

Cross-Domain Robustness of CNN vs Transformer Detectors for Underwater Object Detection

Buthaina Almulla Abdulla Alshehhi Shamma Yaqoob

Mohamed bin Zayed University of Artificial Intelligence

{buthaina.almulla, abdulla.alshehhi, shamma.yaqoob}@mbzuai.ac.ae

Abstract

Underwater object detection is limited by two compound-ing problems: strong domain shift across datasets caused by variable water conditions (attenuation, haze, turbidity, and blur), and label noise arising from missed annotations in low-visibility images. We systematically compare a CNN detector (YOLOv8) and a Transformer detector (DETR) under cross-dataset transfer, physically motivated underwater corruptions, and noisy supervision, and propose a training strategy that improves cross-domain robustness without re-annotation. Our method combines (i) a raw $\leftrightarrow$ enhanced feature-consistency loss using a cheap CLAHE-based underwater enhancer as an auxiliary view, and (ii) a noise-aware supervision reweighting that down-weights samples whose raw and enhanced features strongly disagree. Experiments on DUO and RUOD show three main findings. First, YOLO achieves higher absolute performance than DETR in-domain and cross-domain, but relative robustness is direction-dependent: DETR retains more in-domain mAP under DUO $\rightarrow$ RUOD shift (66.2% vs. 56.4%), whereas YOLO retains more under RUOD $\rightarrow$ DUO (40.3% vs. 29.9%). Second, haze and turbidity are the most destructive corruptions, while blur is the least harmful. Third, the proposed method is beneficial but detector and direction-dependent: for DETR, the full method improves DUO $\rightarrow$ RUOD cross-domain mAP from 0.285 to 0.294 and AP50 from 0.608 to 0.614, whereas for YOLO the best variant is weight only on DUO $\rightarrow$ RUOD and full on RUOD $\rightarrow$ DUO.

1. Introduction

Underwater detection is a critical safety and ecology task: autonomous underwater vehicles rely on it for cleaning marine-debris, biodiversity monitoring, inspection of the pipeline and hull, and reef surveys. However, an underwater detector that works perfectly on one data set often fails when deployed in water that looks different: a different

camera, a different depth, a different level of turbidity, a different sun angle [3]. This problem is not merely empirical: it is embedded in the physics of underwater image formation. Red light is absorbed within meters, producing strong color casts; suspended particles cause forward-scattering (haze) and back-scattering; and low photon counts combined with motion produce blur and noise. Any detector that memorizes the color palette of its training set is fundamentally brittle.

A second, subtler problem has recently surfaced: *benchmark annotation quality itself is a confounder*. Lucas et al. [5] show that when underwater images are enhanced post-hoc, detectors find objects that human annotators missed in the degraded original. These are real objects, correctly detected, yet the evaluation protocol counts them as false positives. This inflates the apparent benefit of simple enhance-then-detect pipelines and penalizes detectors that generalize well from limited supervision.

These two problems together motivate three research questions:

- **RQ1:** Which detector family (CNN or Transformer) transfers better across underwater datasets?
- **RQ2:** How do detectors degrade under *physically-motivated* underwater corruptions, rather than the generic corruptions in ImageNet-C?
- **RQ3:** Can we improve cross-domain robustness *without re-annotation*, using consistency training and noise-aware supervision?

We answer all three through a unified experimental protocol on the DUO [4] and RUOD benchmarks, with 4 shared classes used for fair cross-domain comparison. Our **contributions** are:

1. A controlled comparison of YOLOv8 (CNN) and DETR (Transformer) on two underwater datasets, measuring both absolute in-domain/cross-domain performance and mAP-based retention. YOLOv8 achieves higher absolute performance on every split, while relative robustness is direction-dependent: DETR retains more in-domain mAP under DUO $\rightarrow$ RUOD shift (66.2% vs. 56.4%), whereas YOLO retains more under RUOD $\rightarrow$ DUO

(40.3% vs. 29.9%).

