Unsupervised Unlearnable Segmentation via Semantic Structure Disruption

Jiale Cai[1] **Zhihao Li**[1] **Gezheng Xu**[1,*] **Ruiyi Fang**[1] **Hao Zheng**[2]
Zhimin Mei[1] **Song Tang**[3] **Charles Ling**[1] **Boyu Wang**[1,4]

[1] Western University [2] Shenzhen Campus of Sun Yat-sen University
[3] University of Shanghai for Science and Technology [4] Vector Institute
{zli3446,jcai336,gxu86,charles.ling}@uwo.ca zhengh239@mail.sysu.edu.cn
tangs@usst.edu.cn bwang@csd.uwo.ca

Abstract

Unlearnable examples (UEs) aim to prevent machine learning models from extracting useful information in protected training data. Extending UEs to segmentation is practically important but challenging, because pixel-level annotations are expensive to obtain and therefore cannot be assumed available when generating protective perturbations. This motivates us to study unsupervised unlearnable segmentation, where UE perturbations must be generated without segmentation annotations. Since dense prediction fundamentally relies on region-level affinity and cross-layer feature correspondence, effective segmentation data protection faces a twofold design challenge: perturbations should induce misleading alternatives to these structural priors, while the induced structure must remain easy to fit so that downstream segmentation models can adopt it during training. To address these challenges, we propose **DRIFT** (**D**isrupting **R**egion-level and **I**nterlayer **F**eature s**T**ructure), an unsupervised method that generates unlearnable examples for segmentation. Specifically, DRIFT constructs pseudo region-level affinity and pseudo cross-layer correspondence from clean-image features. It then optimizes perturbations to steer the extracted features of protected images toward the misleading pseudo-structure rather than the clean one, while an auxiliary structure learner, trained to predict pseudo-region assignments, explicitly encourages the induced structure to remain easy to fit. Extensive experiments across four segmentation datasets and multiple architectures demonstrate that DRIFT provides strong protection, consistently degrading the generalization performance of models trained on protected data and remaining competitive with the supervised UE baseline.

1 Introduction

As machine learning models become increasingly pervasive, preventing unauthorized use of data for model training has emerged as an important problem in trustworthy artificial intelligence. The abundance of publicly accessible online data has greatly fueled the success of modern learning systems [1, 2, 3], but it has also raised growing concerns that personal, proprietary, or otherwise sensitive data may be collected and used to train models without the data owners' consent. Unlearnable examples (UEs) have recently been proposed as a data protection mechanism against such unauthorized learning [4, 5, 6, 7, 8, 9, 10]. The key idea is to add imperceptible perturbations to training samples before release, such that models trained on the protected data fail to extract useful and generalizable knowledge, while the data remain visually unchanged to human observers.

*Corresponding authors

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

While existing UE methods primarily target classification [11, 12, 13, 14], extending data protection to segmentation is equally important yet faces distinctive practical and technical challenges. On the practical side, pixel-level segmentation annotations are costly to obtain, and data owners rarely annotate dense labels solely for protection, making realistic protection scenarios largely annotation-free. On the technical side, segmentation relies on fine-grained spatial and structural understanding [15, 16, 17, 18], whereas classification-oriented UE methods are not designed to explicitly disrupt such dense structural dependencies. Consequently, UE research specifically targeting segmentation remains limited. Among existing efforts, UnSeg [19] is a recent notable attempt, but it relies on pixel-level supervision and does not explicitly formulate protection as structural disruption. In practice, publicly released images, such as street-view, remote-sensing, and industrial inspection imagery, often lack segmentation annotations but may later be scraped, automatically annotated, or incorporated into dense-prediction training pipelines. This motivates annotation-free protection that does not require data owners to obtain costly pixel-level labels solely for perturbation generation. This leaves annotation-free segmentation protection underexplored.

These practical and technical considerations suggest that achieving segmentation data protection should not merely transplant classification-oriented objectives, nor depend entirely on segmentation supervision. Instead, without access to segmentation annotations, it must address two key requirements: (i) perturbations must induce misleading alternatives to the structural priors that segmentation models rely on; and (ii) the induced misleading structure must remain easy for downstream segmentation models to fit during training. Motivated by this perspective, we propose **DRIFT** (**D**isrupting **R**egion-level and **I**nterlayer **F**eature s**T**ructure), an unsupervised method to generate unlearnable examples for segmentation. As shown in Figure 1, the core insight of DRIFT is that segmentation data can be made unlearnable by replacing the structural priors it relies on with an easy-to-fit yet misleading pseudo-structure, without requiring any pixel-wise supervision. Specifically, DRIFT targets two such priors: region-level affinity among neighboring tokens and cross-layer correspondence for multi-scale fusion. It constructs pseudo-structures from clean-image features and trains a perturbation generator to steer the features of protected images toward these misleading ones. To ensure that the pseudo-structure is not only imposed as an external target but also learnable from the protected images, DRIFT introduces an auxiliary structure learner that fits the induced pseudo-region assignments and provides a binding objective for optimizing the perturbation generator. Together, DRIFT protects segmentation data without using segmentation annotations, while directly targeting region-level affinity and cross-layer correspondence that segmentation models fundamentally rely on.

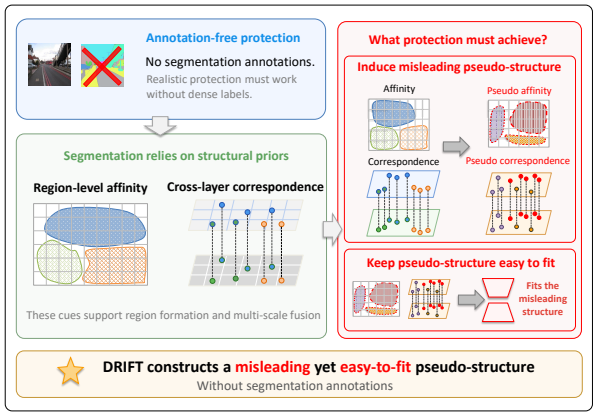


Figure 1: DRIFT protects segmentation data by replacing clean segmentation structure with misleading yet easy-to-fit pseudo-structure in an unsupervised manner.

Our main contributions are summarized as follows:

- We study *unsupervised unlearnable segmentation*, motivated by the observation that pixel-level annotations are costly to obtain in realistic protection scenarios. Unlike classification, protecting dense prediction should directly target segmentation-specific structural cues rather than image-level category evidence or segmentation supervision.
- We propose **DRIFT**, the first unsupervised UE method designed around segmentation-specific structural priors. DRIFT derives pseudo structural targets directly from clean-image features at two complementary levels, region-level affinity and cross-layer correspondence, requiring no segmentation annotations or task-specific supervisory signals during perturbation generation.
- Through extensive experiments, we show that DRIFT consistently degrades downstream segmentation performance without using segmentation annotations, while achieving perfor-

mance **competitive with the supervised segmentation UE method** in many settings. These results demonstrate the effectiveness of unsupervised structural protection for segmentation.

2 Related Work

Unlearnable examples (UEs) aim to protect training data by adding imperceptible perturbations that prevent unauthorized models from learning useful representations. The general principle is to make protected samples easy to fit during training but harmful for generalization, so that models trained on them perform poorly on clean test data. Existing UE methods have been primarily developed for classification, where perturbations are designed to induce spurious or misleading training signals rather than true semantic learning [4, 11, 6, 13, 12, 20, 21, 22]. These methods range from model-dependent approaches that rely on surrogate models or training-time optimization, to model-independent strategies leveraging random convolutions or class-wise linear separability [23, 24, 25].

Although effective in classification, these methods are mostly designed around image-level supervision and classification-oriented learning dynamics, making them less tailored to segmentation, where structural cues play a central role beyond category evidence. UnSeg [19] formulates segmentation UE over instance masks, but it does not explicitly target the structural dependencies that segmentation models rely on, and its perturbation generation depends on supervision. In practice, segmentation annotations are costly to obtain, and realistic data protection scenarios are often annotation-free. Therefore, unsupervised UE for segmentation, where perturbations are crafted without segmentation annotations, remains underexplored. Our work differs from these two lines along distinct axes: unlike classification-oriented UE methods, we target segmentation-specific structural dependencies, including region-level affinity and cross-layer correspondence; unlike annotation-dependent segmentation UE methods such as UnSeg, we generate perturbations without segmentation annotations.

3 Proposed Method

We propose DRIFT (Disrupting Region-level and Interlayer Feature sTructure), an unsupervised method for generating unlearnable examples for the segmentation task. DRIFT uses a frozen feature space to derive clean segmentation-related structural proxies, and then constructs an easy-to-fit, yet structurally misleading pseudo-structure. It optimizes the perturbation generator to induce protected images to exhibit the pseudo-structure, rather than the clean structural proxies, through pseudo-region-level affinity and pseudo-cross-layer correspondence.

3.1 Problem Setup

Let $\mathcal{D} = \{\boldsymbol{x}_i\}_{i=1}^N$ denote the clean training images. We assume that segmentation annotations are unavailable during perturbation generation. Our goal is to learn a perturbation generator $\mathcal{G}_\phi$ that produces a bounded perturbation $\boldsymbol{\delta}_i = \mathcal{G}_\phi(\boldsymbol{x}_i)$ for each image, with $\|\boldsymbol{\delta}_i\|_\infty \leq \epsilon$, and yields the protected sample $\boldsymbol{x}_i^u = \boldsymbol{x}_i + \boldsymbol{\delta}_i$.

The perturbation generator is trained without using segmentation annotations. Instead, we use a frozen ViT-style encoder F to extract token features from the clean and perturbed images. For the input image $\boldsymbol{x}_i$, let

$$T_i^{(\ell)} = F^{(\ell)}(\boldsymbol{x}_i) \in \mathbb{R}^{N_t \times d}, \qquad T_i^{u,(\ell)} = F^{(\ell)}(\boldsymbol{x}_i^u) \in \mathbb{R}^{N_t \times d}, \tag{1}$$

denote the features at layer ℓ, where N_t is the number of spatial tokens and d is the feature dimension.

We use $T_i = T_i^{(L)}$ and $T_i^u = T_i^{u,(L)}$ to denote the last-layer clean and perturbed token features, respectively. DRIFT uses the frozen clean features to construct a misleading pseudo-structure, and optimizes $\mathcal{G}_\phi$ such that the features of perturbed images become more compatible with this pseudo-structure than with the original clean structure.

3.2 Label-Free Pseudo-Structure Construction

Preserving meaningful relationships among local representations has been explored in structure-aware representation learning [26, 27, 28]. Motivated by the importance of structural relationships, we

construct misleading region-level affinities and cross-layer correspondences to disrupt segmentation learning. We construct a pseudo-structure directly from clean-image features, without using segmentation annotations. The goal is to derive a structurally misleading prior that can compete with the original segmentation-related structure while remaining easy to fit under small perturbation budgets. We use n, q, r to denote token indices and $\mathbf{p}_n \in \mathbb{Z}^2$ to denote the 2D token-grid coordinate of token n.

Clean structural primitives. Using the clean last-layer token features extracted by the frozen encoder, we first partition each image $\boldsymbol{x}_i$ into nonoverlapping M proto-regions,

$$\mathcal{U}_i = \{U_i^1, \ldots, U_i^M\}, \tag{2}$$

where each U_i^a is a set of spatial tokens that are coherent in the frozen feature space. These proto-regions serve as label-free proxies for segmentation-related regions.

For each token position n, let $u_i(n) \in \{1, \ldots, M\}$ denote the index of the proto-region containing token n. We define its real support set $\mathcal{P}_i(n)$ as the clean structural neighbors within the same proto-region:

$$\mathcal{P}_i(n) = \left\{ q \;\middle|\; q \in \mathrm{TopK}^{K_i(n)}_{r \in U_i^{u_i(n)},\, r \neq n} \cos\big(T_{i,n}, T_{i,r}\big) \right\}, \tag{3}$$

where

$$K_i(n) = \min\big(K,\, |U_i^{u_i(n)}| - 1\big). \tag{4}$$

Intuitively, $\mathcal{P}_i(n)$ captures the clean affinity evidence that supports region-level grouping around token n.

