/TMI.2018.2837502.

Manish Bhurtel and Danda B. Rawat. Rmae-progress: Advancing semantic segmentation in unstructured environments. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20511–20520, 2026.

Jiahui Chen, Fei Qi, Chengyuan Chang, Qinjie Hu, Kaiwen Fu, Xiaotian Wang, and Kun Liu. Glmsfnet: Global-local vision-mamba with semantic fusion for medical image segmentation. In *Medical Image Computing and Computer Assisted Intervention – MICCAI 2025*, volume 15962, pp. 224–234. Springer Nature Switzerland, 2025. doi: 10.1007/978-3-032-04947-6_22.

Jieneng Chen, Jieru Mei, Xianhang Li, Yongyi Lu, Qihang Yu, Qingyue Wei, Xiangde Luo, Yutong Xie, Ehsan Adeli, Yan Wang, Matthew P. Lungren, Shaoting Zhang, Lei Xing, Le Lu, Alan L. Yuille, and Yuyin Zhou. Transunet: Rethinking the u-net architecture design for medical image segmentation through the lens of transformers. *Medical Image Analysis*, 97:103280, 2024. doi: 10.1016/j.media.2024.103280.

Noel Codella, Veronica Rotemberg, Philipp Tschandl, M. Emre Celebi, Stephen Dusza, David Gutman, Brian Helba, Aadi Kalloo, Konstantinos Liopyris, Michael Marchetti, Harald Kittler, and Allan Halpern. Skin lesion analysis toward melanoma detection 2018: A challenge hosted by the international skin imaging collaboration (isic). *arXiv preprint arXiv:1902.03368*, 2019.

Noel C. F. Codella, David Gutman, M. Emre Celebi, Brian Helba, Michael A. Marchetti, Stephen W. Dusza, Aadi Kalloo, Konstantinos Liopyris, Nabin Mishra, Harald Kittler, and Allan Halpern. Skin lesion analysis toward melanoma detection: A challenge at the 2017 international symposium on biomedical imaging (isbi), hosted by the international skin imaging collaboration (isic). In *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI)*, pp. 168–172, 2018. doi: 10.1109/ISBI.2018.8363547.

Yunxiang Fu, Meng Lou, and Yizhou Yu. Segman: Omni-scale context modeling with state space models and local attention for semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 19077–19087, 2025. doi: 10.1109/CVPR52734.2025.01777.

Mingjun Geng, Runmei Hu, Xiangpeng Bi, Zhiqiang Wei, Pin Sun, and Shugang Zhang. Boundary-aware attention with dual-stream frequency fusion for medical image segmentation. *IEEE Journal of Biomedical and Health Informatics*, 2026. doi: 10.1109/JBHI.2026.3687556.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. In *First Conference on Language Modeling*, 2024.

Jiankang Hong, Ye Luo, Yinan Liu, and Junsong Yuan. From infusion to assimilation distillation for medical image segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20985–20995, 2026.

Jeonghyun Hwang, Seungyeon Rhee, Minjeong Kim, Thanaporn Viriyasaranon, and Jang-Hwan Choi. Marseg: Enhancing medical image segmentation with mar and adaptive feature fusion. In *Medical Image Computing and Computer Assisted Intervention – MICCAI 2025*, volume 15963, pp. 296–305. Springer Nature Switzerland, 2025. doi: 10.1007/978-3-032-04965-0_28.

Boce Kang. Pmrnet: Physics-informed multi-scale refinement network for medical image segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15659–15668, 2026.

Bennett Landman, Zhoubing Xu, Juan Eugenio Iglesias, Martin Styner, Thomas Robin Langerak, and Arno Klein. 2015 miccai multi-atlas labeling beyond the cranial vault – workshop and challenge. Synapse, 2015.

Xingzhu Liang, Leilei Lu, Guoao Ge, Jinliang Zhang, Yu-e Lin, Yuhang Yao, and Nianhan Jie. Nexmamba: Nexus-driven multi-domain mamba network for medical image segmentation. *Biomedical Signal Processing and Control*, 122:110385, 2026. doi: 10.1016/j.bspc.2026.110385.

Jiarun Liu, Hao Yang, Hong-Yu Zhou, Yan Xi, Lequan Yu, Cheng Li, Yong Liang, Guangming Shi, Yizhou Yu, Shaoting Zhang, Hairong Zheng, and Shanshan Wang. Swin-umamba: Mamba-based unet with imagenet-based pretraining. In *Medical Image Computing and Computer Assisted Intervention – MICCAI 2024*, volume 15009, pp. 615–625. Springer Nature Switzerland, 2024a. doi: 10.1007/978-3-031-72114-4_59.

