BackSplit: The Importance of Sub-dividing the Background in Biomedical Lesion Segmentation

Supplementary Material

Contents

A. Full Proofs and Theoretical Insights

Our goal is to prove that using a full K-class training regime we yield a greater Fisher information than training on a coarsened/collapsed binary label. This in turn leads to more statistically efficient predictions for the target class.

A.1. Setup and Notation

The notations used to prove the following sections are used from Sec. 3.1.

Assumption 1 (Regularity and Optimization / Compute)**.** *We work with a parametric family $\{p_\theta(Y \mid X) : \theta \in \Theta\}$, where $\Theta \subset \mathbb{R}^p$ is an open set containing the true parameter $\theta^\star$. We assume:*

1. ***Model regularity.*** *For each fixed input x the conditional density $p_\theta(Y \mid X = x)$ is correctly specified at $\theta^\star$ and is twice continuously differentiable in θ in a neighbourhood of $\theta^\star$. Differentiation and summation over Y can be interchanged, and the Fisher information matrices*

$$I_Y(\theta) = \mathbb{E}_\theta\big[s_Y(\theta)s_Y(\theta)^\top\big], \qquad I_Z(\theta) = \mathbb{E}_\theta\big[s_Z(\theta)s_Z(\theta)^\top\big]$$

   exist and are finite. At $\theta^\star$, $I_Y(\theta^\star)$ and $I_Z(\theta^\star)$ are non–singular.
2. ***Common architecture.*** *The multiclass and collapsed–binary models share the same network architecture and parameterization up to the final classification layer, so that both likelihoods are defined on the same parameter space Θ.*
3. ***Optimization / compute parity.*** *In practice, both models are trained with the same optimization algorithm and comparable compute budgets (e.g., same number of epochs, batch sizes, and learning–rate schedules), and we treat the resulting estimators as approximate maximum–likelihood estimators. In particular, we assume that training converges to stationary points that lie in a neighbourhood of the (local) maximizers of the corresponding log–likelihoods, so that the usual MLE asymptotics apply.*

Under Assumption 1, the standard MLE central limit theorem and delta method results used in Sec. 3 and Sec. A hold for both the multiclass and collapsed–binary estimators.

A.2. Proof of Lemma 1

Lemma 1 (Score Projection [43, 47])**.** *Let $Z = g(Y)$ be a deterministic coarsening of the label Y. Then, under regularity conditions ensuring that differentiation and summation interchange,*

$$\mathbb{E}_\theta[s_Y(\theta) \mid Z, X] = s_Z(\theta).$$

Proof: Since $Z = g(Y)$, we can write:

$$\begin{aligned}\mathbb{E}[s_Y(\theta) \mid Z, X] &= \sum_{y:g(y)=Z} \nabla_\theta \log p_\theta(y \mid X) \frac{p_\theta(y \mid X)}{p_\theta(Z \mid X)} \\ &= \frac{\sum_{y:g(y)=Z} \nabla_\theta p_\theta(y \mid X)}{p_\theta(Z \mid X)} \\ &= \nabla_\theta \log p_\theta(Z \mid X) \\ &= s_Z(\theta)\end{aligned}$$

□

A.3. Proof of Theorem 1

Theorem 1 (Label coarsening reduces expected Fisher information.)**.** *For every $\theta \in \Theta$,*

$$\mathcal{I}_Y(\theta) = \mathcal{I}_Z(\theta) + \mathbb{E}_\theta\big[\operatorname{Var}(s_Y(\theta) \mid Z, X)\big] \succeq \mathcal{I}_Z(\theta)$$

and therefore $\mathcal{I}_Z(\theta) \preceq \mathcal{I}_Y(\theta)$ in the Loewner (positive–semidefinite) order. Equality holds iff $s_Y(\theta)$ is completely determined by (Z, X), i.e. no variation in the complete–data score remains once Z is known.

Proof: From the law of total variance, for any random vector U and any conditional variable V,

$$\mathbb{E}[UU^T] = \mathbb{E}\Big[\mathbb{E}[U \mid V]\,\mathbb{E}[U \mid V]^T\Big] + \mathbb{E}[\operatorname{Var}(U \mid V)]$$

substituting $U = s_Y(\theta)$ and $V = (Z, X)$ we get.

$$\begin{aligned}\mathbb{E}_\theta[s_Y(\theta)s_Y(\theta)^T] = \\ \mathbb{E}_\theta\Big[\mathbb{E}_\theta[s_Y(\theta) \mid (Z,X)]\,\mathbb{E}_\theta[s_Y(\theta) \mid (Z,X)]^T\Big] \\ + \mathbb{E}_\theta[\operatorname{Var}(s_Y(\theta) \mid (Z,X))]\end{aligned}$$

From Lemma 1, we get that $\mathbb{E}[s_Y(\theta) \mid Z, X] = s_Z(\theta)$.

$$\begin{aligned}\mathbb{E}_\theta[s_Y(\theta)s_Y(\theta)^T] = \mathbb{E}_\theta[s_Z(\theta)s_Z(\theta)^T] \\ + \mathbb{E}_\theta[\operatorname{Var}(s_Y(\theta) \mid (Z,X))]\end{aligned}$$

From the definitions of expect Fisher Information (from Sec. 3.1), we get:

$$\mathcal{I}_Y(\theta) = \mathcal{I}_Z(\theta) + \mathbb{E}_\theta[\operatorname{Var}(s_Y(\theta) \mid (Z,X))]$$

Additionally, we can define the missing information as

$$\mathcal{I}_{\text{missing}} = \mathbb{E}_\theta[\operatorname{Var}(s_Y(\theta) \mid (Z,X))]$$

□

A.4. Proof of Corollary 1

Corollary 1 (Asymptotic Efficiency of the Multiclass MLE.)**.** *Under standard regularity and correct-specification assumptions at the true parameter θ^*, the multiclass MLE is (weakly) more statistically efficient than the collapsed-binary MLE for any smooth functional of θ. Equivalently, for any differentiable quantity $g(\theta)$ of interest, the asymptotic estimation variance of $g(\hat{\theta}_Y)$ is no greater than $g(\hat{\theta}_Z)$.*

Proof: Under correct specification and standard regularity at θ^*, the MLE is asymptotically normal with covariance equal to the Fisher Information as MLE Central Limit Theorem converges in distribution:

$$\begin{aligned}\sqrt{n}(\hat{\theta}_Y - \theta^*) &\xrightarrow{d} \mathcal{N}(0, \mathcal{I}_Y(\theta^*)^{-1}) \\ \sqrt{n}(\hat{\theta}_Z - \theta^*) &\xrightarrow{d} \mathcal{N}(0, \mathcal{I}_Z(\theta^*)^{-1})\end{aligned}$$