Pseudo affinity construction. A useful pseudo affinity structure should be locally consistent and easy to fit, yet sufficiently inconsistent with the original clean structure. Purely random neighborhood reassignment is typically too unstable to provide such a structural prior, while fixed reciprocal swapping between region pairs may degenerate into a reversible relabeling of the clean structure. To avoid these failure modes, we construct pseudo affinity through a directed, non-reciprocal partner mapping over proto-regions.

Specifically, for each proto-region U_i^a, we assign a partner region index $\psi_i(a)$ from its spatially adjacent regions:

$$\psi_i(a) \in \mathcal{N}_i(a), \qquad \psi_i(a) \neq a, \tag{5}$$

where $\mathcal{N}_i(a)$ denotes the set of indices of proto-regions spatially adjacent to U_i^a. Reciprocal pairing would reduce the construction to a symmetric exchange between two regions, which may resemble a reversible relabeling, whereas a directed non-reciprocal mapping produces a less symmetric and more misleading structural target. Partner selection is further constrained by size compatibility and region-level feature dissimilarity, so that the pseudo structure remains spatially plausible while being feature-inconsistent with the original grouping. We avoid reciprocal partner assignments whenever a valid alternative exists, as detailed in Appendix D.1.

For token n, we construct its pseudo support from the directed partner region $U_i^{\psi_i(u_i(n))}$:

$$\widetilde{\mathcal{P}}_i(n) = \left\{ q \;\middle|\; q \in \mathrm{TopK}^{\widetilde{K}_i(n)}_{r \in U_i^{\psi_i(u_i(n))}} \cos\big(T_{i,n}, T_{i,r}\big) \right\}, \tag{6}$$

where

$$\widetilde{K}_i(n) = \min\left(K, \left|U_i^{\psi_i(u_i(n))}\right|\right). \tag{7}$$

Thus, the pseudo support is entirely partner-derived in the main construction, ensuring that the pseudo affinity is driven by the directed partner mapping.

Pseudo cross-layer correspondence. The same directed partner mapping also induces a pseudo cross-layer correspondence for multi-scale feature fusion. For each directed pair $U_i^a \to U_i^{\psi_i(a)}$, we compute a 2D displacement direction $\mathbf{d}_i^a$ on the token grid. Here, the boundary band refers to the tokens in a proto-region that are spatially adjacent to its directed partner region, and $\mathbf{d}_i^a$ points from

the boundary band of U_i^a toward the corresponding boundary band of $U_i^{\psi_i(a)}$. The exact boundary-band construction is given in Appendix D.1. Let $\Omega \subset \mathbb{Z}^2$ denote the valid token-grid coordinates. For token n, we define the discrete one-step displacement as

$$\Delta_i(n) = \left\lfloor \mathbf{d}_i^{u_i(n)} \right\rceil, \qquad \Delta_i(n) \in \mathbb{Z}^2, \tag{8}$$

where $\lfloor \cdot \rceil$ denotes element-wise nearest-integer rounding. The pseudo correspondence $\pi_i(n)$ is then defined by

$$\mathbf{p}_{\pi_i(n)} = \Pi_\Omega \left(\mathbf{p}_n + \Delta_i(n) \right), \tag{9}$$

where $\Pi_\Omega(\cdot)$ projects a coordinate to the nearest valid position in Ω. This construction maps each token toward the boundary direction of its directed partner region, thereby replacing the original same-location correspondence with a misleading partner-induced correspondence. Together with the partner-derived pseudo support, this yields a unified pseudo-structure that perturbs both region-level affinity and cross-layer alignment.

3.3 Pseudo-Structure Replacement

We optimize the perturbation generator such that the perturbed features favor the pseudo-structure over the original clean structure, at both the token-affinity and cross-layer correspondence levels. Given the directed pseudo field constructed in Sec. 3.2, we assign each token a boundary-aware contribution weight $w_i(n) \in (0, 1]$, with larger values for tokens closer to directed inter-region boundaries. We use $\mathbb{E}_{n \sim w_i}[\cdot]$ to denote the corresponding boundary-weighted average over tokens, with details provided in Appendix D.1.

Pseudo affinity replacement. Segmentation relies on token-level affinity structure to organize local features into coherent regions. Therefore, if a perturbed token becomes more compatible with its pseudo support than with its original clean support, the region formation rule around that token is shifted from the clean segmentation prior to a misleading pseudo prior.

Let $T_{i,n}^u$ denote the perturbed last-layer token feature at position n, and define $\kappa_\tau(a, b) = \exp(\cos(a, b)/\tau)$. The boundary-weighted pseudo affinity replacement objective is

$$\mathcal{L}_{\text{aff}}^{\text{rep}} = \mathbb{E}_{n \sim w_i} \left[-\log \frac{\frac{1}{|\widetilde{\mathcal{P}}_i(n)|} \sum_{q \in \widetilde{\mathcal{P}}_i(n)} \kappa_\tau(T_{i,n}^u, T_{i,q}^u)}{\frac{1}{|\widetilde{\mathcal{P}}_i(n)|} \sum_{q \in \widetilde{\mathcal{P}}_i(n)} \kappa_\tau(T_{i,n}^u, T_{i,q}^u) + \frac{1}{|\mathcal{P}_i(n)|} \sum_{r \in \mathcal{P}_i(n)} \kappa_\tau(T_{i,n}^u, T_{i,r}^u)} \right], \tag{10}$$

where $\tau > 0$ is the temperature parameter. Minimizing this objective increases the relative average compatibility of token n with its pseudo support over its original clean support. Since both supports are selected under the same Top-K budget and size-compatible partner regions, the objective avoids favoring the pseudo support merely due to support-set cardinality. The weight $w_i(n)$ makes the replacement pressure stronger near directed region boundaries, where segmentation-related region formation is more sensitive.

Pseudo cross-layer correspondence replacement. For a set of selected layer pairs $\mathcal{L}$, we replace the original same-location alignment with the pseudo correspondence $\pi_i(n)$ constructed in Sec. 3.2. We define the boundary-weighted pseudo correspondence replacement objective as

$$\mathcal{L}_{\text{corr}}^{\text{rep}} = \frac{1}{|\mathcal{L}|} \sum_{(\ell_a, \ell_b) \in \mathcal{L}} \mathbb{E}_{n \sim w_i} \left[\cos\big(T_{i,n}^{u,(\ell_a)}, T_{i,n}^{u,(\ell_b)}\big) - \cos\big(T_{i,n}^{u,(\ell_a)}, T_{i,\pi_i(n)}^{u,(\ell_b)}\big) \right]. \tag{11}$$

Minimizing this objective suppresses the original same-position alignment and promotes a misleading yet consistent pseudo correspondence pattern across layers. The boundary-aware weighting further encourages this replacement to concentrate on tokens near inter-region transitions, where disrupting cross-layer alignment is most likely to affect dense prediction.

3.4 Overall Objective and Unsupervised Optimization

The replacement criteria in Sec. 3.3 specify how perturbed features should favor the pseudo-structure over the original clean structure. However, defining a pseudo-structure alone does not guarantee that it will be easy for a learner to adopt. To encourage the injected pseudo-structure to become

learnable under a small perturbation budget, we introduce a lightweight auxiliary structure learner h_θ and optimize DRIFT with a truncated alternating scheme.

We define the unified replacement objective as

$$\mathcal{L}_{\mathrm{rep}} = \mathcal{L}_{\mathrm{aff}}^{\mathrm{rep}} + \mathcal{L}_{\mathrm{corr}}^{\mathrm{rep}}. \tag{12}$$

Given a perturbed sample $\boldsymbol{x}_i^u$, the auxiliary learner h_θ predicts an M-way pseudo-region assignment from the perturbed last-layer token features. For each token n, we define its pseudo-region target as $\widetilde{y}_i(n) = \psi_i(u_i(n))$, which is induced by the directed partner mapping. The learner-side fitting loss is then defined as

$$\mathcal{L}_{\mathrm{fit}}(\theta; \boldsymbol{x}_i^u) = \frac{1}{N_t} \sum_{n=1}^{N_t} \mathrm{CE}\big(h_\theta(T_{i,n}^u), \widetilde{y}_i(n)\big). \tag{13}$$

This objective encourages the auxiliary learner to adopt the pseudo-region assignment implied by the directed partner structure.

In practice, we alternate between a small fixed number of inner updates on θ using $\mathcal{L}_{\mathrm{fit}}$ and one outer update of the perturbation generator. During the outer update, we treat θ as a constant with respect to the generator parameters ϕ, which gives a first-order approximation to the truncated bi-level optimization and ignores the implicit dependence of the inner-updated learner on ϕ. Nevertheless, gradients from $\mathcal{L}_{\mathrm{fit}}$ still back-propagate through the perturbed features T_i^u to update the generator, since T_i^u depends on ϕ through $\mathcal{G}_\phi$. The generator is optimized by

$$\min_{\phi} \ \mathbb{E}_{\boldsymbol{x}_i \sim \mathcal{D}} \Big[\mathcal{L}_{\mathrm{rep}} + \mathcal{L}_{\mathrm{fit}}(\theta; \boldsymbol{x}_i^u) \Big] \quad \text{s.t.} \quad \|\mathcal{G}_\phi(\boldsymbol{x}_i)\|_\infty \le \epsilon. \tag{14}$$

Thus, DRIFT is optimized not merely to distort features, but to promote a pseudo-structure that both replaces the original segmentation-related structure and remains easy for a learner to fit. Full optimization details and the instantiation of h_θ are given in Appendix D.1.

4 Experiments

In this section, we conduct extensive experiments to evaluate the effectiveness of DRIFT across four benchmark datasets. We first examine whether the protected images generated by DRIFT can consistently impair the learning of downstream segmentation models under supervised standard training and evaluation protocols. We then compare DRIFT with representative UE baselines on semantic, instance, and panoptic segmentation tasks to assess its protection performance across different datasets and architectures. Next, we evaluate the robustness of DRIFT against several representative defense strategies and the performance under mixing protected and clean training data. Finally, we perform ablation studies to analyze the contributions of the main components of DRIFT and to better understand the effect of different design choices.

4.1 Experimental Settings

Experimental Setup. We evaluate DRIFT under a segmentation-oriented unlearnable-example setting. Following the comparison protocol and reported settings in UnSeg [19], we report each baseline and metric when they are available under the same evaluation setup. We use four widely adopted benchmarks, including Pascal VOC 2012 [29], Cityscapes [30], ADE20K [31], and COCO [32]. For semantic segmentation, we evaluate on Pascal VOC 2012, Cityscapes, and ADE20K. For instance and panoptic segmentation, we evaluate on Cityscapes, ADE20K, and COCO.

For Pascal VOC 2012, we use DeepLabV3 [33] as a representative semantic segmentation model. For the more complex real-world datasets Cityscapes, ADE20K, and COCO, we adopt Mask2Former [18] and follow the same training procedure in its paper, as done in UnSeg. All downstream models are trained on the protected training sets and evaluated on the corresponding clean validation or test sets.

Evaluation Metrics and Protocol. We follow standard metrics for each task. For semantic segmentation, we report mIoU. For instance segmentation, we report AP. For panoptic segmentation, we report PQ. Lower downstream performance indicates stronger unlearnability and better protection. Following UnSeg, COCO is evaluated for panoptic and instance segmentation only, while for Cityscapes we report the overall AP rather than the size-specific AP breakdown.

Table 1: Main results on ADE20K and COCO using Mask2Former. Lower values indicate stronger unlearnability. Methods are grouped by whether segmentation annotations are used during perturbation generation. The best results among methods without segmentation annotations are **boldfaced**, while the overall best results are underlined. R50: ResNet50, Swin-T: Swin Transformer-Tiny.