Yue Liu, Yunjie Tian, Yuzhong Zhao, Hongtian Yu, Lingxi Xie, Yaowei Wang, Qixiang Ye, Jianbin Jiao, and Yunfan Liu. Vmamba: Visual state space model. In *Advances in Neural Information Processing Systems*, volume 37, pp. 103031–103063, 2024b. doi: 10.52202/079017-3273.

Yujie Lu, Jingwen Li, Sibo Ju, Yanzhou Su, He Yao, Yisong Liu, Min Zhu, and Junlong Cheng. Segmote: Token-level mixture of experts for medical image segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 36332–36342, 2026.

Stephen Price, Danielle L. Cote, and Elke A. Rundensteiner. Promptmoe: A segmentation refinement framework leveraging mixture of experts for improved prompting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6325–6335, 2026.

Md Maklachur Rahman, Soon Ki Jung, and Tracy Hammond. Mambaliteunet: Cross-gated adaptive feature fusion for robust skin lesion segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8556–8565, 2026.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pp. 234–241. Springer, 2015. doi: 10.1007/978-3-319-24574-4_28.

Jiacheng Ruan, Jincheng Li, and Suncheng Xiang. Vm-unet: Vision mamba unet for medical image segmentation. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 2025. doi: 10.1145/3767748.

Hanjun Tao, Hua Wang, and Fan Zhang. Mhmamba: Multi-head mamba for 3d brain tumor segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Findings*, pp. 7388–7397, 2026.

Saad Wazir and Daeyoung Kim. Rethinking decoder design: Improving biomarker segmentation using depth-to-space restoration and residual linear attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 30861–30871, 2025. doi: 10.1109/CVPR52734.2025.02874.

Weitong Wu, Zhaohu Xing, Jing Gong, Qin Peng, and Lei Zhu. Hybridmamba: A dual-domain mamba for 3d medical image segmentation. In *Medical Image Computing and Computer Assisted Intervention – MICCAI 2025*, volume 15962, pp. 279–289. Springer Nature Switzerland, 2025. doi: 10.1007/978-3-032-04947-6_27.

Jiashu Xu. Hc-mamba: Vision mamba with hybrid convolutional techniques for medical image segmentation, 2024.

Yundong Zhang, Huiye Liu, and Qiang Hu. Transfuse: Fusing transformers and cnns for medical image segmentation. In *Medical Image Computing and Computer Assisted Intervention – MICCAI 2021*, pp. 14–24. Springer, 2021. doi: 10.1007/978-3-030-87193-2_2.

Liquan Zhao, Mingxia Cao, and Yanfei Jia. Dmm-unet: dual-path multi-scale mamba unet for medical image segmentation. *Journal of Medical Imaging*, 12(5):054003, 2025. doi: 10.1117/1.JMI.12.5.054003.

Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5122–5130, 2017. doi: 10.1109/CVPR.2017.544.

Kai Zhu, Zhenyu Cui, Zehua Zang, and Jiahuan Zhou. Rs-ssm: Refining forgotten specifics in state space model for video semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10741–10752, 2026.

Yun Zhu, Dong Zhang, Yi Lin, Yifei Feng, and Jinhui Tang. Merging context clustering with visual state space models for medical image segmentation. *IEEE Transactions on Medical Imaging*, 44 (5):2131–2142, 2025. doi: 10.1109/TMI.2025.3525673.

A Implementation and Experimental Details

A.1 Synapse Training Protocol

Synapse uses the standard project split and 224×224 training slices, a batch size of 32, and 300 epochs. Main benchmark statistics are averaged over three independent runs; the ablation variants use the same training and evaluation pipeline. The encoder is initialized from pretrained VMamba weights. We optimize with AdamW using an initial learning rate of 3×10^{-4}, weight decay 10^{-2}, and $(\beta_1, \beta_2) = (0.9, 0.999)$. A cosine schedule decreases the learning rate to 10^{-5} over 300 epochs. Mixed-precision training is used with gradient clipping at norm 1.0. All ablations in Table 4 use the same optimization and evaluation protocol. Synapse follows the reference project's standard split and volume-level checkpointing protocol. Volume evaluation is performed every 100 epochs on the project-provided `test_vol` volume split, and the checkpoint with the highest volume DSC is retained for reporting.