2. A physically-motivated corruption suite (attenuation, haze, turbidity, blur) with five severity levels, producing robustness curves that distinguish graceful degradation (blur, -17%) from catastrophic failure (haze, -86%).
3. A training method that uses an underwater enhancer purely as a *second view* (not as ground truth) to provide a consistency signal, combined with a *noise-aware reweighting* scheme. For DETR, the full method lifts cross-domain mAP from 0.285 to **0.294** ($+0.009$) and AP₅₀ from 0.608 to **0.614** ($+0.006$), outperforming the baseline and either component in isolation on DUO $\rightarrow$ RUOD.
4. A cross-family, cross-directional ablation on both DETR and YOLOv8 showing that gains are jointly detector- and direction-dependent: DETR benefits most from the full combination on DUO $\rightarrow$ RUOD but not on RUOD $\rightarrow$ DUO, whereas YOLO benefits most from noise-aware weighting on DUO $\rightarrow$ RUOD and from the full method on RUOD $\rightarrow$ DUO. This reveals that the effectiveness of consistency training and noise-aware weighting is not uniform across detector families or transfer directions.

2. Related Works

Underwater Object Detection. Methods for underwater object detection fall into two categories: traditional feature-based approaches and deep learning methods. Traditional approaches include sonar and optical imaging with manual feature extraction. Deep learning methods have advanced the field significantly, though challenges such as turbidity, refraction, low contrast, texture loss, and color degradation remain. Studies such as Srividya and Ramya combined learning algorithms with texture features, while Chen et al. proposed monocular visual sensors. Existing datasets are scene-specific and insufficient to assess generalizability across environments, demanding evaluation on multiple datasets [3].

Enhancement and Label Noise. Enhancing underwater images before detection does not always guarantee accurate performance and can sometimes degrade it. Poor object detection results on enhanced images arise mainly from missing human labels: the enhance-then-detect pipeline reports false positives for objects that were real but unlabeled in the degraded original [5]. Reannotation of enhanced images yielded a 5-point precision gain when annotated with 9 labels per image, suggesting enhancement should be used as a preprocessing step before human annotation rather than as a runtime pipeline stage [5].

CNN and Transformer Detectors. YOLO is a state-of-the-art CNN-based detector with fast training and inference times [6], but its inductive bias toward local textures may cause it to struggle under distribution shift. One study proposed an optimized YOLOv8s framework integrating a deformable convolutional feature module, depth-weighted bidirectional feature pyramid, content-aware feature reorganization, and efficient multi-scale attention, improving small-target recognition in underwater environments [7]. However, that study benchmarks on a single dataset and does not assess cross-domain shift. Deformable DETR is an end-to-end transformer-based detector that addresses DETR’s slow convergence and limited spatial resolution through multi-scale deformable attention modules that attend to a small set of keypoints from a reference point, achieving better small-object detection [8]. Its performance in underwater environments is yet to be verified. Our work addresses these gaps by comparing YOLO and DETR across multiple underwater datasets, evaluating not only absolute performance but also cross-domain retention and robustness to underwater corruptions.

3. Method

We train a detector D_θ with two forward passes per image: one on the raw input x and one on its enhanced copy $x_e = E(x)$, where E is a fixed, non-learned underwater enhancer. The overall objective is

$$\mathcal{L} = \mathcal{L}_{\text{sup}} + \lambda \mathcal{L}_{\text{cons}}, \quad (1)$$

where $\mathcal{L}_{\text{sup}}$ is the standard detection loss reweighted by a per-sample confidence score, and $\mathcal{L}_{\text{cons}}$ is a consistency loss that aligns the two views.

3.1. Enhancer E as a second view

We use a CLAHE-based [9] underwater enhancer rather than a learned UIE network. The input image is converted to the LAB colour space; Contrast-Limited Adaptive Histogram Equalization (CLAHE, clip limit 2.0, tile grid 8×8) is applied to the luminance channel L only, leaving the chrominance channels unchanged. The result is converted back to RGB and used as $x_e = E(x)$. We choose this enhancer for three reasons. First, it is deterministic and adds no trainable parameters. Second, its failure modes are known and diverse: it over-corrects already-bright regions and under-corrects extreme haze. Third, and most importantly, we do *not* want x_e to be “correct”, we want it to be *different*. Any transformation that changes the distribution of luminance while preserving object identity is a valid consistency view.