Dataset	Method	Backbone	Panoptic			Instance				Semantic
			PQ	AP^{Th}_{pan}	$mIoU_{pan}$	AP	AP^S	AP^M	AP^L	mIoU
ADE20K	Clean	R50	39.7	26.5	46.1	26.4	10.4	28.9	43.1	47.2
		Swin-T	41.6	27.7	49.3	27.9	10.8	29.8	46.2	47.7
	Without segmentation annotations									
	SynPer [23]	R50	18.6	13.6	28.7	**9.3**	7.1	13.4	**9.7**	25.4
	AR [25]	R50	37.8	24.9	43.1	25.4	9.4	27.7	43.3	43.9
	CUDA [24]	R50	10.7	**8.4**	19.6	12.0	3.9	14.6	22.5	19.6
	DRIFT	R50	**9.2**	9.3	**17.8**	9.4	**2.4**	**10.2**	17.2	**18.2**
		Swin-T	**5.9**	**6.0**	**15.9**	**7.6**	**1.8**	**8.1**	**13.4**	**11.5**
	With segmentation annotations									
	UnSeg [19]	R50	11.7	7.5	17.7	6.2	5.0	8.6	7.3	16.7
		Swin-T	4.1	3.4	10.6	4.1	4.0	5.8	3.4	7.8
COCO	Clean	R50	51.9	41.7	61.7	43.7	23.4	47.2	64.8	-
		Swin-T	53.2	43.3	63.2	45.0	24.5	48.3	67.4	-
	Without segmentation annotations									
	SynPer [23]	R50	11.3	9.5	11.0	10.8	13.4	15.2	**5.0**	-
	CUDA [24]	R50	6.7	4.7	11.2	9.7	3.7	10.9	18.8	-
	DRIFT	R50	**4.1**	**3.3**	**7.8**	**4.3**	**1.6**	**2.6**	5.9	-
		Swin-T	**4.8**	**6.0**	**6.5**	**2.3**	**0.4**	**2.1**	**4.1**	-
	With segmentation annotations									
	UnSeg [19]	R50	4.2	3.2	5.2	4.0	5.8	3.7	1.7	-
		Swin-T	4.1	2.8	6.0	2.7	4.4	1.9	0.7	-

Table 2: Main results on Cityscapes using Mask2Former. The best results among methods without segmentation annotations are **boldfaced**, while the overall best results are underlined.

Dataset	Method	Backbone	Panoptic			Instance	Semantic
			PQ	AP^{Th}_{pan}	$mIoU_{pan}$	AP	mIoU
Cityscapes	Clean	R50	62.1	37.3	77.5	37.4	79.4
		Swin-T	63.9	39.1	80.5	39.7	82.1
	Without segmentation annotations						
	SynPer [23]	R50	30.1	23.0	37.1	20.5	25.5
	AR [25]	R50	51.6	36.0	68.3	35.5	68.9
	CUDA [24]	R50	51.6	31.4	69.1	29.9	65.8
	DRIFT	R50	**4.8**	**2.1**	**10.6**	**1.8**	**12.2**
		Swin-T	**8.7**	**2.6**	**17.4**	**3.1**	**14.5**
	With segmentation annotations						
	UnSeg [19]	R50	5.7	1.1	7.8	2.3	10.9
		Swin-T	7.2	1.7	12.6	1.5	17.8

Baselines. We compare DRIFT with representative perturbation-based UE baselines adapted to the same segmentation evaluation setting, including Synthetic Perturbations (SynPer) [23], Autoregressive Perturbations (AR) [25], Convolution-based Perturbations (CUDA) [24], and Unlearnable Segmentation (UnSeg) [19]. All methods are compared under the same perturbation budget and downstream training protocol for a fair comparison. More details are shown in Appendix D.4.

Implementation Details. Our method is implemented in PyTorch and trained on a single NVIDIA L40S GPU unless otherwise specified. We use the Adam optimizer [34] with an initial learning rate of 0.001. To ensure imperceptibility, we set the perturbation bound ϵ to $8/255$. Results are averaged over five runs with different random seeds. More implementation details are shown in Appendix D.1.

4.2 Main Results

Effectiveness on Mainstream Image Segmentation Tasks. We evaluate DRIFT on three widely-used segmentation benchmarks, ADE20K, COCO, and Cityscapes, using Mask2Former with two strong backbones, ResNet50 [35] and Swin-Tiny [36]. Together, these benchmarks cover panoptic, instance, and semantic segmentation, allowing us to assess whether DRIFT transfers across different dense prediction settings. We compare DRIFT with representative perturbation-based UE baselines, including SynPer, AR, CUDA, and the recent segmentation-specific method UnSeg.

As shown in Tables 1 and 2, DRIFT consistently causes severe degradation of downstream segmentation performance across datasets and evaluation protocols. For example, on ADE20K, DRIFT reduces the PQ / AP / mIoU of the ResNet50-based Mask2Former from 39.7 / 26.4 / 47.2 to 9.2 / 9.4 / 18.2, and degrades the Swin-Tiny-based model to 5.9 / 7.6 / 11.5. Similarly strong degradation is observed on COCO and Cityscapes, where DRIFT also drives panoptic, instance, and semantic metrics to very low levels. These results indicate that DRIFT performs effectively across diverse dense prediction settings rather than exploiting dataset-specific or architecture-specific vulnerabilities.

Among methods that do not use segmentation annotations during perturbation generation, DRIFT consistently delivers the strongest or one of the strongest protection results across datasets. On ADE20K with the ResNet50 backbone, for instance, DRIFT reduces PQ to 9.2, clearly below SynPer (18.6), AR (37.8), and CUDA (10.7). A similar trend is observed on COCO and Cityscapes, where DRIFT remains highly competitive across panoptic, instance, and semantic metrics. This is consistent with the design of DRIFT: rather than merely perturbing pixels, it explicitly replaces the original segmentation-related structure with a pseudo-structure, jointly shifting both within-layer region affinity and cross-layer correspondence away from the clean segmentation prior.

Compared with UnSeg, which uses pixel-level segmentation annotations during perturbation generation, DRIFT remains highly competitive despite operating in a substantially weaker label-free setting. Notably, DRIFT matches or even surpasses UnSeg on several metrics, especially those related to small or fine-grained objects. For example, on ADE20K, DRIFT achieves lower AP^S than UnSeg under both backbones (2.4 vs. 5.0 on R50, and 1.8 vs. 4.0 on Swin-T). At the same time, DRIFT is relatively weaker on some large-object or large-region metrics. This trend is also consistent with our design: the boundary-aware pseudo-structure replacement is particularly effective at shifting local affinity and correspondence, but for very large regions, the induced pseudo structure may still partially overlap with the original category near the boundary rather than being fully displaced away from the true semantic extent. As a result, DRIFT tends to be more effective on small or fine-grained structures, while UnSeg can retain an advantage on some panoptic or large-object metrics. Even so, the overall results show that DRIFT provides competitive protection in segmentation tasks without using pixel-level annotations during the training of the perturbation generator.

Generalization Across DeepLab Variants. Beyond the Mask2Former-based experiments above, we further examine whether DRIFT generalizes within a single architecture family by evaluating it on Pascal VOC 2012 using a representative family of DeepLab models, including DeepLabV1 [37], DeepLabV2 [17], DeepLabV3 [33], and DeepLabV3+ [38]. These variants differ in their architectural designs, such as atrous convolution, multi-scale context aggregation, and encoder-decoder refinement, providing a meaningful testbed for assessing cross-model generalization.

As shown in Table 3, DRIFT consistently achieves strong protection across all four DeepLab variants. Compared with the classification-oriented SynPer baseline, DRIFT leads to substantially stronger degradation of downstream segmentation performance, indicating that its structural perturbations are more suitable for dense prediction tasks. More importantly, although DRIFT optimizes the perturbation generator without using segmentation labels, it remains highly competitive with the supervised segmentation-specific method UnSeg and even achieves better protection on the more recent DeepLabV3 and DeepLabV3+ models. Note that although DRIFT constructs the pseudo-structure from a ViT-based frozen encoder, the resulting perturbations transfer effectively to CNN-based DeepLab architectures. This suggests that the disrupted structural cues are shared across architectural families rather than tied to a specific feature extractor.

4.3 Additional Experiment

Resistance to Potential Defenses. We evaluate the robustness of DRIFT and UnSeg perturbations under several representative defense strategies on Pascal VOC 2012 using DeepLabV3 as the down-

Table 3: Cross-architecture transfer results on semantic segmentation. Lower mIoU indicates stronger unlearnability. The best protection results are **boldfaced**.

Dataset	Method	DeepLabV1	DeepLabV2	DeepLabV3	DeepLabV3+
Pascal VOC 2012	Clean	70.6	76.4	82.6	89.0
	SynPer [23]	19.2	15.4	12.6	12.8
	UnSeg [19]	**6.2**	**4.5**	5.8	5.2
	DRIFT	8.5	5.2	**5.6**	**3.0**

Table 4: Resistance to potential defenses on Pascal VOC 2012 using DeepLabV3. (Clean-training mIoU = 82.6 in Table 3)

Method	None	Gauss.	JPEG	AT	DDC-AT
UnSeg	5.8	**7.3**	44.8	**23.1**	28.5
DRIFT	**5.6**	8.9	**26.4**	24.8	**26.2**

Table 5: Mixing clean and protected training data on Pascal VOC 2012 using DeepLabV3.

Training Data	0%	20%	40%	60%	80%
Clean Only	–	71.5	74.4	74.9	76.1
UnSeg Mixed	5.8	72.4	74.5	**75.2**	**76.1**
DRIFT Mixed	**5.6**	**71.9**	**74.2**	75.5	76.2

stream segmentation model. The evaluated defenses include input transformations, i.e., Gaussian noise and JPEG compression [39], and adversarial-training-based defenses, i.e., AT [40] and DDC-AT [41]. For a fair comparison, we follow the defense configurations used in UnSeg [19] for all defense evaluations. As shown in Table 4, the two methods exhibit different robustness profiles. Under JPEG compression, UnSeg degrades sharply, with mIoU increasing from 5.8 to 44.8, whereas DRIFT maintains substantially stronger protection, increasing from 5.6 to 26.4. This large gap suggests that the structural perturbations induced by DRIFT are less vulnerable to lossy compression, which is one of the most practical and easily deployed input purification techniques. Under Gaussian noise, both methods remain largely effective, with mIoU values of 7.3 for UnSeg and 8.9 for DRIFT, suggesting that neither method is easily neutralized by stochastic pixel-level corruption. Under adversarial-training-based defenses, both methods experience partial recovery, but still substantially degrade downstream segmentation performance. UnSeg is slightly stronger under AT, while the two methods show comparable robustness under DDC-AT. These results suggest that adversarial training alone may be insufficient to recover the region-level affinity and cross-layer correspondence disrupted by DRIFT. Unlike conventional input perturbations, DRIFT explicitly targets structural dependencies during training, which may explain why its protection effect persists even after applying these defenses. Overall, these results indicate that DRIFT remains robust to common defenses, achieving defense robustness comparable to the annotation-dependent UnSeg while requiring no segmentation annotations, and showing a clear advantage under JPEG compression.

Mixing Protected and Clean Training Data. In practical data-protection scenarios, an unauthorized training set may not consist entirely of protected images. Instead, protected samples released by a data owner can be mixed with clean samples collected from other sources. Therefore, it is important to evaluate whether the protected samples still provide limited useful supervision when only part of the training data is unlearnable. Following this motivation, we construct mixed training sets on Pascal VOC 2012 with DeepLabV3, where a given proportion of samples remains clean and the remaining samples are replaced by their protected versions. We also train models on the corresponding clean-only subsets as references, so that the gap between mixed-data training and clean-only training reflects how much useful segmentation information can still be extracted from the protected samples.

As shown in Table 5, DRIFT remains effective in this mixed-data regime. When the training set contains only protected samples, DRIFT leads to near-failure segmentation performance, confirming its strong protection in the fully protected setting. As the clean proportion increases, the performance of models trained on mixed data gradually approaches that of models trained on the corresponding clean-only subsets. This trend indicates that the downstream performance is mainly determined by the available clean samples, while the protected samples generated by DRIFT contribute little additional useful supervision for recovering the correct semantic layout. Notably, despite not using pixel-level segmentation annotations during perturbation generation, DRIFT exhibits a mixed-training trend broadly similar to the supervised segmentation-specific method UnSeg. These results further demonstrate that DRIFT can induce strong segmentation unlearnability in a label-free manner, even when protected data are mixed with clean training samples.