Synapse slices are resized to 224×224. Training uses stochastic joint image–label geometric augmentation with a random right-angle rotation and flip, or a random rotation within approximately $[-20°, 20°]$. The grayscale input is repeated to three channels inside the network to match the pretrained patch embedding. The architecture uses encoder depths $[2, 2, 2, 2]$, decoder depths $[2, 2, 2, 1]$, state dimension 16, and drop-path rate 0.2.

A.2 ACDC Training Protocol

We use a 2-D slice-wise protocol with patient-level separation. The 100 labeled subjects are split 7:1:2 into training, validation, and test sets; subjects do not overlap across subsets. ED/ES NIfTI volumes are unfolded into axial slices for training, while validation and test predictions are reassembled into volumes before metric computation. Each slice is resized to 224×224 and min–max normalized independently. The single MRI channel is repeated internally to three channels to match the ImageNet-pretrained patch embedding.

Training augmentation follows the project ACDC pipeline. With probability 0.5, a slice undergoes a random multiple-of-$90°$ rotation followed by a random horizontal or vertical flip. Otherwise,

with conditional probability 0.5, it receives a random rotation in approximately $[-20^\circ, 20^\circ]$; the remaining samples receive no geometric augmentation. Images use cubic interpolation and masks use nearest-neighbor interpolation. Validation and test use only resize and min–max normalization.

Models are trained for 300 epochs with batch size 32 using AdamW, learning rate 3×10^{-4}, $(\beta_1, \beta_2) = (0.9, 0.999)$, weight decay 10^{-2}, and $\epsilon = 10^{-8}$. Cosine annealing is applied over 300 epochs with minimum learning rate 10^{-5}. Mixed precision and gradient-norm clipping at 1.0 are used. The segmentation loss is multiclass cross-entropy plus soft Dice. Corrective-supervision weights are 0.10/0.15/0.03/0.05 for support/direction/quiet-state/task-state terms, and PMS is weighted by 0.05. Validation is performed every 100 epochs; the checkpoint with the highest validation DSC is retained. The held-out test set is not used for model selection and is evaluated once after training with that checkpoint. HD95 follows the project protocol with unit voxel spacing $(1, 1, 1)$.

A.3 ISIC 2017 and ISIC 2018 Training Protocol

We follow the VM-UNet data organization and augmentation protocol for both ISIC benchmarks. Each dataset uses the project-provided 70%/30% train/held-out split, stored as separate training and validation directories and kept fixed for all reproduced methods. RGB images and binary masks are resized to 224×224. Training augmentation independently applies horizontal flip, vertical flip, and random rotation with probability 0.5; the rotation angle is sampled uniformly from $[0^\circ, 360^\circ]$. Evaluation uses only deterministic preprocessing and resize.

The preprocessing follows the original project pipeline. For ISIC18, the stored training/validation mean–standard-deviation pairs are 157.561/26.706 and 149.034/32.022; for ISIC17 they are 159.922/28.871 and 148.429/25.748. After this affine normalization, each image is min–max normalized and scaled to $[0, 255]$ as in the data loader used for the reported experiments.

Both datasets are trained for 300 epochs with batch size 32, AdamW at learning rate 3×10^{-4}, $(\beta_1, \beta_2) = (0.9, 0.999)$, weight decay 10^{-2}, and $\epsilon = 10^{-8}$. Cosine annealing uses $T_{\max} = 300$ and $\eta_{\min} = 10^{-5}$. Mixed precision and gradient clipping at maximum L_2 norm 1.0 are enabled. Validation is run every 30 epochs and the checkpoint with the highest validation mIoU is used for the reported metrics. The primary binary segmentation objective is BCE plus soft Dice. Corrective-supervision weights are 0.10/0.15/0.03/0.05 for support/direction/quiet-state/task-state terms, and PMS is weighted by 0.05. Binary predictions use a threshold of 0.5. We report DSC/F1, IoU, sensitivity, specificity, and accuracy.

A.4 ADE20K Training Protocol

ADE20K uses the official ADEChallengeData2016 semantic-segmentation split: 20,210 training images and 2,000 validation images. The held-out test annotations are not used. We integrate DiCoSeg into the standard VMamba-T/UPerNet pipeline. Training follows the standard MMSegmentation ADE20K recipe: image loading, annotation loading with `reduce_zero_label=True`, random resize, 512×512 random crop, random horizontal flip, photometric distortion, normalization, and packing. Validation is performed on the official validation set, and single-scale mIoU is reported there.