From Theorem 1, we have $\mathcal{I}_Y(\theta) \succeq \mathcal{I}_Z(\theta)$, if $A \succeq B \succ 0$ then $A^{-1} \preceq B^{-1}$ Loewner order reverses under inversion. So,

$$\mathcal{I}_Y(\theta^*)^{-1} \preceq \mathcal{I}_Z(\theta^*)^{-1}$$

The result so far is about quality of the parameter (weight) estimates. We are interested in the quality of our predictions. To make that final connection, we use the delta method. Consider any smooth function $g(\theta)$ - for instance, the target-class posterior $\tau(\theta) = \eta_c(x_0; \theta)$, where x_0 denotes an arbitrary fixed input. By the delta method,

$$\sqrt{n}(g(\hat{\theta}) - g(\theta_0)) \xrightarrow{d} \mathcal{N}(0, G\Sigma G^T)$$

where $G = \nabla_\theta g(\theta_0)$, θ_0 is a parameter with non-singular Fisher information, and Σ is the asymptotic covariance of the corresponding MLE. Therefore, each of the models can be written as:

$$\begin{aligned}\sqrt{n}(\tau(\hat{\theta}_Y) - \tau(\theta^*)) &\xrightarrow{d} \mathcal{N}(0, \nabla_\tau^T \mathcal{I}_Y(\theta^*)^{-1} \nabla_\tau) \\ \sqrt{n}(\tau(\hat{\theta}_Z) - \tau(\theta^*)) &\xrightarrow{d} \mathcal{N}(0, \nabla_\tau^T \mathcal{I}_Z(\theta^*)^{-1} \nabla_\tau)\end{aligned}$$

From Theorem 1, we can show that,

$$\nabla_\tau^T \mathcal{I}_Y(\theta^*)^{-1} \nabla_\tau \leq \nabla_\tau^T \mathcal{I}_Z(\theta^*)^{-1} \nabla_\tau$$

So,

$$\begin{aligned}\operatorname{Var}\Big(\sqrt{n}\,(\tau(\hat{\theta}_Y) - \tau(\theta^*))\Big) \\ \leq \operatorname{Var}\Big(\sqrt{n}\,(\tau(\hat{\theta}_Z) - \tau(\theta^*))\Big)\end{aligned}$$

Hence the estimates for $p_\theta(Y = c \mid x_0)$ are asymptotically tighter under multiclass training.

□

A.5. Proof of Proposition 1

Proposition 1 (Softmax Expected Fisher Information Decomposition.)**.** *Consider a softmax model with logits $f(x;\theta) \in \mathbb{R}^K$ and class probabilities $\eta = softmax(f)$. Let $J_f(x;\theta) = \partial f(x;\theta)/\partial\theta \in \mathcal{R}^{K\times p}$ denote the Jacobian of the logits with respect to the model parameters and let $e_c \in \mathbb{R}^K$ denote the c-th standard basis (one-hot) vector. Then for any fixed input $X = x$:*

$$\mathcal{I}_Y(\theta \mid X = x) = J_f^T(Diag(\eta) - \eta\eta^T)J_f$$

is the conditional expected Fisher information for the multiclass likelihood, and

$$\mathcal{I}_Z(\theta \mid X = x) = J_f^T\left(\frac{\eta_c}{(1-\eta_c)}(e_c - \eta)(e_c - \eta)^T\right)J_f$$

is the corresponding quantity for the collapsed-binary label. Hence,

$$\mathcal{I}_Y(\theta \mid X = x) = \mathcal{I}_Z(\theta \mid X = x) + (1-\eta_c)J_f^T(Diag(\pi) - \pi\pi^T)J_f \succeq \mathcal{I}_Z(\theta \mid X = x)$$

where $\pi_k = \eta_k/(1-\eta_c)$ for $k \neq c$ and $\pi_c = 0$.

Proof: Let $f(x;\theta) \in \mathbb{R}^K$ be the logits and $\eta = \text{softmax(f)}$. For one observation $(X = x, Y = y)$ the log-likelihood for the datapoint is

$$\ell(\theta; y; x) = \log \eta_y = f_y - \log\sum_{j=1}^{K} e^{f_j}$$

To get the conditional expected fisher information let $J_f(x;\theta) = \nabla_\theta f(x;\theta)$ be the Jacobian of the logits with respect to θ. Then by the chain rule,

$$s_Y(\theta; y; x) = \nabla_\theta \ell = \left(\frac{\partial f}{\partial \theta}\right)^T \nabla_f \ell = J_f^T \nabla_f \log \eta_y$$

The softmax derivatives are,

$$\frac{\partial \eta_y}{\partial f_j} = \eta_y(\delta_{yj} - \eta_j)$$

$$\implies \nabla_f \log \eta_y = \frac{1}{\eta_y}\nabla_f \eta_y = \delta_y - \eta = e_y - \eta$$

δ_{yj} is the Kronecker delta and e_y is the y-th standard basis vector. So the score function can be written as

$$s_Y(\theta; y; x) = J_f^T(e_y - \eta)$$

So the conditional expectation at a point $(X = x, Y = y)$ is

$$\mathcal{I}_Y(\theta \mid x) = \sum_{y=1}^{k} \eta_y [s_Y(\theta; y; x) s_Y(\theta; y; x)^T]$$

$$\mathcal{I}_Y(\theta \mid x) = \sum_{y=1}^{k} \eta_y J_f^T (e_y - \eta)(e_y - \eta)^T J_f$$

The inner sum is the covariance of the one-hot vector e_Y with mean η

$$\sum_{y=1}^{k} \eta_y e_y e_y^T - \eta\eta^T = \text{Diag}(\eta) - \eta\eta^T$$

Therefore,

$$\mathcal{I}_Y(\theta \mid x) = J_f^T(\text{Diag}(\eta) - \eta\eta^T)J_f$$

For the conditional expected fisher information for the collapsed Bernoulli case we start with the log-likelihood.

$$\ell_Z(\theta; z; x) = z \log q + (1-z)\log(1-q)$$

Where $q = q(x;\theta)$. Differentiating w.r.t. θ we get

$$\nabla_\theta \ell_Z = \left(\frac{z}{q} - \frac{1-z}{1-q}\right)\nabla_\theta q = \frac{z-q}{q(1-q)}\nabla_\theta q$$

Therefore the score function for the collapsed Bernoulli is

$$s_Z(\theta; z; x) = \frac{z-q}{q(1-q)}\nabla_\theta q$$

Therefore the score function for the collapsed Bernoulli is

$$s_Z(\theta; z; x) = \frac{z-q}{q(1-q)}\nabla_\theta q$$

So the expected Fisher Information at x is

$$\mathcal{I}_Z(\theta \mid X = x) = \mathbb{E}_\theta[s_Z(\theta) s_Z(\theta)^T \mid X = x]$$

$$\mathcal{I}_Z(\theta \mid X = x) = \frac{\mathbb{E}_\theta[(Z-q)^2 \mid x]}{q^2(1-q)^2}(\nabla_\theta q)(\nabla_\theta q)^T$$

The numerator term is $\mathrm{Var}(Z \mid X = x) = \mathbb{E}[Z \mid X] - (\mathbb{E}[Z \mid X])^2 = q - q^2 = q(1-q)$

$$\mathcal{I}_Z(\theta \mid X = x) = \frac{1}{q(1-q)} (\nabla_\theta q)(\nabla_\theta q)^T$$

To get $\nabla_\theta q$, we take $q = \eta_c(f)$

$$\frac{\partial \eta_c}{\partial f_j} = \eta_c(\delta_{cj} - \eta_j) \implies \nabla_f \eta_c = \eta_c(e_c - \eta)$$