Table 6: SAM 3 evaluation on Pascal VOC 2012. The frozen-inference comparison uses clean versus DRIFT-protected validation inputs. Both encoder-fine-tuning variants are evaluated on clean validation images. Lower mIoU for protected conditions indicates stronger impact.

Setting	Training data	Evaluation input	mIoU ↑
Frozen inference	None	Clean validation	49.6
Frozen inference	None	Protected validation	45.4
Encoder fine-tuning	Clean training	Clean validation	77.3
Encoder fine-tuning	Protected training	Clean validation	56.2

Ablation of DRIFT components. We ablate three key components of DRIFT: pseudo affinity replacement, pseudo cross-layer correspondence replacement, and auxiliary structure fitting. As shown in Figure 2, Full DRIFT reduces the overall mIoU from 82.6 to 5.6, corresponding to a 77.0-point drop from clean training. Removing any component leads to clear performance recovery, confirming that all three components contribute to the final unlearnability strength. In particular, removing pseudo affinity replacement weakens DRIFT most substantially, increasing the mIoU to 42.3 and reducing the drop from clean to 40.3. This indicates that shifting token-level region affinity is the primary driver of DRIFT. Removing auxiliary structure fitting also noticeably weakens the performance, yielding 21.9 mIoU, which suggests that merely defining pseudo-structure is insufficient unless it is further bound to the learner as a misleading structure. By contrast, removing cross-layer correspondence replacement results in a smaller but still clear recovery, with 11.2 mIoU, showing that it provides complementary gains by further disrupting multi-scale feature alignment. More results on several classes results are provided in Figure 3 in the Appendix E.1.

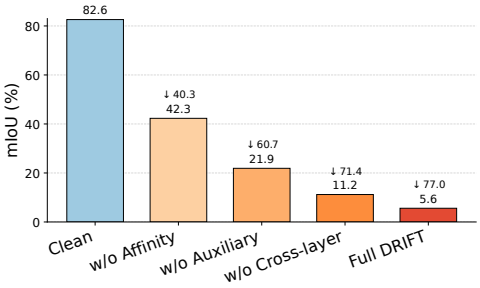


Figure 2: Ablation of DRIFT components on Pascal VOC 2012 semantic segmentation.

Evaluation on Segmentation Foundation Models. To evaluate our method's performance on foundation models, we evaluate DRIFT with SAM3 [42] on Pascal VOC in two settings: (a) Frozen inference: the same pretrained checkpoint and prompts are applied to both clean and protected validation images; (b) Encoder fine-tuning: only the encoder is fine-tuned on either clean or protected training images, while all other modules remain fixed. Both variants are evaluated on clean validation images. As shown in Table 6, DRIFT causes a 4.2-point drop in frozen inference, but a much larger 21.1-point drop after fine-tuning. The impact of DRIFT on frozen inference is relatively limited; however, fine-tuning a model on protected data leads to a significant drop in performance on clean validation images. This supports the conclusion that DRIFT achieves its protective effect primarily by disrupting structural learning during the training process, rather than simply making the images difficult to segment during inference.

5 Conclusion and Limitation

In this paper, we studied unsupervised unlearnable segmentation, motivated by both the high cost of obtaining pixel-level annotations in realistic protection scenarios and the limitation of existing classification-oriented UE methods in disrupting segmentation-specific structural dependencies. To address this problem, we proposed **DRIFT**, an unsupervised method that constructs pseudo region-level affinity and pseudo cross-layer correspondence from clean-image features, and uses an auxiliary structure learner to encourage the induced misleading pseudo-structure to be easy for downstream learners to fit. Extensive experiments across four datasets and multiple architectures show that DRIFT consistently degrades downstream segmentation performance while remaining competitive with the supervised UE segmentation baseline. DRIFT still has limitations: pseudo-structure construction and perturbation generator optimization introduce additional computational cost, and on large-area categories, the pseudo-structure is less fully redirected away from the original object. Future work may explore more efficient optimization and stronger protection for unlearnable segmentation.

Acknowledgement

This work is supported by the Vector Institute and the Natural Sciences and Engineering Research Council of Canada (NSERC) Discovery Grants program.

References

[1] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[2] Tom B Brown. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

[3] Weiming Liu, Xiaolin Zheng, Jiajie Su, Longfei Zheng, Chaochao Chen, and Mengling Hu. Contrastive proxy kernel stein path alignment for cross-domain cold-start recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 35(11):11216–11230, 2023.

[4] Hanxun Huang, Xingjun Ma, Sarah Monazam Erfani, James Bailey, and Yisen Wang. Unlearnable examples: Making personal data unexploitable. In *International Conference on Learning Representations*, 2021.

[5] Shaopeng Fu, Fengxiang He, Yang Liu, Li Shen, and Dacheng Tao. Robust unlearnable examples: Protecting data against adversarial learning. In *International Conference on Learning Representations*, 2022.

[6] Jie Ren, Han Xu, Yuxuan Wan, Xingjun Ma, Lichao Sun, and Jiliang Tang. Transferable unlearnable examples. *arXiv preprint arXiv:2210.10114*, 2022.

[7] Minghui Li, Xianlong Wang, Zhifei Yu, Shengshan Hu, Ziqi Zhou, Longling Zhang, and Leo Yu Zhang. Detecting and corrupting convolution-based unlearnable examples. In *Proceedings of the 39th AAAI Conference on Artificial Intelligence (AAAI'25)*, 2025.

[8] Xianlong Wang, Hangtao Zhang, Wenbo Pan, Ziqi Zhou, Changsong Jiang, Li Zeng, and Xiaohua Jia. Dual-branch robust unlearnable examples. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2026.

[9] Xianlong Wang, Minghui Li, Wei Liu, Hangtao Zhang, Shengshan Hu, Yechao Zhang, Ziqi Zhou, and Hai Jin. Unlearnable 3d point clouds: Class-wise transformation is all you need. In *Proceedings of the 38th Annual Conference on Neural Information Processing Systems (NeurIPS'24)*, 2024.

[10] Zhihao Li, Gezheng Xu, Jiale Cai, Ruiyi Fang, Di Wu, Qicheng Lao, Charles Ling, and Boyu Wang. When priors backfire: On the vulnerability of unlearnable examples to pretraining. In *The Fourteenth International Conference on Learning Representations*, 2026.

[11] Shuang Liu, Yihan Wang, and Xiao-Shan Gao. Game-theoretic unlearnable example generator. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 21349–21358, 2024.

[12] Chaochao Chen, Jiaming Zhang, Yuyuan Li, and Zhongxuan Han. One for all: A universal generator for concept unlearnability via multi-modal alignment. In *International Conference on Machine Learning*, pages 7700 – 7711, 2024.

[13] Zhihao Li, Jiale Cai, Gezheng Xu, Hao Zheng, Qiuyue Li, Fan Zhou, Shichun Yang, Charles Ling, and Boyu Wang. Versatile transferable unlearnable example generator. In *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pages 17495–17522. Curran Associates, Inc., 2025.

[14] Derui Wang, Minhui Xue, Bo Li, Seyit Camtepe, and Liming Zhu. Provably unlearnable data examples. *arXiv preprint arXiv:2405.03316*, 2024.

[15] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[16] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[17] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017.

[18] Bowen Cheng, Ishan Misra, Alexander G Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention mask transformer for universal image segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1290–1299, 2022.

[19] Ye Sun, Hao Zhang, Tiehua Zhang, Xingjun Ma, and Yu-Gang Jiang. Unseg: One universal unlearnable example generator is enough against all image segmentation. *arXiv preprint arXiv:2410.09909*, 2024.

[20] Jiaming Zhang, Xingjun Ma, Qi Yi, Jitao Sang, Yu-Gang Jiang, Yaowei Wang, and Changsheng Xu. Unlearnable clusters: Towards label-agnostic unlearnable examples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3984–3993, 2023.

[21] Xinwei Liu, Xiaojun Jia, Yuan Xun, Siyuan Liang, and Xiaochun Cao. Multimodal unlearnable examples: Protecting data against multimodal contrastive learning. *arXiv preprint arXiv:2407.16307*, 2024.

[22] Jiale Cai, Gezheng Xu, Zhihao Li, Ruiyi Fang, Ruizhi Pu, Di Wu, Qicheng Lao, Charles Ling, and Boyu Wang. FUSE: Full-spectrum unlearnable examples via spectral equalization. In *Forty-third International Conference on Machine Learning*, 2026.

[23] Da Yu, Huishuai Zhang, Wei Chen, Jian Yin, and Tie-Yan Liu. Availability attacks create shortcuts. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 2367–2376, 2022.

[24] Vinu Sankar Sadasivan, Mahdi Soltanolkotabi, and Soheil Feizi. Cuda: Convolution-based unlearnable datasets. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3862–3871, 2023.

[25] Pedro Sandoval-Segura, Vasu Singla, Jonas Geiping, Micah Goldblum, Tom Goldstein, and David Jacobs. Autoregressive perturbations for data poisoning. *Advances in Neural Information Processing Systems*, 35:27374–27386, 2022.

[26] Ruiyi Fang, Liangjian Wen, Zhao Kang, and Jianzhuang Liu. Structure-preserving graph representation learning. In *2022 IEEE International Conference on Data Mining (ICDM)*, pages 927–932. IEEE, 2022.

[27] Ruiyi Fang, Bingheng Li, Jingyu Zhao, Ruizhi Pu, Qiuhao Zeng, Gezheng Xu, Charles Ling, and Boyu Wang. Homophily enhanced graph domain adaptation. *Forty-Second International Conference on Machine Learning*, 2025.

[28] Ruiyi Fang, Shuo Wang, Ruizhi Pu, Qiuhao Zeng, Hao Zheng, Ziyan Wang, Jiale Cai, Zhimin Mei, Song Tang, Charles Ling, et al. Graph domain adaptation via homophily-agnostic reconstructing structure. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 21047–21055, 2026.

[29] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88:303–338, 2010.

[30] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016.

[31] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 633–641, 2017.

[32] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[33] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017.

[34] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

[35] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[36] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021.

[37] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. *arXiv preprint arXiv:1412.7062*, 2014.

[38] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 801–818, 2018.

[39] Zhuoran Liu, Zhengyu Zhao, and Martha Larson. Image shortcut squeezing: Countering perturbative availability poisons with compression. In *International conference on machine learning*, pages 22473–22487. PMLR, 2023.

[40] Aleksander Madry. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018.

[41] Xiaogang Xu, Hengshuang Zhao, and Jiaya Jia. Dynamic divide-and-conquer adversarial training for robust semantic segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7486–7495, 2021.

[42] Nicolas Carion, Laura Gustafson, Yuan-Ting Hu, Shoubhik Debnath, Ronghang Hu, Didac Suris Coll-Vinent, Chaitanya Ryali, Kalyan Vasudev Alwala, Haitham Khedr, Andrew Huang, et al. Sam 3: Segment anything with concepts. In *International conference on learning representations*, volume 2026, pages 138846–138923, 2026.

[43] Changjian Shui, Ruizhi Pu, Gezheng Xu, Jun Wen, Fan Zhou, Christian Gagné, Charles X Ling, and Boyu Wang. Towards more general loss and setting in unsupervised domain adaptation. *IEEE Transactions on Knowledge and Data Engineering*, 35(10):10140–10150, 2023.

[44] Qiuhao Zeng, Wei Wang, Fan Zhou, Charles Ling, and Boyu Wang. Foresee what you will learn: data augmentation for domain generalization in non-stationary environment. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pages 11147–11155, 2023.

[45] Qiuhao Zeng, Changjian Shui, Long-Kai Huang, Peng Liu, Xi Chen, Charles Ling, and Boyu Wang. Latent trajectory learning for limited timestamps under distribution shift over time. In *The Twelfth International Conference on Learning Representations*, 2024.