Optimization runs for 160k iterations with effective batch size 16. AdamW uses learning rate 6×10^{-5}, $(\beta_1, \beta_2) = (0.9, 0.999)$, and weight decay 0.01; normalization parameters are excluded from weight decay. The first 1,500 iterations use linear warmup from a factor of 10^{-6} to the base learning rate. From iteration 1,500 to 160k, the learning rate follows polynomial decay with power 1.0 and minimum value 0. Validation is run every 16k iterations. Apart from the DiCoSeg mechanism and its associated supervision, the VMamba-T/UPerNet training recipe is unchanged. The Persistent Task State uses $K = 150$ channels. Corrective targets use the same multiclass construction as Synapse: Y_l is the one-hot segmentation target resized to the current decoder scale, $P_l = \mathrm{softmax}(\widetilde{T}_{l-1})$, $\Delta_l^\star = Y_l - P_l$, $m_l = \frac{1}{2}\sum_c |\Delta_{l,c}^\star|$, $(\tau_s, t_s, \tau_q) = (0.20, 0.05, 0.10)$, and support/direction/quiet-state/task-state weights 0.10/0.15/0.03/0.05.

A.5 CGDA AND CFB IMPLEMENTATION DETAILS

CGDA. Each CGDA-VSS block contains one learnable scalar a and uses $\alpha = \tanh(a)$ with $\epsilon = 10^{-6}$ in the consistency score. The score c_k is evaluated in FP32 and detached from the gradient graph; gradients flow through the learned coefficient and the weighted directional states. The coefficient is signed rather than restricted to be nonnegative: $\alpha > 0$ follows the measured consistency ordering, $\alpha < 0$ reverses it, and $\alpha = 0$ recovers equal-path aggregation exactly.

Learned behavior of CGDA. We evaluate the learned signed coefficients and the realized directional redistribution of the full Synapse model over all 12 test volumes (1,568 slices). The coefficients are block dependent, including two negative values in deeper stages, while the realized path weights remain close to the pretrained equal-path reference and depart from it by different amounts across blocks. Table 6 reports both the learned coefficient and the resulting test-time redistribution.

Table 6: Learned CGDA coefficients and realized directional redistribution on Synapse.

Block	α	mean$\lvert w-1\rvert$	mean$(w_{\max}-w_{\min})$
E1-B1	+0.04324	0.01732	0.04359
E1-B2	+0.01934	0.00739	0.02044
E2-B1	+0.02765	0.01167	0.03115
E2-B2	+0.00943	0.00347	0.00961
E3-B1	+0.00572	0.00207	0.00589
E3-B2	-0.02769	0.01137	0.03014
E4-B1	+0.04498	0.00590	0.01720
E4-B2	-0.04763	0.00974	0.02651
Macro avg.	–	0.00862	0.02307

The block-wise pattern is not monotonic with depth: the largest realized spread occurs in E1-B1, whereas E3-B1 remains closest to equal-path aggregation. This is consistent with CGDA acting as a pretrained-compatible, data-dependent adjustment rather than a separate routing system. Because the realized weights depend on αc_k, coefficient magnitude alone does not determine the amount of redistribution.

The Fixed-Strength CGDA control sets $\alpha = 1$ throughout optimization, contrasting learned signed weighting from a pretrained-compatible zero start with unit-strength consistency weighting throughout training.

CSE. For a stage with feature width C and K task channels, decoder and encoder features are LayerNorm-normalized and projected to $H = \max(C/4, 24)$ channels. The aligned state $\widetilde{T}_{l-1}$ is converted to probabilities and projected by a 1×1 convolution to the same width. The three tensors are concatenated and fused by a 1×1 convolution, GroupNorm, SiLU, a depthwise 3×3 convolution, GroupNorm, and SiLU. Separate 1×1 heads produce one-channel Correction Support and K-channel Correction Direction. The direction head is zero-initialized; the support head is initialized to a probability of 0.10.