3.2. Consistency loss

Let f and f_e be the intermediate representations produced by the detector on the raw view x and enhanced

view x_e respectively. The exact feature used differs by architecture: for **YOLO**, f is the output of the CNN backbone (layers 0–9),¹ For **DETR**, f is the `encoder_last_hidden_state` — the output of the full Transformer encoder (ResNet-50 backbone + Transformer encoder layers) — reused from the supervised detection forward at no extra computational cost, while f_e is obtained from a single additional no-grad forward on x_e . We implement $\mathcal{L}_{\text{cons}}$ as the mean squared error between these representations:

$$\mathcal{L}_{\text{cons}} = \frac{1}{N} \|f - f_e\|_2^2, \quad (2)$$

where N is the total number of feature elements. Crucially, $\mathcal{L}_{\text{cons}}$ contains no ground-truth labels; it teaches the encoder to produce invariant representations regardless of the luminance distribution of the input, which directly targets the cross-domain failure mode.

3.3. Noise-aware supervised loss

For each image i in a batch of size B , we compute a reliability weight $w_i \in (0, 1]$ from the mean feature disagreement:

$$u_i = \text{Disagree}(f_i, f_{e,i}) = \frac{1}{N} \|f_i - f_{e,i}\|_2^2, \quad (3)$$

$$u_{i,\text{norm}} = \frac{u_i - u_{\min}}{u_{\max} - u_{\min} + \epsilon}, \quad (4)$$

$$w_i = \exp(-\alpha u_{i,\text{norm}}), \quad (5)$$

where $u_{\min}$ and $u_{\max}$ are the minimum and maximum uncertainty values within the current batch and $\alpha = 2.0$ controls the sharpness of down-weighting. High disagreement between f and f_e indicates that either the annotation is incomplete (an object present in x_e but absent from the label) or the scene is genuinely ambiguous; in both cases we want to trust the supervision signal less. The per-image weights are then collapsed into a single batch-level scalar that uniformly scales the detection loss:

$$w = \frac{1}{B} \sum_{i=1}^B w_i, \quad \mathcal{L}_{\text{sup}} = w \cdot \mathcal{L}_{\text{det}}(p, y). \quad (6)$$

This collapsing step is necessary because the Ultralytics YOLO trainer exposes only a batch-level loss at the hook site, and we apply the same treatment to DETR for consistency. For YOLO specifically, because the Ultralytics trainer has already backpropagated the detection loss before our hook fires, w is applied by scaling all parameter gradients in-place rather than scaling the forward-pass loss directly; the result is mathematically equivalent.

¹In the `full` variant, YOLO’s backbone runs three times per batch: the Ultralytics detection forward, a no-grad pass for uncertainty estimation, and a with-grad pass for $\mathcal{L}_{\text{cons}}$.

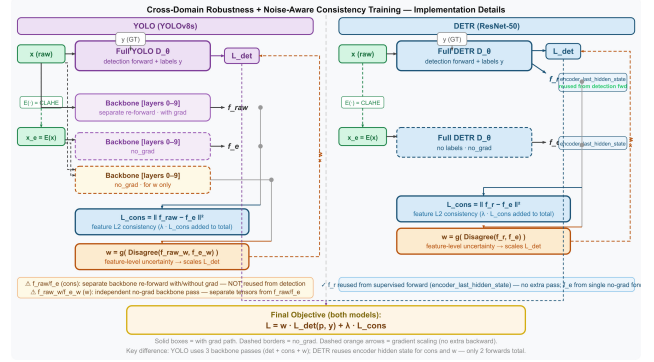


Figure 1. **Overview of the proposed dual-view training framework.** Each image is processed as a raw view and a CLAHE-enhanced view. Feature consistency is enforced between the two representations, while a batch-averaged uncertainty weight modulates the supervised detection update.

3.4. Batch-size sensitivity.

Because u_{norm} is normalized within each batch, at batch size 2 there are only two samples per normalization, producing near-constant $w \approx 1$ and making the weighting functionally equivalent to no weighting. This primarily affects DETR, which trains at batch size 2. YOLO trains at batch size 16 and does not exhibit this collapse (weighting is effective for YOLO throughout training). We therefore expect larger DETR gains if batch size is increased to ≥ 8 .

3.5. Why the combination matters

Consistency alone ($\lambda > 0$, $w \equiv 1$) still trains on every noisy label and is weak early in training because raw and enhanced features are initially similar. Reweighting alone ($\lambda = 0$) has no stable uncertainty signal to act on: without consistency training, $\|f - f_e\|$ reflects appearance variation rather than label ambiguity, and at batch size 2 the within-batch normalization further collapses w toward 1. Only the full objective simultaneously supplies an uncertainty signal and uses it to reduce trust in unreliable supervision. As a consequence of this coupling, as $\mathcal{L}_{\text{cons}}$ decreases over training the disagreement u_i shrinks, the normalized range narrows, and $w_i \rightarrow 1$ asymptotically; the weighting is therefore strongest early in training when raw-versus-enhanced disagreement is most informative about label reliability.