[46] Qiuhao Zeng, Wei Wang, Fan Zhou, Gezheng Xu, Ruizhi Pu, Changjian Shui, Christian Gagné, Shichun Yang, Charles X Ling, and Boyu Wang. Generalizing across temporal domains with koopman operators. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 16651–16659, 2024.

[47] Li Yi, Gezheng Xu, Pengcheng Xu, Jiaqi Li, Ruizhi Pu, Charles Ling, A Ian McLeod, and Boyu Wang. When source-free domain adaptation meets learning with noisy labels. *arXiv preprint arXiv:2301.13381*, 2023.

[48] Gezheng Xu, Li Yi, Pengcheng Xu, Jiaqi Li, Ruizhi Pu, Changjian Shui, A Ian McLeod, Boyu Wang, and Charles Ling. Unraveling the mysteries of label noise in source-free domain adaptation: Theory and practice. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(5):4007–4023, 2025.

[49] Gezheng Xu, Hui Guo, Li Yi, Charles Ling, Boyu Wang, and Grace Yi. Revisiting source-free domain adaptation: a new perspective via uncertainty control. In *International Conference on Learning Representations*, volume 2025, pages 92900–92939, 2025.

[50] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

Appendix

This appendix provides a comprehensive analysis and in-depth exploration of our proposed DRIFT (Disrupting Region-level and Interlayer Feature sTructure).

A Potential Social Impact

Our work introduces DRIFT, an unsupervised unlearnable example framework for protecting segmentation data from unauthorized model training. DRIFT is designed to support privacy-preserving data protection, especially in scenarios where pixel-level annotations are unavailable or too costly to obtain solely for perturbation generation. By reducing the ability of downstream models to learn useful segmentation representations from protected data, DRIFT may help data owners better control the use of their released visual content. At the same time, unlearnable perturbations could potentially be misused to interfere with legitimate model training or benchmark construction. We therefore emphasize that DRIFT should only be used for ethical data-protection purposes and in compliance with applicable laws, dataset licenses, and community standards. All experiments are conducted on publicly available segmentation benchmarks, including PASCAL VOC 2012, ADE20K, COCO, and Cityscapes, for research purposes only.

B Notations

A summary of the main notation used throughout DRIFT is provided in Table 7. Throughout the paper, all structural proxies and pseudo-region targets are constructed without segmentation labels.

Table 7: Main notation used in DRIFT.

Symbol	Description
$\mathcal{D} = \{x_i\}_{i=1}^{N}$	Unlabeled images used to train the perturbation generator.
G_ϕ, δ_i, x_i^u	Perturbation generator, its bounded perturbation, and protected image $x_i + \delta_i$.
$F^{(\ell)}$, $T_i^{(\ell)}$, $T_i^{u,(\ell)}$	Frozen encoder at layer ℓ and its clean/protected token features.
N_t, d, p_n	Number of spatial tokens, token feature dimension, and 2D coordinate of token n.
$\mathcal{U}_i = \{U_i^a\}_{a=1}^{M}$	Nonoverlapping proto-region partition obtained from clean features.
$u_i(n)$, M	Proto-region index of token n and number of proto-regions.
$P_i(n)$, $\widetilde{P}_i(n)$	Original within-region support and partner-derived pseudo support of token n.
K, $K_i(n)$, $\widetilde{K}_i(n)$	Support budget and effective original/pseudo support sizes.
$\mathcal{N}_i(a)$, $\psi_i(a)$	Adjacent proto-region indices and directed partner index for region a.
$B_i^{a\to b}$, $s_i(a,b)$	Boundary band facing region b and partner-selection score.
d_i^a, $\Delta_i(n)$, $\pi_i(n)$	Directed displacement, its rounded token-grid step, and shifted correspondence index.
$w_i(n)$, σ	Boundary-aware token weight and its distance-decay scale.
$\mathcal{L}$, τ	Selected cross-layer pairs and affinity temperature.
h_θ, $\widetilde{y}_i(n)$	Auxiliary learner and pseudo-region label $\psi_i(u_i(n))$.
$\mathcal{L}_{\text{aff}}^{\text{rep}}$, $\mathcal{L}_{\text{corr}}^{\text{rep}}$	Pseudo affinity and cross-layer correspondence replacement losses.
$\mathcal{L}_{\text{fit}}$, $\mathcal{L}_{\text{rep}}$	Auxiliary structure-fitting loss and summed replacement loss.
S, ϵ	Number of auxiliary inner updates and ℓ_∞ perturbation budget.

C Future Directions: Unlearnable Segmentation under Distribution Shifts

An interesting direction for future research is to investigate the effectiveness of unlearnable examples under distribution shifts and domain adaptation. Existing studies have explored how models generalize across different domains and evolving data distributions [43, 44, 45, 46]. In addition, recent source-free domain adaptation methods have investigated the influence of noisy supervision

and representation uncertainty on model adaptation [47, 48, 49]. These studies suggest potential learning strategies that an unauthorized user might employ when training on protected data. While DRIFT demonstrates cross-dataset transferability of its perturbation generator, its robustness against downstream domain adaptation and generalization strategies remains unexplored. Investigating whether such strategies can recover useful segmentation representations from protected data would be a valuable direction for future work.

D Experimental Details

D.1 More Implementation Details of DRIFT

Training details. We build DRIFT on a frozen DINOv2-base (ViT-B/14) feature extractor [50]. Inputs are resized to 320×320, yielding a 23×23 token grid, i.e., $N_t = 529$ spatial tokens per image. For each image, we cluster the last-layer tokens into $M = 20$ proto-regions by cosine k-means for 8 iterations. The support size is set to $K = 8$. $\mathcal{L}_{\text{aff}}^{\text{rep}}$ uses temperature $\tau = 0.1$. The selected cross-layer pairs are $\{(3, 11), (6, 11)\}$. The auxiliary structure learner h_θ is instantiated as a 2-layer Transformer encoder with a learnable positional embedding and a final linear classifier to M logits. We perform 3 truncated inner updates per outer update, optimize θ with Adam (lr $= 10^{-3}$), and optimize the perturbation generator ϕ with Adam (lr $= 10^{-2}$). We train DRIFT for 60 epochs on PASCAL VOC 2012, 50 epochs on Cityscapes, and 30 epochs on ADE20K and COCO, using a batch size of 16 for all datasets. Training DRIFT on Pascal VOC 2012 takes approximately 1.5 hours on a single NVIDIA L40S GPU.

Partner selection. We use n, q, r to denote token indices and $\mathbf{p}_n \in \mathbb{Z}^2$ to denote the 2D token-grid coordinate of token n. In practice, the adjacency set $\mathcal{N}_i(a)$ defined in Sec. 3.2 is constructed on the token grid. Two proto-regions U_i^a and U_i^b are considered adjacent if there exist $n \in U_i^a$ and $q \in U_i^b$ such that

$$\|\mathbf{p}_n - \mathbf{p}_q\|_1 = 1. \tag{15}$$

For each adjacent pair, we define the directed boundary band

$$B_i^{a\to b} = \left\{ n \in U_i^a \;\middle|\; \exists q \in U_i^b,\ \|\mathbf{p}_n - \mathbf{p}_q\|_1 = 1 \right\}. \tag{16}$$

Thus, $B_i^{a\to b}$ is a set of token indices on the boundary of U_i^a facing U_i^b.

We select the partner region index $\psi_i(a)$ from adjacent region indices using a size-aware and feature-aware score. For each adjacent candidate $b \in \mathcal{N}_i(a)$, we compute

$$s_i(a, b) = \frac{|B_i^{a\to b}|}{\sqrt{|U_i^a||U_i^b|}} + \left(1 - \cos(\bar{T}_i^a, \bar{T}_i^b)\right), \tag{17}$$

where

$$\bar{T}_i^a = \frac{1}{|U_i^a|} \sum_{n\in U_i^a} T_{i,n}. \tag{18}$$

We choose the highest-scoring adjacent candidate while avoiding reciprocal pairs whenever an alternative candidate exists. This yields a directed, non-reciprocal partner mapping.

Boundary-aware token weighting. After partner selection, we write

$$B_i^a := B_i^{a\to\psi_i(a)} \tag{19}$$

for the directed boundary band of region U_i^a toward its selected partner. For token $n \in U_i^a$, the boundary-aware contribution weight is

$$w_i(n) = \exp\left(-\frac{d(\mathbf{p}_n, B_i^a)^2}{2\sigma^2}\right), \qquad d(\mathbf{p}_n, B_i^a) = \min_{q\in B_i^a} \|\mathbf{p}_n - \mathbf{p}_q\|_2. \tag{20}$$

The boundary-weighted token average used in Sec. 3.3 is implemented as

$$\mathbb{E}_{n\sim w_i}[f(n)] := \frac{\sum_{n=1}^{N_t} w_i(n) f(n)}{\sum_{n=1}^{N_t} w_i(n)}. \tag{21}$$

The weight $w_i(n)$ is used only to weight token-wise replacement losses, not to rank pseudo-support candidates or construct pseudo correspondences.

Displacement direction. For pseudo cross-layer correspondence, we compute the centroid of a directed boundary band as

$$\mathbf{c}_i^{a\to b} = \frac{1}{|B_i^{a\to b}|} \sum_{q\in B_i^{a\to b}} \mathbf{p}_q. \tag{22}$$

The displacement direction from U_i^a to its partner is

$$\mathbf{d}_i^a = \frac{\mathbf{c}_i^{\psi_i(a)\to a} - \mathbf{c}_i^{a\to\psi_i(a)}}{\left\|\mathbf{c}_i^{\psi_i(a)\to a} - \mathbf{c}_i^{a\to\psi_i(a)}\right\|_2}. \tag{23}$$

The architecture of perturbation generator. In the main paper, we devise a generator to produce transferable perturbations and craft unlearnable examples. We denote the perturbation generator by $\mathcal{G}_\phi$, implemented as a standard encoder–decoder network. The architecture consists of three down-sampling convolution layers, four residual blocks [35], and three transposed convolution layers. The detailed architecture is shown in Table 8.

Table 8: The architecture of the perturbation generator.

Block Name	Layer	Number
Down-sampling layers		
Conv	Conv (3 × 3) InstanceNorm ReLU	× 3
Bottleneck layers		
Residual	ReflectionPad Conv (3 × 3) BatchNorm ReLU ReflectionPad Conv (3 × 3) BatchNorm	× 4
Up-sampling layers		
ConvTranspose	ConvTranspose (3 × 3) InstanceNorm ReLU	× 2
ConvTranspose	ConvTranspose (6 × 6) Tanh	× 1

Exact construction and optimization order. For each clean image, cosine k-means assigns every spatial token to exactly one of M proto-regions. Thus, the constructed regions satisfy

$$\bigcup_{a=1}^{M} U_i^a = \{1,\dots,N_t\}, \qquad U_i^a \cap U_i^b = \emptyset \quad (a \neq b). \tag{24}$$

The clean last-layer features determine the original support $P_i(n)$, the directed region partner $\psi_i(a)$, and the partner-derived support $\widetilde{P}_i(n)$. We select a spatially adjacent partner with high boundary-contact and feature-dissimilarity score defined in the implementation details. Reciprocal pairing is avoided whenever another valid adjacent candidate exists; otherwise, the available adjacent candidate is retained. The same partner assignment determines the directed boundary band, token weights $w_i(n)$, shifted correspondence $\pi_i(n)$, and pseudo-region label $\widetilde{y}_i(n) = \psi_i(u_i(n))$.