SPCT propagation operator. We denote the spatial propagation operator by $\mathcal{P}_l$ to distinguish it from the Persistent Task State T_l. Let $p = \pi(\widetilde{T}_{l-1})$ be the task probabilities and e a low-dimensional, ℓ_2-normalized edge descriptor obtained by projecting the LayerNorm-normalized encoder feature. For neighboring positions i, j, define

$$d^p_{ij} = \mathrm{mean}_c\, |p_{i,c} - p_{j,c}|, \qquad d^e_{ij} = \mathrm{mean}_r\, |e_{i,r} - e_{j,r}|, \tag{19}$$

$$\rho(d;\kappa) = \frac{d}{d+\kappa+\epsilon}, \qquad g_{ij} = \exp\left[-\gamma_p \rho(d^p_{ij};\kappa_p) - \gamma_e \rho(d^e_{ij};\kappa_e)\right], \tag{20}$$

where γ_p, γ_e are bounded learned scales and $\kappa_p, \kappa_e > 0$ are learned softness parameters. For a one-dimensional scan of a field V_i, the forward normalized recurrence is

$$N^{\rightarrow}_i = V_i + g_{i-1,i} N^{\rightarrow}_{i-1}, \qquad M^{\rightarrow}_i = 1 + g_{i-1,i} M^{\rightarrow}_{i-1}, \tag{21}$$

with $N_1^{\rightarrow} = V_1$ and $M_1^{\rightarrow} = 1$; the backward recurrence is symmetric. The bidirectional result is

$$\bar{V}_i = \frac{N_i^{\rightarrow} + N_i^{\leftarrow} - V_i}{M_i^{\rightarrow} + M_i^{\leftarrow} - 1 + \epsilon}. \tag{22}$$

We apply equation 22 horizontally and vertically. Let $m_i^h = (M_i^h - 1)_+$ and $m_i^v = (M_i^v - 1)_+$ denote the corresponding nonlocal masses. The region-level propagated field is

$$V_i^{\mathrm{reg}} = \frac{m_i^h \bar{V}_i^h + m_i^v \bar{V}_i^v}{m_i^h + m_i^v + \epsilon}, \tag{23}$$

with $V_i^{\mathrm{reg}} = V_i$ when both masses vanish.

A local geometry refinement then uses the task-probability structure tensor. Central differences give gradients (G_x, G_y); after 3×3 averaging,

$$J_{xx} = \langle G_x^2 \rangle_c, \quad J_{yy} = \langle G_y^2 \rangle_c, \quad J_{xy} = \langle G_x G_y \rangle_c, \tag{24}$$

$$\theta = \tfrac{1}{2} \operatorname{atan2}(2J_{xy}, J_{xx} - J_{yy} + \epsilon), \quad n = (\cos\theta, \sin\theta), \quad t = (-\sin\theta, \cos\theta). \tag{25}$$

The boundary strength is $b = s/(s+\kappa_b)$ with $s = \sqrt{J_{xx} + J_{yy} + \epsilon}$. For the eight radius-1 neighbors with unit direction u_j, tangent and normal alignments are $(t^\top u_j)^2$ and $(n^\top u_j)^2$. Define $a_{ij}^t = (t_i^\top u_j)^2$ and $a_{ij}^n = (n_i^\top u_j)^2$. For the eight radius-1 neighbors $\mathcal{N}(i)$, the tangent estimate and normal barrier are

$$V_i^{\tan} = \frac{V_i + \sum_{j \in \mathcal{N}(i)} g_{ij} a_{ij}^t V_j}{1 + \sum_{j \in \mathcal{N}(i)} g_{ij} a_{ij}^t}, \qquad r_i = \frac{\sum_{j \in \mathcal{N}(i)} a_{ij}^n (1 - g_{ij})}{\sum_{j \in \mathcal{N}(i)} a_{ij}^n + 10^{-4}}. \tag{26}$$

Their bounded conjunction

$$\omega_i^{\mathrm{geo}} = \frac{2 b_i r_i}{b_i + r_i + 10^{-4}} \tag{27}$$

controls a residual correction from the region scan toward the tangent estimate,

$$\mathcal{P}_l(V)_i = V_i^{\mathrm{reg}} + \sigma(\zeta_l)\, \omega_i^{\mathrm{geo}} \left(V_i^{\tan} - V_i^{\mathrm{reg}} \right), \tag{28}$$

where $\sigma(\zeta_l)$ is the learned geometry scale, initialized to 0.5. Replicate padding is used at image boundaries.

SPCT applies this same $\mathcal{P}_l$ once to the concatenated field $[S_l \odot Q_l,\, S_l]$. Splitting the result into numerator N_l^T and propagated support S_l^T yields

$$Q_l^T = \begin{cases} N_l^T / S_l^T, & S_l^T > 0.05, \\ Q_l, & \text{otherwise}, \end{cases} \tag{29}$$

followed by equation 10. The transport mixing coefficient is sigmoid-parameterized and initialized to 0.25; the Persistent Task State step is similarly parameterized and initialized to 0.50. All conductance and geometry computations are performed in FP32. The stage-level implementation is therefore: (1) bilinearly align T_{l-1}; (2) estimate (S_l, Q_l); (3) form the joint field $[S_l \odot Q_l, S_l]$; (4) apply the shared operator $\mathcal{P}_l$ using equations 20–28; (5) recover Q_l^T by support normalization; and (6) form C_l, update T_l, and realize ΔF_l. The supplementary code implements the same sequence.