4. Experiments

4.1. Experimental Setup

Datasets. DUO [4] contains 6,671 training images and 1,111 test images across 4 classes: holothurian, echinus, scallop, and starfish. The images are typically green-tinted with dense object layouts (9.5 annotations per image on av-

erage). **RUOD** [2] contains 7,569 training images and 3,297 test images across 10 classes, including the four classes shared with DUO. Compared with DUO, RUOD exhibits brighter and more diverse colour distributions together with sparser annotations, averaging 4.7 instances per image. This difference in annotation density is an important factor in the asymmetric cross-domain behaviour discussed in Section 4.2. To prevent train–test leakage across datasets, we perform perceptual-hash (pHash/DCT) deduplication with Hamming threshold 4 over all 18,648 images and confirm the absence of exact or near-duplicate overlap.

Shared-class evaluation. For cross-domain experiments we restrict both predictions and ground truth to the 4 classes shared by both datasets, and remap category indices to a common zero-indexed space.

Models. We evaluate two detector families. DETR [1] is a Transformer-based end-to-end detector with a ResNet-50 backbone and a Transformer encoder–decoder trained using Hungarian bipartite matching, without hand-crafted anchors or non-maximum suppression. **YOLOv8s** is a CNN-based anchor-free single-stage detector with a hierarchical feature pyramid and strong inductive biases, including local receptive fields and translation equivariance.

Training. DETR is fine-tuned from facebook/detr-resnet-50 for 30 epochs with batch size 2. We use AdamW with weight decay 10^{-4} , gradient clipping at 0.1, mixed-precision training, and a three-group learning-rate schedule with backbone learning rate 10^{-5} and a $10\times$ lr for the Transformer and classification head. YOLOv8s is fine-tuned for 50 epochs with batch size 16 and image resolution 640×640 , using the default Ultralytics training configuration, including SGD with warmup and cosine scheduling, together with mosaic, mixup, and HSV augmentation. Both models are trained independently on DUO and RUOD.

Metrics. We report COCO-style detection metrics computed with pycocotools, including AP@0.5 and mAP@[.5:.95]. All results are measured on held-out test splits not used during training.

4.2. RQ1: Baseline and Cross-Domain Performance

Table 1 reports both in-domain and cross-domain results. YOLO consistently outperforms DETR in-domain: 0.797 vs. 0.694 AP@0.5 on DUO and 0.862 vs. 0.655 on RUOD. The mAP@[.5:.95] gap is also substantial (0.640 vs. 0.426 on DUO; 0.575 vs. 0.351 on RUOD), indicating stronger overall localisation quality for YOLO in-domain.

All models degrade cross-domain, confirming that underwater domain shift is genuine. In absolute terms, YOLO

Table 1. In-domain and cross-domain results. Cross-domain uses the four shared classes only. Retention is cross-domain mAP divided by the matched in-domain mAP.

Train → Test	Model	AP50	mAP	Ret. (%)
DUO → DUO	DETR	0.694	0.426	–
	YOLO	0.797	0.640	–
RUOD → RUOD	DETR	0.655	0.351	–
	YOLO	0.862	0.575	–
DUO → RUOD	DETR	0.608	0.282	66.2
	YOLO	0.629	0.361	56.4
RUOD → DUO	DETR	0.245	0.105	29.9
	YOLO	0.367	0.232	40.3

achieves higher cross-domain performance in both directions: 0.629 AP@0.5 and 0.361 mAP for DUO→RUOD versus DETR’s 0.608 and 0.282, and 0.367 AP@0.5 and 0.232 mAP for RUOD→DUO versus DETR’s 0.245 and 0.105. However, the mAP-based retention ratio reveals a directional difference. Under DUO→RUOD, DETR retains 66.2% of its in-domain mAP, compared with 56.4% for YOLO, indicating stronger relative robustness for DETR in that direction. Under RUOD→DUO, the pattern reverses: YOLO retains 40.3% of its in-domain mAP, whereas DETR retains only 29.9%, showing that the reverse transfer is harder for both models, but especially for DETR.