A token-level example. Suppose token n belongs to proto-region A and $\psi_i(A) = B$, where B is adjacent to A. The original support $P_i(n)$ consists of the most feature-similar tokens to n within A (excluding n itself), whereas $\widetilde{P}_i(n)$ selects its most similar clean-feature tokens from B. Thus, affinity replacement encourages the protected token to favor a misleading partner region over its

Algorithm 1 Training and deployment of DRIFT

Require: Unlabeled image set $\mathcal{D}$, frozen encoder F, region number M, support size K, perturbation budget ϵ, inner steps S, learning rates α, β
Ensure: Trained perturbation generator G_ϕ

Initialize the generator G_ϕ and auxiliary learner h_θ
for each training epoch and minibatch $\mathcal{B} \subseteq \mathcal{D}$ **do**
  for each image $x_i \in \mathcal{B}$ **do**
    Extract frozen clean features $T_i^{(\ell)} = F^{(\ell)}(x_i)$
    Partition T_i into $\mathcal{U}_i$ by cosine k-means
    Construct original Top-K supports $P_i(n)$ for all tokens
    Select adjacent region partners ψ_i using $s_i(a, b)$
    Construct pseudo supports $\widetilde{P}_i(n)$
    Construct $\pi_i(n)$, $w_i(n)$, and $\widetilde{y}_i(n) = \psi_i(u_i(n))$
  end for
  for $s = 1, \ldots, S$ **do**
    Form $x_i^u = x_i + G_\phi(x_i)$ with $\|G_\phi(x_i)\|_\infty \le \epsilon$
    Compute protected last-layer features and $\mathcal{L}_{\text{fit}}$
    $\theta \leftarrow \theta - \alpha \nabla_\theta \mathcal{L}_{\text{fit}}$
  end for
  Freeze θ; recompute protected features as needed
  Compute $\mathcal{L}_{\text{rep}} = \mathcal{L}_{\text{aff}}^{\text{rep}} + \mathcal{L}_{\text{corr}}^{\text{rep}}$
  $\phi \leftarrow \phi - \beta \nabla_\phi (\mathcal{L}_{\text{rep}} + \mathcal{L}_{\text{fit}})$
end for
return G_ϕ; protect new images with one bounded generator pass

original region. For cross-layer replacement, the facing boundary bands of A and B determine a unit direction toward B; rounding this direction yields $\Delta_i(n)$ and the shifted position $\pi_i(n)$. Consequently, the same mapping $A \to B$ replaces both the within-region affinity and the original same-position cross-layer correspondence. The auxiliary learner uses the target $\widetilde{y}_i(n) = B$ for token n. All of these targets are computed from clean-image features alone.

Alternating updates. The auxiliary learner and the perturbation generator cooperate rather than optimize adversarial objectives. With ϕ fixed, the learner is updated for S inner steps to minimize $\mathcal{L}_{\text{fit}}$. During the generator update, θ is held fixed and the generator minimizes $\mathcal{L}_{\text{rep}} + \mathcal{L}_{\text{fit}}$. Gradients of the fitting loss still pass through the frozen encoder's computational graph to the protected images and G_ϕ, but are not propagated through the inner parameter updates of h_θ. This is a first-order truncated alternating optimization scheme, rather than a GAN-style discriminator/generator min–max objective.

D.2 SAM 3 Evaluation Protocol

We evaluate DRIFT on Pascal VOC 2012 with SAM 3 in two complementary settings, designed to separate an inference-time effect of input perturbations from the effect of learning on protected training data.

Frozen inference. We apply the same pretrained SAM 3 checkpoint and prompting protocol to clean and DRIFT-protected validation images. All model parameters remain frozen. This setting measures the effect of adding DRIFT perturbations directly to the inference inputs, without additional training.

Encoder fine-tuning. Starting from the same pretrained checkpoint, we fine-tune only the SAM 3 image encoder on either the clean or DRIFT-protected Pascal VOC training images; all remaining model modules are held fixed. The two runs follow the same training and prompting protocols and are both evaluated on the original clean validation images. Segmentation labels are used for downstream fine-tuning and evaluation but are never accessed when constructing the DRIFT perturbations.

As reported in the main-text SAM 3 evaluation, the frozen model shows a 4.2-point mIoU decrease from direct input perturbation, while training with protected images causes a 21.1-point decrease

relative to clean-data fine-tuning. This comparison supports the interpretation that DRIFT mainly impairs learning from protected data rather than merely making protected inputs difficult to segment.

D.3 Why Protection Can Be Weaker on Large Regions

DRIFT places stronger replacement weight near the directed boundary band B_U of a proto-region U. For $n \in U$, this weight is

$$w_U(n) = \exp\left(-\frac{d(p_n, B_U)^2}{2\sigma^2}\right). \tag{25}$$

To characterize how much of a region lies near the affected boundary, we define its mean replacement weight as

$$\rho(U) = \frac{1}{|U|}\sum_{n\in U} w_U(n). \tag{26}$$

The Gaussian decay makes tokens near the facing boundary contribute more strongly, whereas tokens deep in the region interior receive smaller weights. For a compact region, if the effective affected band has width proportional to σ, its total weight scales approximately with $\sigma|B_U|$. Under the additional assumption that the length of the partner-facing boundary scales with the linear size of the region, $|B_U| = O(\sqrt{|U|})$, we obtain the heuristic scaling relation

$$\rho(U) \;\propto\; \frac{\sigma|B_U|}{|U|} \;=\; O\left(\frac{\sigma}{\sqrt{|U|}}\right). \tag{27}$$

Thus, for similarly shaped regions and a fixed boundary-decay scale, a larger region tends to have a smaller average boundary-aware replacement weight. More interior tokens may consequently preserve parts of the original affinity and cross-layer correspondence, whereas small or thin regions can be affected more broadly. This is consistent with the qualitative challenging cases dominated by large foreground objects. The scaling is an explanatory geometric approximation, not a universal bound for arbitrary disconnected or irregular regions; $\rho(U)$ is also not itself a direct measure of downstream unlearnability.

D.4 Implementation of Baselines

Following UnSeg, we consider both generic perturbation-based UE baselines and a segmentation-specific baseline for comparison. Specifically, SynPer [23], AR [25], and CUDA [24] are treated as *annotation-free* perturbation baselines, since they do not require segmentation annotations during perturbation generation. These methods were originally developed for image-level learning settings, so we adapt them to the segmentation benchmarks by applying their perturbation-generation procedures to the segmentation training images and then training downstream segmentation models on the resulting protected datasets under the same protocol as our method. For a fair comparison, their deployment in the segmentation setting follows the protocol used in UnSeg [19]. In addition, we include UnSeg [19] as a *segmentation-specific* baseline. Unlike the generic baselines and DRIFT, UnSeg uses segmentation-style mask annotations during perturbation generation to define protected regions and optimize region-aware perturbations.

For a fair comparison, we reproduce all baselines using their official implementations when available, or otherwise follow the configurations reported in the original papers and in UnSeg as closely as possible. We align the perturbation budget with our setting ($\epsilon = 8/255$) and use the same downstream segmentation architectures, training splits, and evaluation protocols across methods. Unless otherwise specified, all compared methods are evaluated on the same protected training sets and tested under the same segmentation benchmarks as DRIFT.

E More Ablation Study

E.1 Several classes ablation results

To complement the overall mIoU ablation in the main text, we show the overall mIoU and per-class mIoU on selected representative categories in Figure 3. The heatmap shows that the selected class behavior is generally consistent with the overall trend: removing any DRIFT component leads to

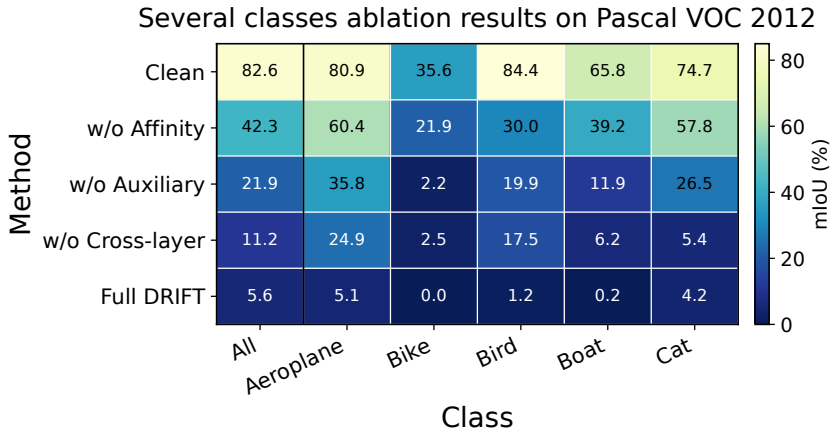


Figure 3: Ablation results of DRIFT on selected Pascal VOC 2012 classes.

performance recovery, while Full DRIFT achieves the strongest degradation across both the overall metric and individual categories. In particular, removing pseudo affinity replacement results in the most substantial recovery for most reported categories, confirming that replacing the original token-level region affinity with pseudo affinity is the dominant factor behind DRIFT's protection effect. Removing auxiliary structure fitting also noticeably weakens the performance, suggesting that the constructed pseudo-structure must be effectively bound to the protected images to be adopted by downstream learners. In comparison, removing cross-layer correspondence replacement gives a smaller but still visible recovery, indicating that cross-layer disruption provides complementary benefits by further weakening multi-scale structural alignment. These per-class results support the component analysis in the main text and show that the effectiveness of DRIFT does not rely on a single category-specific failure case.

E.2 Ablation of pseudo-structure construction

We further ablate the key design choices in pseudo-structure construction on Pascal VOC 2012. DRIFT first partitions clean frozen token features into group-aware proto-regions, assigns each proto-region a directed non-reciprocal partner from adjacent and size-compatible regions, constructs partner-derived pseudo supports, and applies boundary-aware weighting in the replacement objectives. We consider the following variants: *Random region partition*, which replaces group-aware partition with random token grouping; *Random partner mapping*, which assigns partner regions randomly rather than using adjacent and size-compatible ones; *Reciprocal partner mapping*, which replaces directed non-reciprocal assignment with symmetric swapping; and *Uniform weighting*, which removes boundary-aware modulation and applies the replacement pressure uniformly over tokens.

As shown in Table 9, Full DRIFT achieves the strongest unlearnability, reducing the clean-test mIoU to 5.6. *Random region partition* and *Random partner mapping* perform worst, showing that both meaningful clean geometry and structurally valid partner assignment are essential. *Reciprocal partner mapping* is clearly better than random partner assignment but still much worse than Full DRIFT, suggesting that directed, non-reciprocal transfer is more effective than symmetric swapping. *Uniform weighting* further improves over these variants but remains weaker than Full DRIFT, indicating that boundary-aware weighting helps concentrate the replacement pressure on structurally sensitive regions. Overall, these results show that DRIFT relies on the joint effect of group-aware partition, directed partner assignment, partner-derived pseudo support, and boundary-aware replacement weighting.

E.3 Parameter sensitivity

Sensitivity to the number of proto-regions M. We study the sensitivity of DRIFT to the number of proto-regions M used in pseudo-structure construction. Recall that M controls how the clean frozen token features are partitioned into group-aware proto-regions, which further determine the

Table 9: Ablation of pseudo-structure construction on Pascal VOC 2012 semantic segmentation. Lower mIoU indicates stronger unlearnability. Full DRIFT uses group-aware proto-regions, directed non-reciprocal partner mapping, partner-derived pseudo support, and boundary-aware replacement weighting.

Method	All	Aeroplane	Bike	Bird	Boat	Cat
Clean	82.6	80.9	35.6	84.4	65.8	74.7
Random region partition	38.3	55.7	18.5	32.7	37.8	50.4
Random partner mapping	33.1	53.2	18.6	24.4	36.7	48.6
Reciprocal partner mapping	21.5	35.7	13.3	20.6	16.9	11.0
Uniform weighting	16.5	21.8	16.9	15.7	10.4	9.9
Full DRIFT	**5.6**	**5.1**	**0.0**	**1.2**	**0.2**	**4.2**

Table 10: Sensitivity to the number of proto-regions M on Pascal VOC 2012 semantic segmentation. Lower mIoU indicates stronger unlearnability.

M	All	Aeroplane	Bike	Bird	Boat	Cat
5	22.0	35.7	13.3	17.0	20.5	13.3
10	15.5	21.8	16.9	15.8	10.4	19.7
20	**5.6**	**5.1**	**0.0**	**1.2**	**0.2**	**4.2**
30	11.2	16.9	2.5	17.5	6.2	5.4
40	28.5	25.1	11.2	40.6	31.5	38.9

real support, directed partner mapping, pseudo support, and pseudo cross-layer correspondence. Therefore, M affects both the granularity and stability of the constructed pseudo-structure.