Therefore,

$$\nabla_\theta q = \left(\frac{\partial f}{\partial \theta}\right)^T \nabla_f \eta_c = J_f^T(\eta_c(e_c - \eta))$$

Since $q = \eta_c$

$$\mathcal{I}_Z(\theta \mid X = x) = J_f^T \frac{1}{\eta_c(1-\eta_c)} (\eta_c(e_c - \eta))(\eta_c(e_c - \eta)^T) J_f$$

$$\mathcal{I}_Z(\theta \mid X = x) = J_f^T \left(\frac{\eta_c}{(1-\eta_c)} (e_c - \eta)(e_c - \eta)^T\right) J_f$$

To relate the two, express the class probabilities as $\eta = [\,\eta_c,\, (1-\eta_c)\pi\,]$, where $\pi_k = \eta_k/(1-\eta_c)$ for $k \neq c$ and $\pi_c = 0$. Substituting this form into the multiclass curvature in logit space gives

$$\begin{aligned} &\mathrm{Diag}(\eta) - \eta\eta^\top \\ &= \frac{\eta_c}{1-\eta_c}(e_c - \eta)(e_c - \eta)^\top + (1-\eta_c)\big(\mathrm{Diag}(\pi) - \pi\pi^\top\big). \end{aligned}$$

Multiplying by $J_f^\top$ and J_f on both sides yields the desired decomposition:

$$\begin{aligned} &\mathcal{I}_Y(\theta \mid X = x) = \mathcal{I}_Z(\theta \mid X = x) \\ &+ (1-\eta_c)\, J_f^\top \big(\mathrm{Diag}(\pi) - \pi\pi^\top\big) J_f \succeq \mathcal{I}_Z(\theta \mid X = x). \end{aligned}$$

The second term is positive semi–definite and quantifes the information lost by collapsing all non–target classes into a single background label.

□

A.6. Limitations for Classification Tasks

While the proposed theory extends naturally to segmentation, it becomes less reliable in standard classification settings due to severe class imbalance. When K classes are mapped into a single "target vs. rest" binary task, the positive class often occupies a vanishing fraction of the data, yielding posterior probabilities $\eta_c(x;\theta)$ that are extremely close to 0 or 1. In this regime, the Bernoulli Fisher term $\eta_c(1-\eta_c)$ degenerates, leading to ill-conditioned or singular information matrices. As a result, the asymptotic efficiency and variance comparisons derived under balanced or well-behaved posteriors no longer hold. In practice, this means that for highly imbalanced classification datasets, coarsening amplifies numerical instability and the theoretical ordering $\mathcal{I}_Y(\theta) \succeq \mathcal{I}_Z(\theta)$ may not provide a useful description of estimator behavior.

A.7. Note on Natural Images vs 3D Medical Images

The observed strength of the proposed framework in 3D medical imaging tasks can be attributed to the high variability and structured heterogeneity of the *background* in volumetric scans. Unlike 2D natural-image segmentation, where the "non-target" regions are often homogeneous and semantically uninformative (e.g., sky, wall, road), medical volumes contain multiple anatomical structures—organs, vessels, and tissues—each contributing distinct non-target gradients. This diversity amplifies the within-rest Fisher term in our decomposition, yielding a substantial gap between the multiclass and collapsed-binary information. Consequently, modeling each anatomical class preserves rich curvature directions in parameter space, improving statistical efficiency and convergence. In contrast, for dichotomous segmentation tasks in natural images or datasets with relatively uniform backgrounds, the variation is minimal, making the binary and multiclass formulations nearly equivalent.

B. Heuristics for selecting auxiliary classes

In practice, the choice of auxiliary support structures plays an important role in maximizing the benefits of BackSplit. We outline several practical heuristics that can guide this selection.

1. Select the organ containing the lesion. The most effective auxiliary structure is typically the organ in which the lesion resides, as it fully encloses the target region. Including this structure allows the model to learn fine-grained variations in local tissue appearance, which directly improves boundary precision and reduces ambiguity.

2. Include adjacent or surrounding organs. Structures that spatially border the lesion serve as anatomical anchors. These surrounding organs provide contextual cues that help the model disambiguate lesion boundaries and reduce drift into neighboring regions. This is particularly useful for lesions located near organ interfaces.

3. Add organs prone to false positives. A third useful strategy is to incorporate organs whose appearance may cause confusion with the target lesion. These structures often contain components or textures that resemble the lesion class, leading to false positives in standard training. By ex-

plicitly modeling these tissues, the network learns discriminative cues that reduce such errors and improve robustness.

These heuristics offer a practical starting point for selecting auxiliary structures and can be adapted based on dataset characteristics, anatomical complexity, and known failure modes of baseline models.

B.1. Limitations

A key assumption of BackSplit is the availability of auxiliary background structures. While such annotations are increasingly accessible in common modalities such as CT, MRI, and X-ray, often via large pretrained models (e.g., TotalSegmentator [11]), this may not hold for less common modalities or domains lacking broad segmentation models. In these cases, obtaining auxiliary structures can be more challenging. To mitigate this, we suggest leveraging interactive segmentation methods to efficiently generate approximate background annotations, which, as demonstrated in our experiments, can still provide meaningful performance gains despite being noisy.

B.2. Future Work on Heuristics

While BackSplit demonstrates consistent improvements across diverse settings, the choice of auxiliary structures remains an open question that warrants deeper theoretical and empirical investigation. In particular, it is unclear how the proximity and relevance of auxiliary structures influence performance. For example, when no structure directly encloses the target, it remains to be understood whether supervising nearby or partially overlapping structures is sufficient. Additionally, factors such as target size and anatomical context may modulate the observed gains, e.g., whether smaller lesions benefit more from organ-level supervision than larger ones. Addressing these questions would provide a more principled understanding of how to optimally select auxiliary structures and further improve the effectiveness of the proposed paradigm.

C. Model Weights and Code

Code and pretrained weights can be found at this link github.com/rachitsaluja/BackSplit. All models are implemented in PyTorch [48], and inference can be performed directly using the nnU-Net [28] library.

D. Training Configuration and Hyperparameters

The following section summarizes the training configurations used for all models.

D.1. U-Nets

All U-Net [50] models were trained using the nnU-Net [28] 3D full-resolution configuration. A batch size of 2 and varying patch sizes were used across experiments, depending on dataset-specific memory requirements. All data were normalized according to imaging modality (CT or MRI) and subsequently resampled to the standardized nnU-Net resolution. The network backbone followed nnU-Net's generic 3D U-Net design, employing InstanceNorm3d and LeakyReLU nonlinearities.

Training used nnU-Net's default Dice + Cross-Entropy loss with deep supervision enabled. Optimization was performed with stochastic gradient descent (momentum 0.99), an initial learning rate of 0.01, weight decay of 3×10^{-5}, and the PolyLR learning rate schedule. Each model was trained for 1000 epochs, with 250 iterations per epoch. All experiments were conducted on an NVIDIA A100 GPU.

D.2. ResEncU-Net

The ResEncU-Net [29] models follow the same training configuration as the U-Nets, differing only in network architecture. We use the Residual Encoder U-Net with the Medium configuration, as implemented in the nnU-Net framework. All optimization settings, normalization procedures, and training schedules remain identical to those described in the U-Net subsection.