SFC and task-state propagation. The deepest encoder feature is LayerNorm-normalized and linearly projected to the K task channels to initialize T_0. At every finer decoder stage, the carried state is resized by bilinear interpolation with `align_corners=False` before CSE/SPCT; after the update in equation 11, the new logit-space state is passed forward without an additional projection. For SFC, $X_l = \mathrm{LN}(E_l) - \mathrm{LN}(D_l)$ uses parameter-free LayerNorm. The local transform ϕ_l is a depthwise 3×3 convolution at width C, followed by GroupNorm/SiLU, a 1×1 reduction to $H = \max(C/4, 24)$, GroupNorm/SiLU, and a final 1×1 projection back to C. A 1×1 task-to-channel projection followed by $\tanh$ produces M_l. The realized correction is added to $D_l + E_l$ with a learned scalar residual scale initialized to 0.05.

A.6 Corrective Supervision

For stage l, let $\widetilde{T}_{l-1}$ denote the previous Persistent Task State after spatial alignment to the current decoder resolution, and let Y_l be the one-hot segmentation target resized to the same resolution. For the multiclass datasets (Synapse, ACDC, and ADE20K), the prior task probability and target correction are

$$P_l = \mathrm{softmax}(\widetilde{T}_{l-1}), \qquad \Delta_l^\star = Y_l - P_l. \tag{30}$$

We measure the magnitude of a task-space correction with the class-count-invariant variation

$$m_l = \frac{1}{2} \sum_c \left| \Delta_{l,c}^\star \right|. \tag{31}$$

The support target is a soft threshold of this magnitude,

$$\hat{S}_l = \sigma\left(\frac{m_l - \tau_s}{t_s} \right), \qquad \tau_s = 0.20, \quad t_s = 0.05. \tag{32}$$

Support supervision applies binary cross-entropy to the CSE support logits. Direction supervision uses Smooth-L_1 distance between Q_l and $\Delta_l^\star$, weighted by $\hat{S}_l$. Quiet-state regularization is applied where $m_l \le \tau_q$ with $\tau_q = 0.10$, penalizing the magnitude of the Corrective State C_l. Task-state supervision applies cross-entropy plus soft Dice to the updated task state T_l.

The corrective objective used across datasets is

$$\mathcal{L}_{\mathrm{corr}} = 0.10\mathcal{L}_{\mathrm{sup}} + 0.15\mathcal{L}_{\mathrm{dir}} + 0.03\mathcal{L}_{\mathrm{quiet}} + 0.05\mathcal{L}_{\mathrm{state}}. \tag{33}$$

For these multiclass datasets, the final segmentation objective is cross-entropy plus soft Dice. For binary ISIC segmentation, Y_l and P_l reduce to foreground probabilities, so $m_l = |\Delta_l^\star|$ and the same support and quiet-state thresholds are used.

A.7 Prediction-Mass Stability

Prediction-Mass Stability (PMS) is used only as a training stabilizer. For each foreground class c present in a sample, let $M_{p,c}$ and $M_{y,c}$ denote the mean predicted and target probability mass. We define

$$b_c = \frac{M_{p,c} - M_{y,c}}{M_{p,c} + M_{y,c} + \epsilon}, \tag{34}$$

and penalize only meaningful under-coverage,

$$\mathcal{L}_{\mathrm{PMS}} = \frac{1}{|\mathcal{C}_+|} \sum_{c \in \mathcal{C}_+} \mathrm{ReLU}(-b_c - \delta), \qquad \delta = 0.03, \tag{35}$$

where $\mathcal{C}_+$ contains foreground classes whose target mass exceeds 10^{-4}. Its training weight is 0.05. The corresponding ablation is reported in Appendix E.

A.8 Matched Deep Supervision Control

The Matched Deep Supervision control keeps the DiCoSeg architecture unchanged but removes Corrective Supervision. Three ordinary segmentation heads are attached at the decoder correction scales. Each head is trained with the same cross-entropy plus soft-Dice objective as the final prediction; the three losses are averaged and weighted by 0.33, matching the sum of the corrective auxiliary coefficients used in the implementation. PMS remains enabled in this control.