As shown in Table 10, DRIFT achieves the strongest unlearnability at $M = 20$, where the clean-test mIoU drops to 5.6. When M is too small, the proto-regions become overly coarse. In this case, many tokens lie far from directed region boundaries, making the boundary-aware replacement pressure less effective over the whole region and leaving more clean region-level affinity intact. As a result, the protective effect becomes weaker. In contrast, when M is too large, the proto-regions become overly fragmented. Although more tokens are close to region boundaries, the directed partner regions also become small and less stable, providing insufficient partner-derived support for constructing a learnable misleading pseudo-structure. Consequently, the protection strength drops again. These results suggest that an intermediate granularity is important: it should be fine enough to expose tokens to effective structural replacement, but coarse enough to preserve stable region-level partner structure. Overall, we use $M = 20$ as the default setting.

We also test the effectiveness of M across three datasets in Table 11. Across all three datasets, moderate variations of M consistently retain strong protection, and the same default $M = 20$ is used without dataset-specific tuning. This indicates that DRIFT is not dependent on a dataset-specific or narrowly optimized choice of M.

Table 11: Cross-Dataset M effectiveness.

M	VOC	Cityspaces	ADE20K
10	15.5	18.1	19.5
20	**5.6**	**14.5**	**11.5**
30	11.2	19.3	16.4

Sensitivity to the support size K. We further study the sensitivity of DRIFT to the support size K, which determines how many neighboring tokens are selected to form the support set for each token during pseudo-structure construction. Intuitively, K controls the density of the affinity signal used in both real-support approximation and pseudo-support replacement. Therefore, it directly affects how informative and stable the constructed pseudo-structure is.

As shown in Table 12, DRIFT achieves the strongest unlearnability at $K = 8$, where the clean-test mIoU drops to 5.6. When K is too small (e.g., $K = 2$), the support set becomes overly sparse, so each token receives only limited affinity evidence. This makes the pseudo-replacement signal weaker and less stable, resulting in noticeably reduced protection strength. As K increases to a moderate range (e.g., $K = 4$ to 12), the support becomes sufficiently informative while remaining

Table 12: Sensitivity to the support size K on Pascal VOC 2012 semantic segmentation. Lower mIoU indicates stronger unlearnability.

K	All	Aeroplane	Bike	Bird	Boat	Cat
2	11.2	6.8	14.6	7.6	13.4	9.2
4	6.2	9.1	1.2	1.4	3.2	8.9
8	**5.6**	**5.1**	**0.0**	**1.2**	**0.2**	**4.2**
12	6.3	10.1	0.4	4.9	2.0	8.2
16	6.1	11.7	0.0	1.3	1.1	6.4

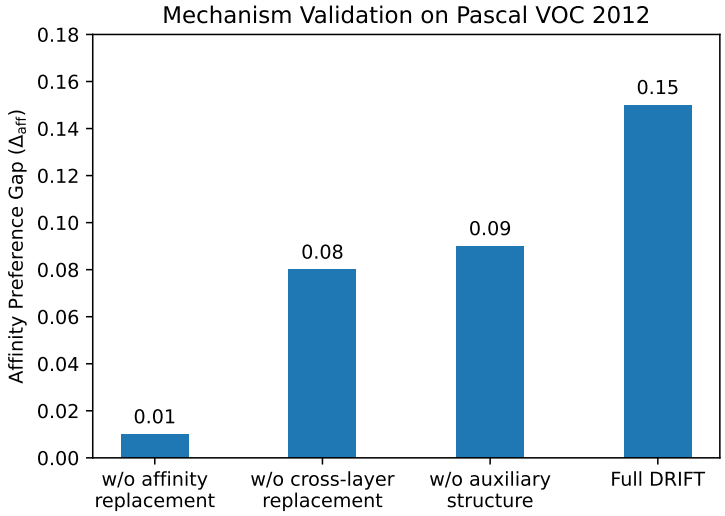


Figure 4: Affinity preference gap under different ablation variants on Pascal VOC 2012. Larger values indicate a stronger preference for pseudo support over the original real support.

selective, leading to consistently strong unlearnability, with the best result obtained at $K = 8$. When K becomes too large (e.g., $K = 16$), the support set becomes less selective and may exceed what the partner region can stably provide, especially when the partner region itself is not large enough. In this case, the pseudo support may include less compatible tokens, which weakens the quality of the induced pseudo-structure and slightly reduces the performance. Overall, these results suggest that DRIFT is relatively robust to K within a moderate range, while a balanced support size gives the strongest protection in our setting.

E.4 Mechanism Validation

To further verify whether DRIFT truly shifts perturbed features from the real structure toward the pseudo-structure, we evaluate whether a perturbed token becomes more compatible with its pseudo support than with its original real support. For image $\boldsymbol{x}_i$, we define

$$\Delta_{\text{aff}}^{(i)} = \frac{1}{N_t} \sum_{n=1}^{N_t} \left[\frac{1}{|\widetilde{\mathcal{P}}_i(n)|} \sum_{p \in \widetilde{\mathcal{P}}_i(n)} \cos\big(T_{i,n}^u, T_{i,p}^u\big) - \frac{1}{|\mathcal{P}_i(n)|} \sum_{q \in \mathcal{P}_i(n)} \cos\big(T_{i,n}^u, T_{i,q}^u\big) \right], \tag{28}$$

and average it over the dataset:

$$\Delta_{\text{aff}} = \frac{1}{N} \sum_{i=1}^{N} \Delta_{\text{aff}}^{(i)}. \tag{29}$$

A larger Δ_{aff} indicates that perturbed tokens exhibit a stronger preference for pseudo support over the original clean support.

We compute Δ_{aff} for different ablation variants of DRIFT on Pascal VOC 2012, including *w/o affinity replacement*, *w/o cross-layer replacement*, *w/o auxiliary structure fitting*, and *Full DRIFT*. As shown

Table 13: Sensitivity to frozen feature extractors on Pascal VOC 2012 using DeepLabV3. The perturbation generator is retrained for each extractor. Lower mIoU indicates stronger protection.

Frozen Feature Extractor	mIoU ↓
DINOv2	5.6
CLIP-ViT	6.2
MAE	5.4

in Figure 4, Full DRIFT achieves the largest positive affinity preference gap, indicating that perturbed tokens are, on average, more compatible with their pseudo support than with their original real support. This averaged cosine gap is non-trivial because it is computed over all tokens and support pairs in the dataset, rather than over selected high-response regions only. When affinity replacement is removed, the gap nearly vanishes, confirming that this component is the primary driver for shifting token-level compatibility from the original real support toward the pseudo support. Removing cross-layer replacement or auxiliary structure fitting also reduces the gap, suggesting that these components further help stabilize the adoption of the pseudo-structure, although their effects are secondary to the direct affinity replacement objective. Overall, these results provide diagnostic evidence that DRIFT does not merely perturb features arbitrarily, but systematically redirects token-level affinity toward the constructed pseudo-structure.

E.5 Sensitivity to Frozen Feature Extractors

We investigate whether DRIFT relies on a specific pretrained feature extractor for pseudo-structure construction. In addition to DINOv2, we consider CLIP-ViT and MAE, which employ different pretraining objectives. For each extractor, we retrain the perturbation generator and evaluate the resulting protected images on Pascal VOC 2012 using DeepLabV3.

As shown in Table 13, DRIFT remains effective across all three extractors, with downstream mIoU ranging from 5.4 to 6.2. Replacing DINOv2 with CLIP-ViT or MAE does not substantially weaken the protection effect. These results suggest that DRIFT does not depend on a particular pretrained representation. Instead, its effectiveness mainly relies on constructing misleading pseudo-structures from relative token relationships in the frozen feature space.

E.6 Cross-Dataset Generator Transferability

Although DRIFT optimizes a perturbation generator for each protected dataset by default, we further investigate whether the learned generator can transfer to unseen datasets without additional optimization.

Specifically, we use Pascal VOC 2012 as the held-out target dataset and train perturbation generators using ADE20K, Cityscapes, or their combined unlabeled training images. The trained generators are then directly applied to Pascal VOC without any target-specific fine-tuning. We evaluate the protected images using DeepLabV3 under the same downstream training protocol.

As shown in Table 14, the generator trained directly on Pascal VOC achieves the strongest protection, reducing the downstream mIoU to 5.6. Nevertheless, generators trained on other datasets also remain effective, yielding mIoU values between 11.7 and 18.5, compared with 82.6 under clean training. These results demonstrate that the learned perturbation generator can transfer across datasets without accessing the target dataset during generator optimization. While dataset-specific optimization provides stronger protection, it is not strictly necessary for DRIFT to achieve substantial unlearnability.

E.7 Computational Efficiency

We compare the computational cost of DRIFT with UnSeg, which also requires offline optimization of a perturbation generator. We distinguish between the offline optimization cost and the deployment cost of generating protected images.

Offline optimization cost. We report GPU-hours measured on a single NVIDIA L40S GPU. Under our reproduction, UnSeg requires approximately 41 GPU-hours to train a universal generator using the

Table 14: Cross-dataset generator transferability on Pascal VOC 2012 using DeepLabV3. Generators trained on external datasets are directly applied to Pascal VOC without target-specific fine-tuning. Lower mIoU indicates stronger protection.

Generator Training Data	VOC mIoU ↓
Pascal VOC (target-specific)	5.6
ADE20K	18.5
Cityscapes	11.7
ADE20K + Cityscapes	14.2

Table 15: Computational cost comparison between UnSeg and DRIFT on a single NVIDIA L40S GPU. Offline GPU-hours are reported for different optimization scopes and are not directly comparable. Generation costs are measured under the same deployment setting.

Method	Offline (GPU-h)	Peak Memory (MB)	Time/Image (ms)
UnSeg	41	2901	113
DRIFT	1.5	112.6	3.57

external HQSeg-44K dataset with segmentation supervision. In contrast, DRIFT performs annotation-free, dataset-specific optimization, requiring 1.5 GPU-hours on Pascal VOC 2012. Across the other datasets, DRIFT requires approximately 0.6, 1.8, and 8.4 GPU-hours on Cityscapes, ADE20K, and COCO, respectively.

However, these offline costs are not directly comparable: UnSeg trains a universal generator that can be reused across datasets, whereas DRIFT optimizes a generator separately for each protected dataset. Therefore, the reported GPU-hours reflect different optimization scopes.

Deployment efficiency. For a controlled comparison, we evaluate the trained generators under the same GPU, input resolution, and batch size. As shown in Table 15, DRIFT requires only 3.57 ms per image and 112.6 MB of peak GPU memory, compared with 113 ms and 2901 MB for UnSeg. Once trained, DRIFT generates perturbations through a single generator forward pass, enabling efficient protection of large image collections.

F Visualization

F.1 Qualitative visualization of feature structure

To further understand how DRIFT affects the structural cues learned by downstream segmentation models, we provide qualitative visualizations of representative training samples. For each example, we compare the response map from a downstream model trained on clean data with that from the same architecture trained on DRIFT-protected data. Importantly, these response maps are extracted from the downstream segmentation model rather than the frozen encoder used by DRIFT, avoiding circular evidence from the feature space used for pseudo-structure construction. Specifically, we visualize the spatial response map computed as the ℓ_2 norm of the last decoder feature at each spatial location, normalized to $[0, 1]$ for display.

Figure 5 presents several successful cases. For clean images, the response maps typically show coherent activations over the main object or scene region, reflecting the original segmentation-related structure. After applying DRIFT, although the protected images remain visually recognizable, the corresponding responses become more fragmented, less object-centric, partially shifting toward local edges, object parts, or regions outside the original object extent. In many cases, the protected responses no longer cover the full semantic extent of the original object, but instead concentrate on incomplete or distorted local patterns. This visual pattern reflects the intended mechanism of DRIFT: replacing the original structural cues with a misleading pseudo-structure rather than simply corrupting input appearance.