Directional asymmetry. DUO→RUOD transfers better than RUOD→DUO for both detectors. DETR retains 66.2% versus 29.9% of its in-domain mAP, while YOLO retains 56.4% versus 40.3%. We attribute this asymmetry to three compounding factors: (1) **Annotation density:** DUO’s 9.5 annotations/image forces finer-grained discrimination, whereas RUOD’s 4.7 annotations/image provides weaker supervision. (2) **Label space mismatch:** RUOD-trained models distribute probability mass across 10 classes, diluting confidence for the four shared classes at test time. (3) **Visual distribution direction:** DUO’s narrow green-tinted distribution is effectively nested within RUOD’s broader distribution, so DUO-trained features cover more shared-class appearances in RUOD than the reverse.

CNN vs Transformer generalisation. YOLO’s CNN backbone exploits local receptive fields and translation equivariance that are naturally robust to global colour-style shifts. Low-level edge and texture detectors learned on DUO transfer well to RUOD because the local appearance of shared underwater objects remains similar despite changes in global scene statistics. DETR’s global self-attention learns scene-level context and inter-object relationships that can be more dataset-specific, which may increase sensitivity when that context changes. At the same time, DETR’s stronger mAP retention in the DUO→RUOD

direction suggests that its degradation is less severe relative to its own in-domain baseline.

RQ1 answer: *YOLO achieves higher absolute cross-domain performance in both directions, but relative robustness is direction-dependent: DETR retains more of its in-domain mAP under DUO→RUOD shift, whereas YOLO retains more under RUOD→DUO. The RUOD→DUO direction is harder for both models, especially for DETR.*

4.3. RQ2: Corruption Robustness

We design four physics-motivated underwater image corruptions, each applied at five severity levels ($s \in \{1, 2, 3, 4, 5\}$) to the DUO test set. Figure 2 shows sample DUO images under each corruption at all severity levels. The corruptions simulate real degradation phenomena encountered in underwater imaging:

(1) **Attenuation** simulates wavelength-dependent light absorption, which attenuates red light faster than blue/green at depth. We scale RGB channels independently as $r \leftarrow r(1 - 0.12s)$, $g \leftarrow g(1 - 0.06s)$, $b \leftarrow b(1 - 0.02s)$, and compress contrast as $\text{out} = 128 + (\text{out} - 128)(1 - 0.05s)$.

(2) **Haze/backscatter** simulates light scattering by suspended particles. We blend the image with a scattering color vector $A = [0.6, 0.75, 0.85]$ (blue-dominant, consistent with water optics) at strength $\alpha = 0.08s$: $\text{out} = (1 - \alpha)\text{img} + \alpha A$, followed by contrast reduction $\text{out} = 0.5 + (\text{out} - 0.5)(1 - 0.15s)$.

(3) **Turbidity noise** simulates random scattering from suspended particles using additive Gaussian noise with $\sigma = 5 + 6s$, applied independently to all channels.

(4) **Blur** simulates loss of optical sharpness using Gaussian smoothing with kernel size $k = 1 + 2s$.

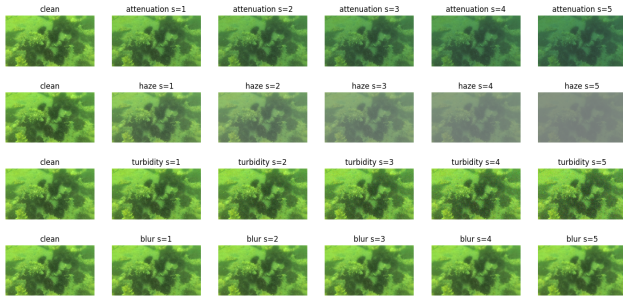


Figure 2. **Visual comparison of underwater corruptions.** Each strip shows the clean reference followed by five increasing severity levels. From top to bottom: attenuation, haze, turbidity, and blur.

Quantitative Results. Figure 3 plots mAP against corruption severity for YOLO and DETR. Table 2 summarizes clean and worst-case ($s = 5$) performance together with

relative drops.