A.9 Ablation Definitions

Table 7 specifies the intervention behind each Synapse control. The architecture controls isolate CGDA and CFB; the state controls alter support or task-state persistence; the transport controls distinguish local, naive, and support-preserving propagation; and the supervision controls compare corrective targets with ordinary decoder-scale prediction heads.

Table 7: Synapse ablation definitions.

Variant	Modification relative to DiCoSeg
VSS Baseline	Disable CGDA and bypass CFB correction, retaining ordinary encoder–decoder fusion.
CGDA Only	Enable CGDA and bypass CFB correction.
CFB Only	Use equal-path VSS aggregation and retain the full CFB decoder.
w/o Correction Support	Set Correction Support to one and disable support supervision.
w/o Persistent Task State	Reuse the projected deepest task state at each decoder stage instead of accumulating stage-wise updates.
Naive Correction Transport	Transport the correction direction directly rather than using support-preserving normalization.
w/o Correction Transport	Use only the local correction direction in the Corrective State.
Discrepancy-Only Correction	Apply the discrepancy-derived feature without task-conditioned modulation.
w/o Corrective Supervision	Keep the architecture unchanged and set all corrective-supervision weights to zero.
Matched Deep Supervision	Replace Corrective Supervision with three ordinary decoder-scale segmentation heads with the same total auxiliary weight.
Fixed-Strength CGDA	Fix the CGDA consistency coefficient to $\alpha = 1$ from initialization instead of learning it from the zero start.
w/o Prediction-Mass Stability	Disable the PMS training stabilizer while keeping the architecture and Corrective Supervision unchanged.

Discrepancy-Only Correction retains the discrepancy-derived feature while removing task-conditioned modulation. The two transport controls compare local-only correction and direct direction transport with SPCT, while Matched Deep Supervision replaces corrective targets with ordinary prediction targets at the same decoder scales. Together with the architecture controls, these variants expose the roles of representation formation, corrective state, transport, and learning within the same decoder pipeline.

B Efficiency Analysis

We measure inference efficiency on a single NVIDIA A800 using 224×224 inputs and batch size 1. Runtime statistics use 30 warm-up iterations followed by 100 timed repetitions. Table 8 reports wall-clock latency, throughput, and peak allocated GPU memory together with the parameter and FLOP counts used in the main text.

Table 8: Inference efficiency at 224×224 on a single NVIDIA A800. Runtime uses batch size 1, 30 warm-up iterations, and 100 timed repetitions. Params are in millions, FLOPs in G, latency in ms/image, and peak allocated memory in MiB.

Method	Params	FLOPs	Mean ↓	Median ↓	P90 ↓	FPS ↑	Memory ↓
VSS Baseline	18.516	4.491	8.77	8.76	8.84	114.08	192.27
DiCoSeg	18.782	4.611	37.78	36.18	40.91	26.47	210.70

The parameter count increases by 1.44% and peak allocated memory by 9.59%, while mean latency increases by $4.31\times$. The runtime difference is substantially larger than the 2.67% FLOP increase, showing that static FLOPs alone do not capture the wall-clock cost of the corrective decoder.

C Additional Medical-Segmentation Results

C.1 ACDC Class-Wise Results

The aggregate ACDC improvement in Table 2 is shared across all three annotated cardiac structures rather than being driven by a single class. Table 9 reports the corresponding class-wise DSC values.

Table 9: Class-wise ACDC DSC.

Method	RV	Myocardium	LV
U-Net	88.06	88.18	93.73
TransFuse	88.97	89.14	94.10
VM-UNet	88.86	89.35	94.02
HC-Mamba	88.79	88.62	93.69
DMM-UNet	88.47	88.66	93.76
Swin-UMamba	89.56	89.84	94.78
CCViM	89.52	89.63	94.42
NexMamba	88.75	88.61	94.03
BDFNet	88.49	88.69	93.79
PMRNet	90.08	89.70	94.79
DiCoSeg	**90.87**	**90.78**	**95.33**

DiCoSeg gives the highest DSC for the right ventricle, myocardium, and left ventricle. Relative to the strongest competing class-wise values, the gains are 0.79, 0.94, and 0.54 points, respectively. The improvement is therefore distributed across both ventricular cavities and the thinner myocardial structure.

C.2 ISIC Accuracy, Specificity, and Sensitivity

The main text reports mIoU and DSC. Table 10 complements those overlap measures with accuracy, specificity, and sensitivity to show how the lesion predictions trade foreground recovery against background rejection.