D.3. SegResNet

For the SegResNet experiments, we use the MONAI [4] SegResNet [46] implementation wrapped inside the nnU-Net framework, keeping all data preprocessing, normalization, and resampling steps identical to the nnU-Net. The primary differences lie in the network architecture and optimizer settings. Unlike the standard nnU-Net configuration, which uses SGD, SegResNet is optimized using Adam with a learning rate of 1×10^{-4}, a weight decay of 1×10^{-5}, an ε value of 1×10^{-5} and a PolyLR learning rate schedule. Models were trained using the default Dice + Cross-Entropy loss without deep supervision enabled.

D.4. SwinUNETR

For the transformer-based experiments (as shown in Sec. G.5), we employ the 3D SwinUNETR [22] architecture from MONAI [4], integrated into the nnU-Net framework while keeping all preprocessing, normalization, and resampling procedures consistent with the standard nnU-Net pipeline. The main differences arise in the network architecture and optimization strategy.

SwinUNETR is trained using the AdamW optimizer with an initial learning rate of 8×10^{-4}, a weight decay of 0.01, and an ε value of 1×10^{-5}. To better suit transformer training dynamics, we replace nnU-Net's default PolyLR schedule with a CosineAnnealingLR scheduler, setting $T_{\max}$ to the total number of training epochs and a minimum learning rate of 1×10^{-6}.

All other training settings—including loss function

Auxiliary Task (Support Structure)	Dice ↑ ⊕ 7 clicks	Dice ↑ ⊕10 clicks
(a) KiTS23		
Kidney	0.7420	0.7419
Tumor	0.5457	0.5428
(b) PANTHER-MR		
Pancreas	0.6779	0.6823
(c) NSCLC-Rad		
Esophagus	0.7155	0.7256
Heart	0.8408	0.8390
Lung Left	0.9606	0.9687
Lung Right	0.9627	0.9688
Spinal Cord	0.8244	0.8263
(d–e) AutoPET (MSWAL)		
Aorta	0.8421 (0.9335)	0.8506 (0.9335)
Gall Bladder	0.6765 (0.8003)	0.6750 (0.8003)
Kidney Left	0.9333 (0.9585)	0.9334 (0.9585)
Kidney Right	0.9416 (0.9589)	0.9396 (0.9589)
Liver	0.9388 (0.9662)	0.9360 (0.9662)
Pancreas	0.7228 (0.8497)	0.7374 (0.8497)
Postcava	0.7648 (0.7724)	0.7804 (0.7724)
Spleen	0.9379 (0.9655)	0.9353 (0.9655)
Stomach	0.8718 (0.8895)	0.8791 (0.8895)

Table 5. Simulated interactive segmentation results using 7 and 10 user clicks (⊕) across five datasets for generating support structures. For KiTS, PANTHER-MR, and NSCLC-Rad, ground-truth masks were used to randomly sample 7 or 10 seed points, followed by segmentation using nnInteractive. For AutoPET and MSWAL, the seed points were sampled from model-derived organ segmentations.

(Dice + Cross-Entropy), batch size, and epoch count—are kept consistent with the U-Net and SegResNet configurations unless otherwise noted.

D.5. Minimal Parameter Overhead of BackSplit

An advantage of BackSplit is its minimal parameter overhead. Even in the most demanding setting—where the largest number of auxiliary classes is added—the increase in model parameters is approximately 0.02%. This negligible overhead ensures that BackSplit remains practical, adding virtually no computational burden while providing strong performance gains.

E. Interactive Segmentation Performance

Table 5 reports the segmentation quality of auxiliary support structures generated using nnInteractive [30] with 7 and 10 positive clicks. Although the resulting masks exhibit only modest accuracy, they are sufficiently informative for BackSplit to produce strong performance gains in downstream lesion segmentation tasks. This demonstrates that BackSplit remains effective even when auxiliary labels are noisy and derived from lightweight interactive segmentation rather than precise manual annotations.

Auxiliary Task (Support Structure)	Dice ↑ TotalSegmentator	Dice ↑ VIBESegmentator
(a–b) AutoPET (MSWAL)		
Aorta	0.9228 (0.9216)	0.7748 (0.8211)
Gall Bladder	0.8307 (0.7612)	0.3875 (0.4609)
Kidney Left	0.9192 (0.9427)	0.9266 (0.8799)
Kidney Right	0.9214 (0.9433)	0.9282 (0.8723)
Liver	0.9761 (0.9719)	0.571 (0.8673)
Pancreas	0.8834 (0.8841)	0.6592 (0.697)
Postcava	0.9062 (0.8614)	0.8275 (0.7092)
Spleen	0.9631 (0.9578)	0.848 (0.8284)
Stomach	0.9473 (0.9377)	0.8728 (0.8036)

Table 6. Auxiliary segmentation performance from large pretrained models on AutoPET and MSWAL. Reported Dice scores reflect organ segmentations generated by TotalSegmentator and VIBESegmentator, with values in parentheses indicating corresponding performance on MSWAL. These automatically derived masks are used as support structures for BackSplit.

F. Large Models Segmentation Performance

To further assess the robustness of BackSplit, we evaluate auxiliary segmentations produced by large pretrained models—TotalSegmentator [11] and VIBESegmentator [20]—on the AutoPET and MSWAL datasets. These models are widely used for comprehensive whole-body and abdominal organ parsing and provide a strong reference point for high-quality automatic segmentation. We compare their outputs with our model-derived segmentations obtained from a U-Net trained on the AbdomenAtlas-Mini1.0 [49] dataset. As shown in Table 6, the pretrained models achieve strong organ segmentation accuracy on AutoPET and MSWAL.

G. BackSplit Performance during different training settings

In this section, we evaluate the effectiveness of BackSplit across different training conditions and architectural variations to further demonstrate its robustness as a general paradigm.

G.1. Evaluating BackSplit Under Varying Batch Sizes

To assess the robustness of BackSplit, we first evaluate its performance under varying batch sizes. Given typical GPU memory constraints for 3D medical segmentation models, we consider realistic batch sizes of 2 and 4, and conduct experiments on the KiTS23 [25], PANTHER [3],

Method	Patch Size Variation 1			Patch Size Variation 2		
	Dice ↑	HD-95 ↓	NSD ↑	Dice ↑	HD-95 ↓	NSD ↑
(a) Dataset: KiTS23 with Target Class = {Cyst}						
Patch Size Value	[128,128,128]			[160,128,128]		
Regular Training	0.2033	425.3273	0.1906	0.2117	422.4271	0.1981
BackSplit	0.5297	249.5371	0.6703	0.5295	251.8412	0.6732
(b) Dataset: PANTHER-MR with Target Class = {Tumor}						
Patch Size Value	[48,192,224]			[48,224,224]		
Regular Training	0.3828	157.3470	0.2320	0.4409	152.3861	0.2623
BackSplit	0.4906	54.2010	0.2796	0.4732	52.0249	0.2718
(c) Dataset: NSCLC-Radiomics with Target Class = {GTV}						
Patch Size Value	[64,192,192]			[32,160,192]		
Regular Training	0.5279	140.2698	0.4049	0.4984	164.0937	0.3771
BackSplit	0.5862	124.2557	0.4533	0.5724	123.7844	0.4431

Table 7. Single-fold evaluation of BackSplit vs. regular training using a U-Net backbone across two patch-size configurations (batch size fixed at 2). Results are shown for (a) KiTS23, (b) PANTHER-MR, and (c) NSCLC-Radiomics. Across all datasets and both patch-size settings, BackSplit consistently outperforms regular training on Dice, HD-95, and NSD metrics, demonstrating robustness to changes in patch size.