Figure 6 shows several challenging cases. These examples mainly correspond to large-area or spatially dominant foregrounds, where the protected response maps still retain partial overlap with the original object extent. Compared with the successful cases, the responses are less fully redirected away from

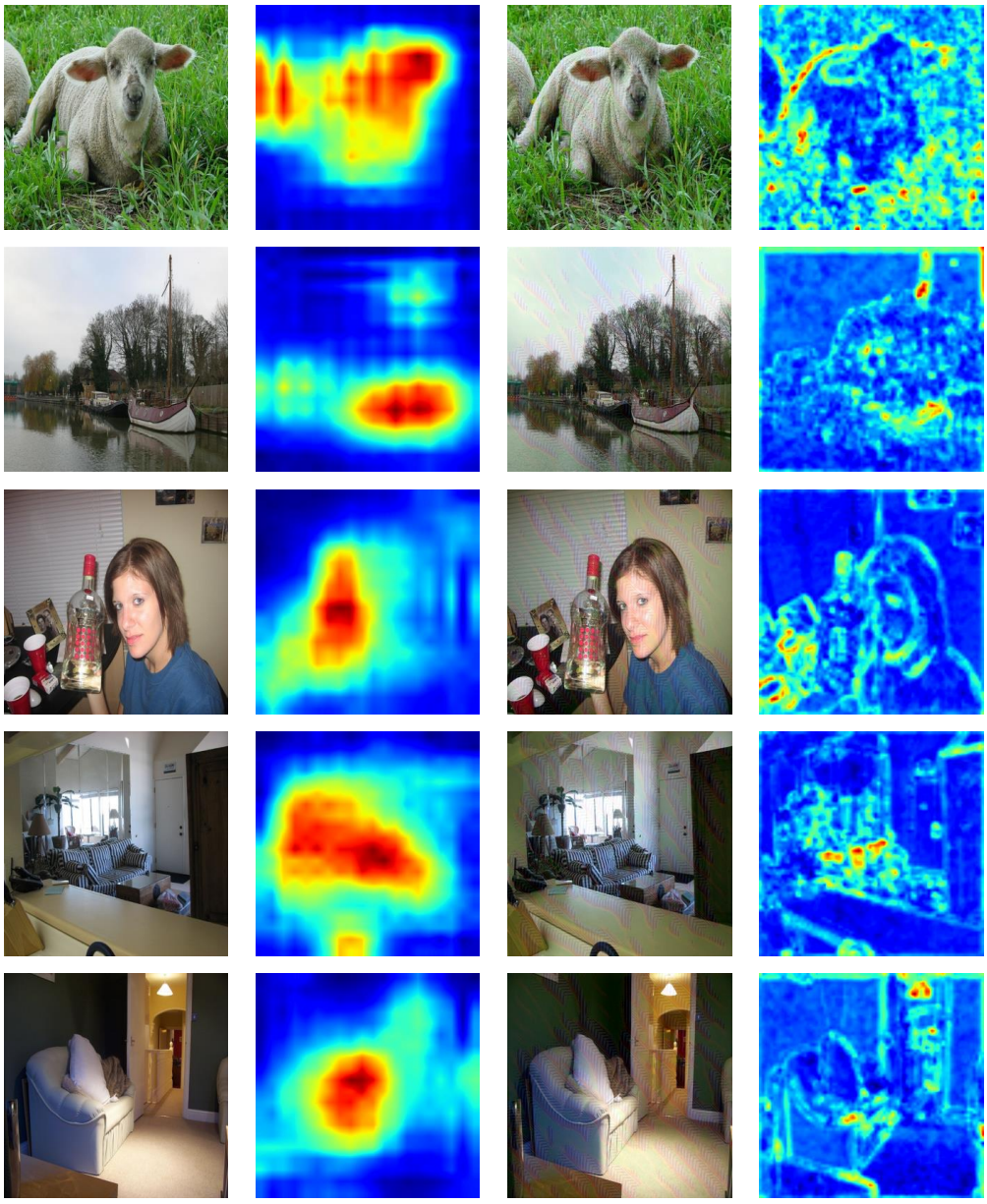


Figure 5: Qualitative visualization of successful DRIFT cases. (a) Clean image. (b) Feature map of the clean image. (c) DRIFT-protected image. (d) Feature map of the protected image.

the semantic object body, although they remain less coherent and more biased toward boundaries or local parts. This observation suggests that DRIFT is generally effective at distorting segmentation-related structure, but its protection can be weaker on very large or structurally dominant regions. As noted in the limitation discussed in the conclusion, when a single semantic region dominates the spatial extent, boundary-aware replacement pressure affects a smaller fraction of the object interior, making the original internal support harder to fully disrupt.

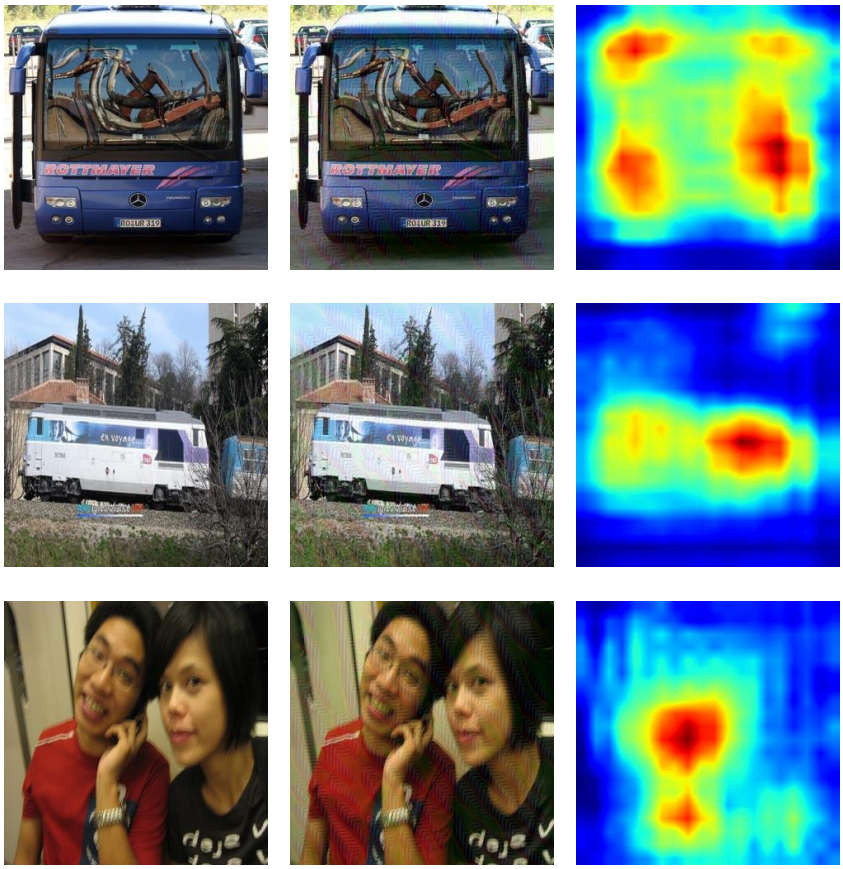


Figure 6: Qualitative visualization of challenging DRIFT cases on training images. (a) Clean image. (b) DRIFT-protected image. (c) Feature map of the protected image.

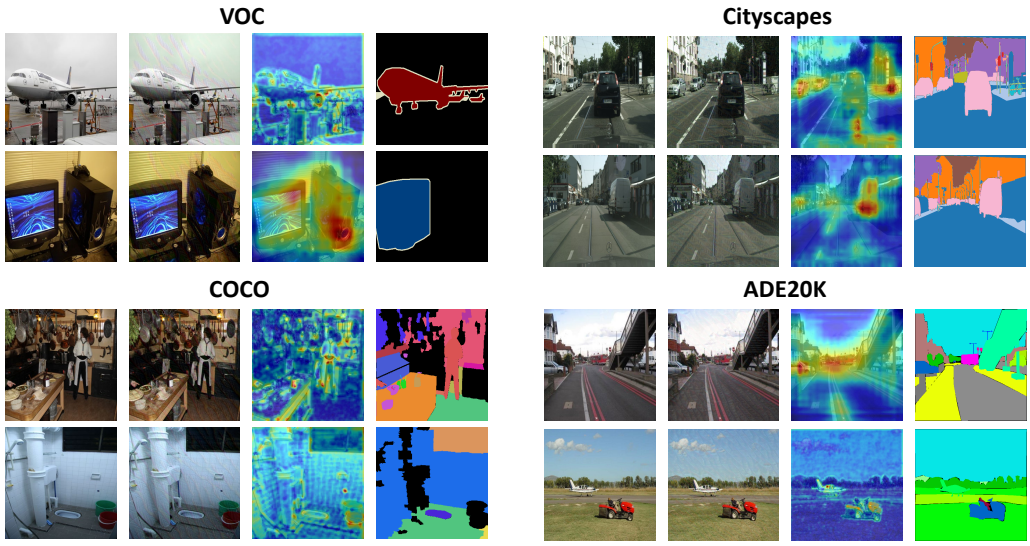


Figure 7: Cross-dataset qualitative visualization of DRIFT on Pascal VOC 2012, COCO, Cityscapes, and ADE20K. From left to right: clean image, DRIFT-protected image, feature response map of the protected sample, and ground-truth annotation.

F.2 Cross-dataset qualitative visualization

We further provide qualitative visualizations across four datasets, including Pascal VOC 2012, COCO, Cityscapes, and ADE20K, to examine how DRIFT affects feature responses under different scene types and segmentation settings. For each example, we show the clean image, the DRIFT-protected image, the corresponding feature response map of the protected sample (computed as in Sec. F.1), and the ground-truth annotation for reference. The ground-truth labels are used only for visualization and qualitative comparison, not for perturbation generation.

As shown in Figure 7, the protected images remain visually close to the clean inputs, indicating that DRIFT does not rely on obvious perceptual corruption. However, the corresponding response maps are often no longer well aligned with the full semantic extent of the ground-truth objects or regions. Instead, activations become more localized around boundaries, object parts, thin structures, or fragmented local regions. On Pascal VOC, this behavior can be observed on foreground objects such as airplanes and monitors, where protected responses emphasize partial contours or local regions rather than the whole object body. On COCO and ADE20K, where scenes are more cluttered and contain multiple semantic regions, the responses become more dispersed and tend to follow scattered local structures that do not correspond to any single ground-truth region. On Cityscapes, the protected responses fragment into boundary-localized cues across roads, vehicles, and poles, rather than forming coherent region-level support that a segmentation network can use to delineate full semantic regions.

The four datasets cover very different scene complexities, ranging from object-centric VOC, to cluttered multi-object COCO and ADE20K, and to structured urban Cityscapes. Across all of them, the protected responses are substantially less aligned with full semantic regions while the input appearance remains visually plausible, providing qualitative evidence that DRIFT's structural disruption generalizes across scene types rather than relying on dataset-specific cues.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: We present the contributions and scope in the abstract and introduction.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: We have discussed the limitations in the conclusion section.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [N/A]

Justification: This paper does not include theoretical results.

Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: All the information required to reproduce the main results is included in the experiment settings.

   Guidelines:

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

5. **Open access to data and code**

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

Answer: [Yes]

Justification: Source code will be released after acceptance. We also provide a demo code snippet of the method implementation in the supplementary materials.

Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: Relevant details are discussed in the experiment settings.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: We report results averaged over five runs with different random seeds.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: We describe the compute resources in the implementation details.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: The research conducted in the paper conforms with the NeurIPS Code of Ethics.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: The potential social impacts are discussed in the Appendix A.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

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

    Answer: [N/A]

    Justification: This paper aims at data protection, and there are no safety risks.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: Existing assets are properly cited.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

    Answer: [Yes]

    Justification: Our code will be released after acceptance with detailed documents.

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: This paper does not involve these contents.

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: The experiments in this paper do not involve human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

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

Answer: [N/A]

Justification: LLM is used only for polishing the writing. The core method development in this research does not involve LLMs as important components.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.