Table 10: Additional ISIC17/18 metrics.

	ISIC17			ISIC18		
Method	Acc.	Spe.	Sen.	Acc.	Spe.	Sen.
U-Net	94.99	97.09	86.03	93.97	96.11	87.33
TransFuse	96.17	97.91	87.49	94.72	96.62	88.84
VM-UNet	96.36	98.23	87.07	94.93	96.72	89.35
HC-Mamba	96.18	98.12	86.58	94.82	96.81	88.64
DMM-UNet	96.37	97.98	88.40	94.78	96.71	88.79
Swin-UMamba	96.56	**98.49**	86.98	95.03	**96.91**	89.19
CCViM	96.54	98.10	88.78	95.10	96.84	89.70
NexMamba	96.28	97.99	87.80	94.80	96.44	89.70
BDFNet	96.41	98.24	87.31	94.85	96.45	89.87
MambaLiteUNet	95.98	97.69	88.72	95.01	96.50	90.38
PMRNet	96.60	98.35	87.93	95.05	96.72	89.87
DiCoSeg	**96.70**	98.28	**88.83**	**95.28**	96.83	**90.45**

DiCoSeg attains the highest accuracy and sensitivity on both ISIC17 and ISIC18. Specificity remains close to the strongest value in each benchmark (0.21 points below the best on ISIC17 and 0.08 below on ISIC18), so the higher lesion recall is not accompanied by a large loss in background specificity.

D Case-Level Synapse Analysis

We compare DiCoSeg with the VSS Baseline on each Synapse test volume. DiCoSeg improves mean DSC on 9 of 12 cases and reduces mean HD95 on 10 of 12 cases; both metrics improve simultaneously on 8 cases. Averaged over the 12 cases, the DSC change is +1.93 percentage points and the HD95 reduction is 6.12, closely matching the aggregate comparison in Table 4.

Table 11: Case-level Synapse comparison. Positive ΔDSC and ΔHD95 favor DiCoSeg.

Case	Base DSC	DiCoSeg DSC	ΔDSC	Base HD95	DiCoSeg HD95	ΔHD95
0001	74.58	78.74	+4.16	46.11	28.54	+17.57
0002	85.96	88.62	+2.66	17.19	5.77	+11.42
0003	73.12	77.21	+4.08	60.47	80.45	-19.98
0004	79.81	83.21	+3.39	27.39	6.00	+21.39
0008	71.68	68.36	-3.32	10.22	11.90	-1.68
0022	90.53	91.56	+1.03	9.62	2.26	+7.36
0025	79.08	86.37	+7.29	29.47	20.40	+9.08
0029	82.21	84.27	+2.06	4.94	3.74	+1.20
0032	88.94	88.87	-0.07	15.65	13.60	+2.05
0035	88.62	88.82	+0.21	3.37	3.23	+0.15
0036	85.94	85.91	-0.03	9.55	6.21	+3.34
0038	83.92	85.60	+1.68	24.82	3.26	+21.56

The largest DSC gain occurs on case 0025 (+7.29 points), while the largest HD95 reduction occurs on case 0038 (21.56), showing both overlap-dominant and boundary-dominant improvements. Case 0008 decreases on both metrics, while case 0003 improves DSC but has a higher HD95, illustrating that regional overlap and worst-case boundary distance can change differently at the case level.

E Additional Initialization and Stabilization Controls

Two secondary controls examine optimization choices surrounding the core architecture. Fixed-Strength CGDA fixes the signed consistency coefficient to $\alpha = 1$ from the first optimization step, while DiCoSeg learns it from the pretrained-compatible zero start. The PMS control leaves the architecture unchanged and removes only the prediction-mass stabilizer described in Appendix A.7.

Table 12: CGDA initialization and PMS controls on Synapse.

Variant	DSC ↑	HD95 ↓
Fixed-Strength CGDA	83.25	16.98
w/o Prediction-Mass Stability	83.79	16.71
DiCoSeg	**83.96**	**15.45**

Applying CGDA at fixed unit strength lowers DSC by 0.71 points and increases HD95 by 1.53 relative to DiCoSeg. Together with the learned behavior in Table 6, this control shows that forcing $\alpha = 1$ throughout optimization is inferior to learning a small signed coefficient from the pretrained-compatible zero initialization. Removing PMS produces a smaller change (-0.17 DSC and +1.26 HD95), consistent with its role as a training stabilizer rather than a component of the corrective decoder.