Method	Batch Size = 2			Batch Size = 4		
	Dice ↑	HD-95 ↓	NSD ↑	Dice ↑	HD-95 ↓	NSD ↑
(a) Dataset: KiTS23 with Target Class = {Cyst}						
Regular Training	0.2033	425.3273	0.1906	0.2630	404.6592	0.2624
BackSplit	0.5297	249.5371	0.6703	0.5261	249.6824	0.6716
(b) Dataset: PANTHER-MR with Target Class = {Tumor}						
Regular Training	0.3828	157.3470	0.2320	0.3636	202.4875	0.2129
BackSplit	0.4906	54.2010	0.2796	0.4553	59.1720	0.2696
(c) Dataset: NSCLC-Radiomics with Target Class = {GTV}						
Regular Training	0.5279	140.2698	0.4049	0.5530	136.1963	0.4270
BackSplit	0.5862	124.2557	0.4533	0.5549	127.0088	0.4353

Table 8. Single-fold evaluation of BackSplit vs. regular training using a U-Net backbone across two batch sizes (2 and 4). Results are shown for (a) KiTS23, (b) PANTHER-MR, and (c) NSCLC-Radiomics. In all datasets and for both batch-size settings, BackSplit consistently outperforms regular training across Dice, HD-95, and NSD metrics, demonstrating robustness to changes in batch size.

and NSCLC-Radiomics [2] datasets. In all cases, BackSplit consistently outperforms regular training, demonstrating its effectiveness irrespective of batch size (as shown in Tab. 8). All experiments were performed on a single fold using U-Net [50] with the nnU-Net framework [28].

Method	with Deep Supervision [35]			without Deep Supervision [35]		
	Dice ↑	HD-95 ↓	NSD ↑	Dice ↑	HD-95 ↓	NSD ↑
(a) Dataset: KiTS23 with Target Class = {Cyst}						
Regular Training	0.2033	425.3273	0.1906	0.1339	441.7240	0.1300
BackSplit	0.5297	249.5371	0.6703	0.4758	281.8879	0.6137
(b) Dataset: PANTHER-MR with Target Class = {Tumor}						
Regular Training	0.3828	157.3470	0.2320	0.4412	179.4845	0.2590
BackSplit	0.4906	54.2010	0.2796	0.4776	46.5688	0.2905
(c) Dataset: NSCLC-Radiomics with Target Class = {GTV}						
Regular Training	0.5279	140.2698	0.4049	0.5059	158.2856	0.3765
BackSplit	0.5862	124.2557	0.4533	0.5829	115.2792	0.4564

Table 9. Single-fold evaluation of BackSplit with and without Deep Supervision [35] using a U-Net backbone. Results are shown for (a) KiTS23, (b) PANTHER-MR, and (c) NSCLC-Radiomics. Across all datasets and under both training conditions, BackSplit consistently outperforms regular training on Dice, HD-95, and NSD metrics, demonstrating that its benefits hold regardless of whether Deep Supervision is employed.

G.2. Evaluating BackSplit Under Varying Patch Sizes

To further evaluate the robustness of BackSplit, we examine its performance under varying patch sizes for each dataset. Patch size is a critical factor in 3D medical segmentation due to differences in anatomical scale and memory constraints. We conduct experiments on the KiTS23 [25], PANTHER [3], and NSCLC-Radiomics [2] datasets using multiple patch-size configurations. Across all settings, BackSplit consistently outperforms regular training, demonstrating its stability with respect to patch-size variation (as shown in Tab. 7). All experiments were performed on a single fold using U-Net [50] within the nnU-Net framework [28] keeping batch size constant, at a size of two.

G.3. BackSplit with and without Deep Supervision

We also assess the impact of deep supervision [35], a mechanism used in the nnU-Net framework to provide intermediate supervision from the layers and enhance performance in 3D segmentation networks. For each dataset, we compare models trained with and without deep supervision while keeping all other training settings fixed. In both configurations, BackSplit consistently outperforms standard training, indicating that its benefits are complementary to deep supervision rather than dependent on it (as shown in Tab. 9). All experiments were conducted on a single fold using U-Net [50] within the nnU-Net framework [28].

Method	3D U-Net [28, 50]			2D U-Net [28, 50]		
	Dice ↑	HD-95 ↓	NSD ↑	Dice ↑	HD-95 ↓	NSD ↑
(a) Dataset: KiTS23 with Target Class = {Cyst}						
Regular Training	0.2033	425.3273	0.1906	0.3234	352.3277	0.4231
BackSplit	0.5297	249.5371	0.6703	0.4111	313.9846	0.5126
(b) Dataset: PANTHER-MR with Target Class = {Tumor}						
Regular Training	0.3828	157.3470	0.2320	0.2349	265.2399	0.1312
BackSplit	0.4906	54.2010	0.2796	0.3611	137.2006	0.2031
(c) Dataset: NSCLC-Radiomics with Target Class = {GTV}						
Regular Training	0.5279	140.2698	0.4049	0.4695	102.8473	0.3676
BackSplit	0.5862	124.2557	0.4533	0.4753	125.5158	0.3728

Table 10. Single-fold evaluation of BackSplit using 3D and 2D U-Net architectures on (a) KiTS23, (b) PANTHER-MR, and (c) NSCLC-Radiomics. Across all datasets and for both 3D and 2D models, BackSplit consistently outperforms regular training on Dice, HD-95, and NSD metrics. These results demonstrate that the benefits of the BackSplit paradigm hold irrespective of network dimensionality.

G.4. BackSplit Performance on 2D U-Net Architectures

We additionally evaluate BackSplit using both 3D U-Net and 2D U-Net architectures to assess its effectiveness across different network dimensionalities. While 3D models typically capture richer volumetric context and 2D models offer computational efficiency, BackSplit yields consistent performance improvements in both settings. This demonstrates that the paradigm is not tied to a specific architectural dimensionality and remains effective whether the model processes full 3D volumes or individual 2D slices (as shown in Tab. 10). All experiments were performed on a single fold using implementations with the nnU-Net framework [28] using their "3d_fullres" and "2d" configurations.

G.5. BackSplit Performance on Transformer-based Architectures

We also evaluate BackSplit on transformer-based architectures, despite prior work indicating that such models often underperform compared to convolutional counterparts in medical image segmentation [29]. In particular, we employ a 3D SwinUNETR [22]. Even in this setting, BackSplit yields consistent improvements over standard training (as shown in Tab. 11), demonstrating that its benefits extend beyond convolutional models and remain effective for transformer-based architectures. However, as noted in the literature it does not perform as well as U-Nets.

Method	3D U-Net [28, 50]			3D SwinUNETR [22]		
	Dice ↑	HD-95 ↓	NSD ↑	Dice ↑	HD-95 ↓	NSD ↑
(a) Dataset: KiTS23 with Target Class = {Cyst}						
Regular Training	0.2033	425.3273	0.1906	0.1336	438.8709	0.1035
BackSplit	0.5297	249.5371	0.6703	0.3863	309.6816	0.5010
(b) Dataset: PANTHER-MR with Target Class = {Tumor}						
Regular Training	0.3828	157.3470	0.2320	0.3113	163.1136	0.1480
BackSplit	0.4906	54.2010	0.2796	0.3705	59.1148	0.1926
(c) Dataset: NSCLC-Radiomics with Target Class = {GTV}						
Regular Training	0.5279	140.2698	0.4049	0.4936	154.8597	0.3674
BackSplit	0.5862	124.2557	0.4533	0.5344	131.8715	0.3942

Table 11. Single-fold evaluation of BackSplit on 3D U-Net and transformer-based (3D SwinUNETR) architecture across (a) KiTS23, (b) PANTHER-MR, and (c) NSCLC-Radiomics. In all datasets and for both architectures, BackSplit consistently outperforms regular training on Dice, HD-95, and NSD metrics. These results demonstrate that the BackSplit paradigm generalizes beyond CNNs and remains effective for transformer-based segmentation models.

H. Empirical Analysis of BackSplit with Increasing Auxiliary Classes

In this section, we analyze how performance metrics change when the number of auxiliary classes used in BackSplit training is increased linearly, and compare these results against standard training.

For this experiment, we use the AutoPET [19] dataset and incrementally add auxiliary classes from the AbdomenAtlas1.0Mini [49] dataset (with the left and right kidneys added simultaneously as two separate classes). This yields seven distinct models, in addition to the regular training model and the BackSplit model containing all auxiliary structures, as shown in the main paper. All models were trained for a single fold using the U-Net [50] architecture within the nnU-Net framework [28].

We observe that the effect of BackSplit is almost immediate: even adding a single auxiliary structure yields a notable performance gain (as shown in Fig. 4). This is intuitive, as the inclusion of even one anatomical structure provides a meaningful contextual anchor for identifying lesions. Although adding multiple structures shows a mix of positive and negative trends whose underlying causes remain unclear, BackSplit consistently outperforms regular training across all settings. As noted in Sec. 5, identifying which support structures are most beneficial remains an important direction for future work.

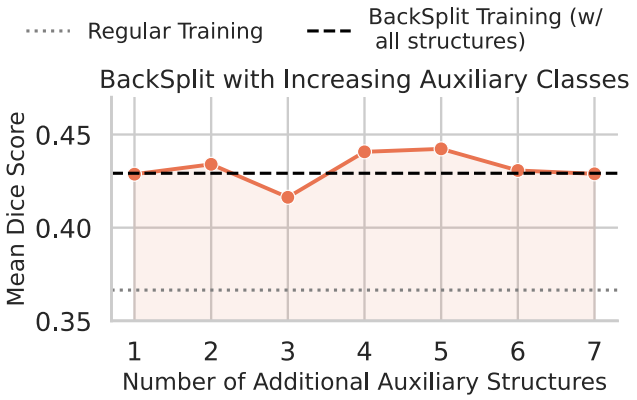

Figure 4. Adding even a single auxiliary structure yields an immediate improvement over regular training, and although incorporating multiple structures introduces mixed behavior with both gains and small drops, performance remains consistently well above the regular-training baseline. This experiment is conducted on the AutoPET dataset by incrementally adding auxiliary classes from AbdomenAtlas1.0Mini using a U-Net backbone.

I. BackSplit vs. Virtual Classes

We also conduct a brief comparison against the virtual classes paradigm [5], which has been explored in classification tasks by adding an additional class only at the softmax layer during training. This approach is hypothesized to encourage more discriminative feature embeddings and thereby improve performance. To emulate this idea in our setting, we train a U-Net with an added empty class, effectively mimicking the virtual class formulation.

We observe a similar effect in which the virtual class formulation yields a modest increase in performance metrics; however, these gains remain substantially lower than those achieved with BackSplit, as shown in Tab. 12. These results suggest that the virtual class approach primarily encourages more discriminative features compared to standard training, whereas BackSplit provides improved predictions through reduced variance and richer anatomical context.

From a practical implementation standpoint, we again use U-Net [50] within the nnU-Net framework [28] to demonstrate this effect on the KiTS23 [25] dataset for a single fold. The virtual class is implemented by adding an additional prediction channel without providing corresponding labels in the dataset, and performance is evaluated solely on the cyst label.

J. Extended Analysis of BackSplit under Fine-Tuning

We further analyze the behavior of BackSplit in the fine-tuning setting, providing additional guidance for readers on how to apply the paradigm effectively for improved performance.

Method	KiTS23 Dice ↑	KiTS23 HD-95 ↓	KiTS23 NSD ↑
Regular Training	0.2033	425.3273	0.1906
Training with Virtual Class	0.2469	414.6015	0.2467
BackSplit	0.5297	249.5371	0.6703

Table 12. Single-fold evaluation on KiTS23 (Target = Cyst) using a U-Net backbone. BackSplit substantially outperforms both regular training and the Virtual Class paradigm across all metrics.

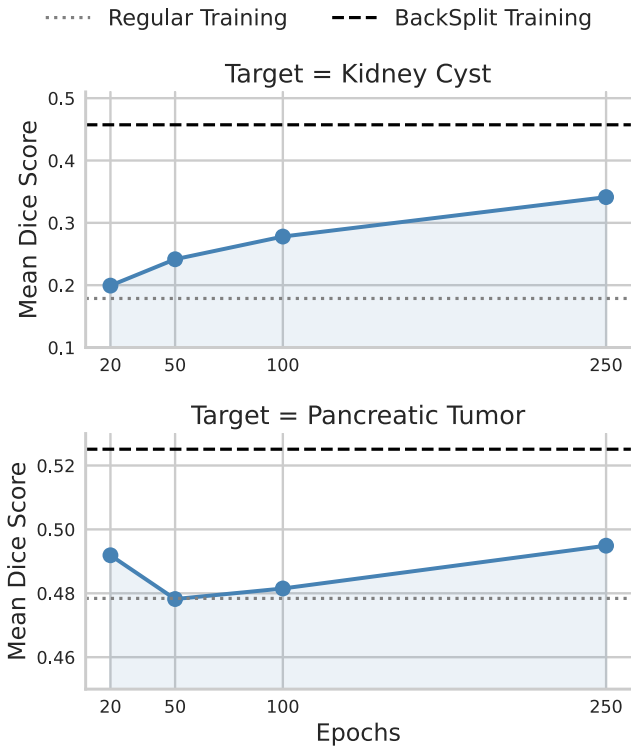

Figure 5. **(Top)** Effect of fine-tuning with BackSplit on a pretrained binary model for kidney cyst segmentation (KiTS23). Mean Dice steadily improves with training epochs, approaching full BackSplit performance. **(Bottom)** Similar trend observed for pancreatic tumor segmentation (PANTHER-MR), where fine-tuning progressively narrows the gap between regular and full BackSplit.

mance.

BackSplit is most effective in settings with a limited number of support structures. We demonstrate this empirically through three experiments on the KiTS23 [25], PANTHER [3], and AutoPET [19] datasets (the latter under two configurations). As shown in the main paper, KiTS23 exhibits clear improvements (Fig. 3 Left and Fig. 5 Top), and PANTHER displays a similar trend (as shown in Fig. 5 Bottom). However, when fine-tuning AutoPET with BackSplit, the model struggles to surpass the baseline performance (as shown in Fig. 6 Top).

We attribute this weaker performance to the experimental setup used for AutoPET, where BackSplit incorporated

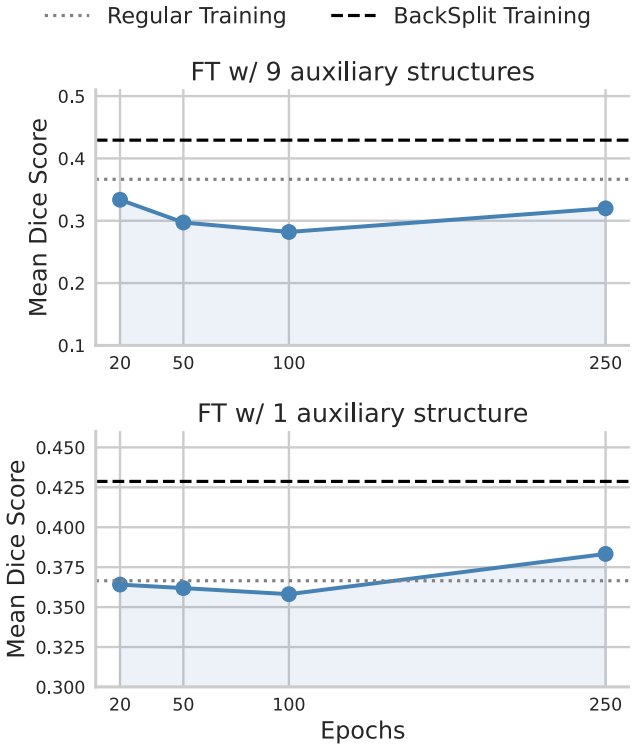

Figure 6. **(Top)** Fine-tuning on AutoPET with 9 auxiliary support structures. When many auxiliary labels are present, the model tends to learn the "easier" auxiliary classes first, delaying progress on the harder lesion target. Under a limited training budget (250 epochs), the model does not sufficiently reach the target class and fails to surpass the regular-training baseline. **(Bottom)** Fine-tuning with only 1 auxiliary structure avoids this and mirrors the behavior seen in KiTS23 and PANTHER, where performance steadily improves with training epochs.

a large number of support structures (nine in total). In such cases, the model tends to learn "easier" auxiliary labels first, delaying progress on the harder target label. With a training budget of only 250 epochs, the model does not sufficiently progress to learning the target class, which leads to the observed underperformance.

To mitigate this effect, we fine-tune the model using only a subset of support structures rather than all nine throughout training. Under this setting, AutoPET follows the same trend observed for KiTS23 and PANTHER (as shown in Fig. 6 Bottom). This provides a practical way to manage training dynamics when dealing with a large number of auxiliary structures. Here, we just add one additional support structure.

All experiments in this section are conducted using U-Net [50] via the nnU-Net [28] framework. For Fig. 5, we report full 5-fold cross-validation results. In contrast, Fig. 6 presents results from a single fold, as AutoPET is a substantially larger dataset and requires significantly more computational resources. This experiment is included primarily to provide empirical support for the claims made in this section. Additionally, following standard practice, we fine-tune the models using a lower initial learning rate of 1×10^{-3}, compared to the typical 1×10^{-2} used in nnU-Net training.

	Random Init		TS-CT [11] Init	
Method	**Regular**	**BackSplit**	**Regular**	**BackSplit**
KiTS23	0.2033	0.5297	0.36	0.4072
NSCLC-Rad	0.5279	0.5862	0.4716	0.5004

Table 13. Comparison of Dice scores for regular training and BackSplit under different initialization strategies: random initialization and pretrained TotalSegmentator-CT (TS-CT) initialization. Results are reported for KiTS23 and NSCLC-Radiomics. BackSplit consistently improves performance over regular training in both settings, demonstrating that its benefits persist when fine-tuning from strong pretrained models as well as when training from scratch.

As initialization from pretrained models becomes increasingly common in medical image segmentation (e.g., TotalSegmentator [11], MultiTalent [54]), we additionally evaluate BackSplit in this setting. Specifically, we initialize from the TotalSegmentator-CT model and fine-tune using the training protocol described above on the KiTS23 and NSCLC-Radiomics datasets. Across both datasets, BackSplit continues to yield consistent performance improvements over standard fine-tuning, demonstrating that its benefits extend beyond training from scratch and remain effective when applied on top of strong pretrained representations(as shown in Tab. 13).

K. Qualitative Performance

In this section, we present qualitative comparisons illustrating the improvements achieved by BackSplit. In Fig. 7, we compare predictions from standard training and BackSplit across multiple architectures, highlighting clearer lesion boundaries under BackSplit. In Fig. 8, we show qualitative results using auxiliary labels derived from interactive segmentation. Despite the noisier supervision, BackSplit continues to produce stable and accurate lesion segmentations, demonstrating its robustness even under imperfect auxiliary annotations.

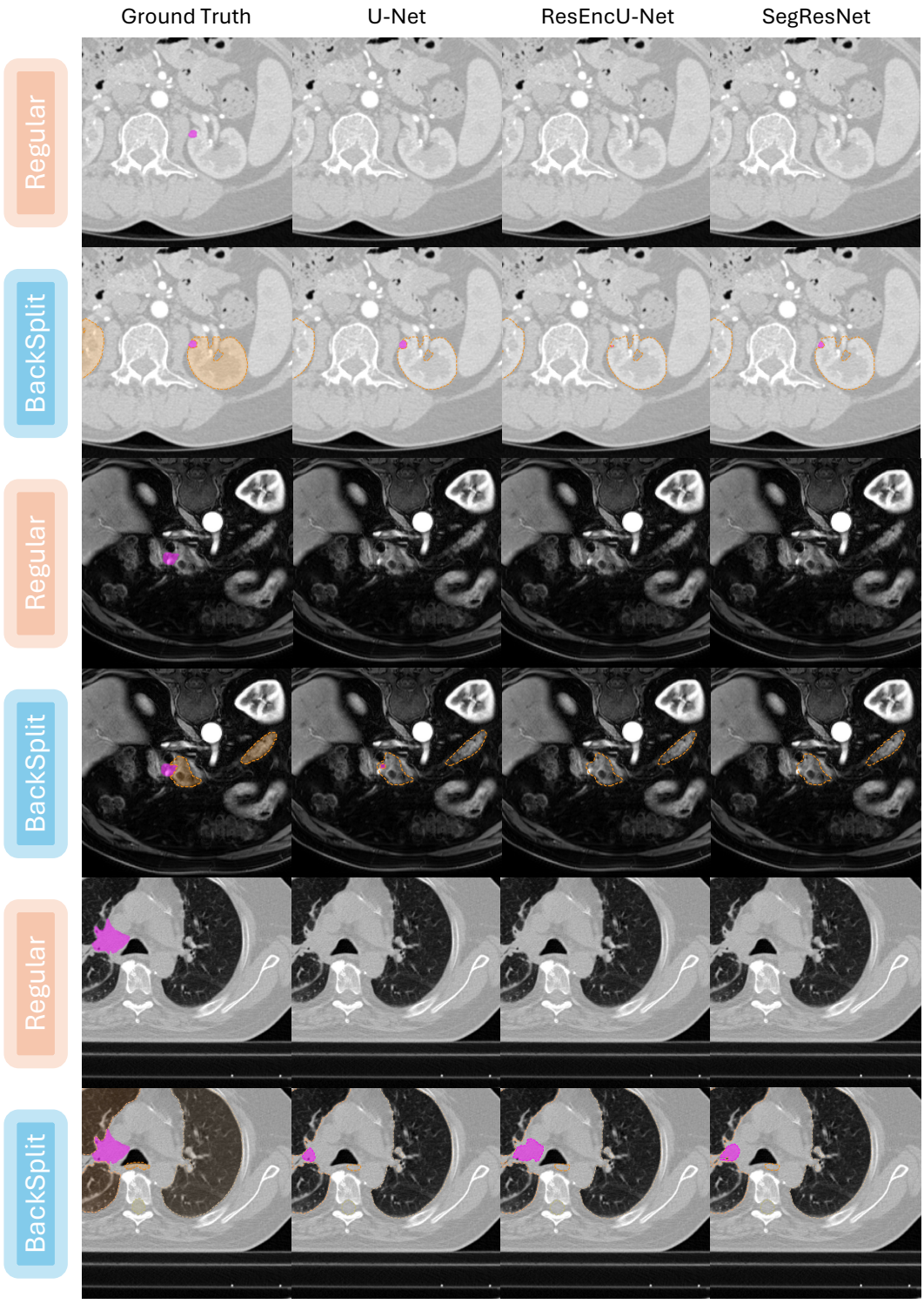

Figure 7. Qualitative comparison on three datasets: KiTS23 (top block), PANTHER-MR (middle block), and NSCLC-Radiomics (bottom block)—across three architectures (U-Net, ResEncU-Net, SegResNet). Each row pair shows regular training (top) and BackSplit (bottom). Regular training frequently misses small or low-contrast lesions, whereas BackSplit, with auxiliary anatomical structures, produces more complete and precise lesion masks. Lesions are shown in pink, while auxiliary support structures predicted under BackSplit are shown in yellow/orange hues, highlighting the additional contextual cues that lead to improved segmentation quality.

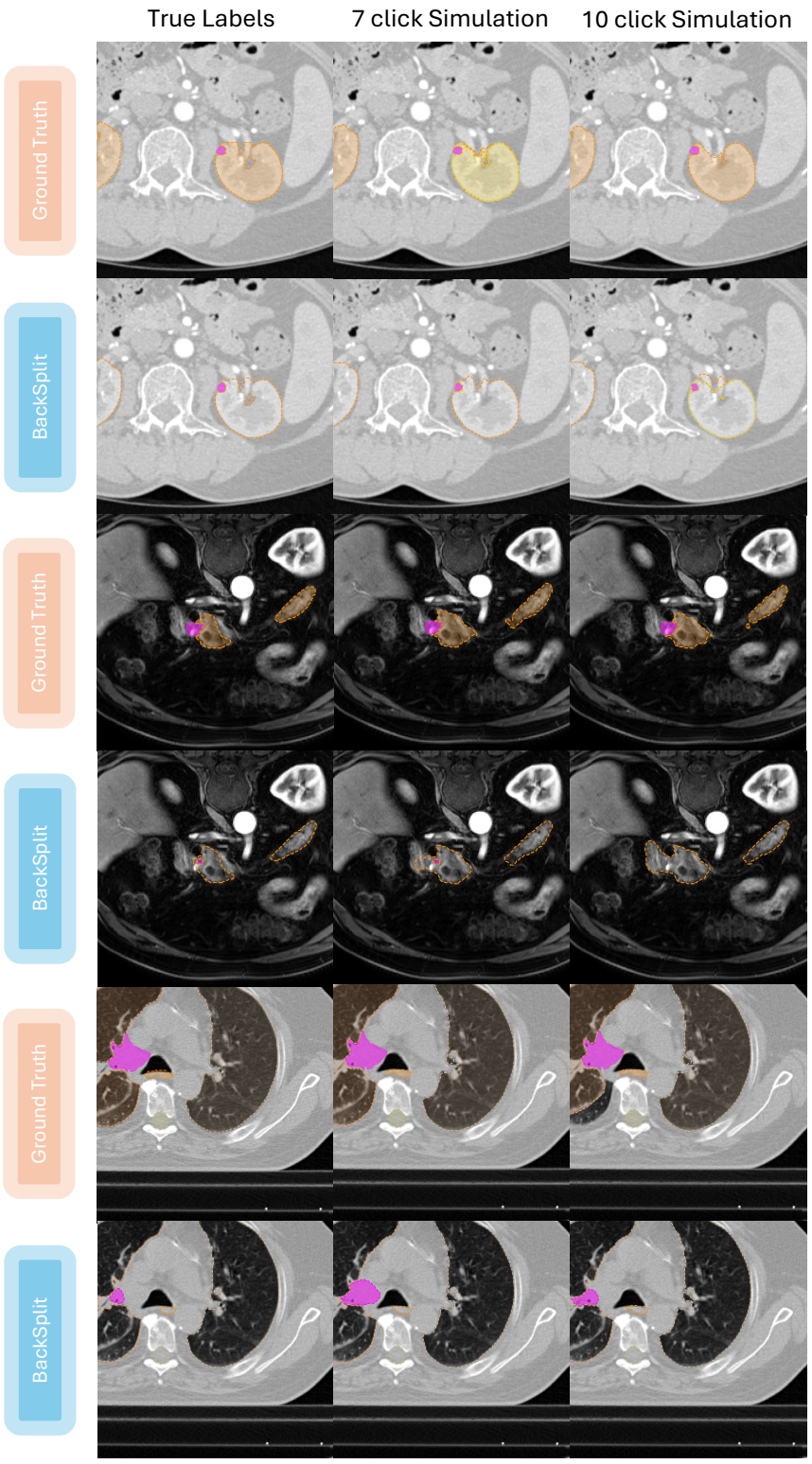

Figure 8. Qualitative comparison of BackSplit using true auxiliary labels versus noisy auxiliary structures generated by nnInteractive with 7 and 10 positive-click simulations. Rows correspond to three datasets: KiTS23, PANTHER-MR, and NSCLC-Radiomics—and each row pair shows Ground Truth auxiliary labels (top) and BackSplit predictions (bottom). U-Net backbone is used for all experiments. Lesions are shown in pink, while auxiliary support structures are shown in yellow/orange hues. Despite substantial noise in the interactively generated auxiliary masks, BackSplit maintains strong lesion segmentation quality, demonstrating robustness to imperfect supervision.