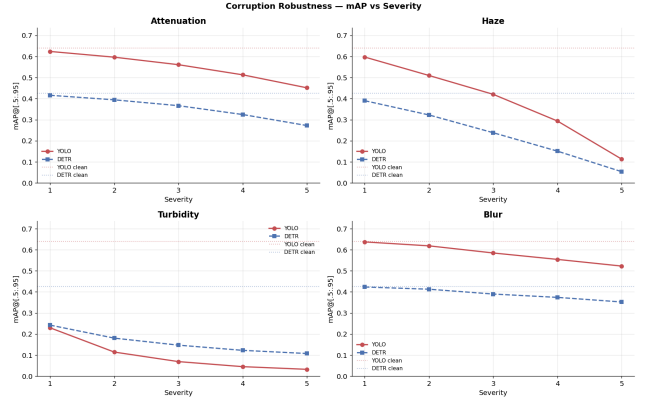


Figure 3. mAP versus corruption severity for YOLO and DETR on the DUO test set. Severity 0 denotes clean images.

Table 2. Corruption robustness summary. Clean denotes mAP on the uncorrupted test set ($s = 0$). For each model, the clean value is shown once and omitted thereafter. $s = 5$ denotes mAP at maximum corruption severity; Drop is the relative reduction from clean to $s = 5$.

Model	Corruption	Clean	$s = 5$	Drop (%)
DETR	Attenuation	0.426	0.272	36.2
	Haze	—	0.053	87.6
	Turbidity	—	0.108	74.6
	Blur	—	0.352	17.4
YOLO	Attenuation	0.640	0.451	29.5
	Haze	—	0.113	82.3
	Turbidity	—	0.032	95.0
	Blur	—	0.523	18.3

Haze is the most damaging corruption. Both models degrade catastrophically under severe haze. DETR falls from 0.426 to 0.053 mAP at severity 5 (87.6% drop), while YOLO falls from 0.640 to 0.113 (82.3% drop). This is expected: haze introduces a dense blue-white scattering veil that simultaneously suppresses contrast, blurs object boundaries, and shifts the color distribution well outside the training domain. At high severities, objects become difficult even for human inspection, making reliable detection nearly impossible. DETR is generally more affected at intermediate severities, consistent with the idea that its global attention mechanism depends more strongly on coherent object boundaries and scene structure, whereas YOLO’s local convolutional features can preserve some usable signal under partial degradation.

Turbidity causes severe but non-uniform degradation. Turbidity reduces DETR mAP by 74.6% (0.426→0.108) and YOLO by 95.0% (0.640→0.032) at severity 5. However, the relative behavior differs by severity. At low corruption levels, DETR degrades more sharply: at severity 1

it drops to 0.242, whereas YOLO retains 0.485. At high corruption levels, the trend reverses and YOLO collapses more severely. This suggests that DETR is more sensitive to moderate feature corruption, likely because noisy feature maps disrupt attention assignments across object and background regions. In contrast, YOLO’s convolutional pyramid appears more resilient to mild noise, but once turbidity becomes extreme, the local texture cues on which it relies are largely destroyed.

Attenuation causes moderate degradation. Attenuation reduces DETR mAP by 36.2% (0.426→0.272) and YOLO by 29.5% (0.640→0.451), making it clearly less harmful than haze or turbidity. This is because attenuation shifts color balance while largely preserving relative object-background structure and boundary information. The degradation curves are smooth and monotonic for both models, consistent with a gradual loss of discriminative color cues rather than abrupt structural corruption. YOLO is modestly more robust than DETR in this setting.

Blur is the most benign corruption and produces a mild non-monotonic response for YOLO. Blur produces the smallest worst-case drops for both models: 17.4% for DETR and 18.3% for YOLO. DETR declines smoothly from 0.426 to 0.352 as blur severity increases. YOLO, however, shows a slight improvement at low severities before declining at higher levels. A plausible explanation is that mild Gaussian smoothing suppresses high-frequency under-water noise, such as suspended particles, caustics, and sensor artifacts, thereby acting as an implicit denoiser. Only at larger kernel sizes does blur begin to remove genuine object-boundary information and reduce performance.

RQ2 answer: *Both detectors degrade substantially under haze and turbidity, with haze the most catastrophic corruption and blur the most benign. Attenuation causes only moderate degradation. YOLO remains stronger in absolute mAP, but robustness depends on corruption type and severity: it is more resilient to attenuation and mild blur, while turbidity reveals a severity-dependent inversion in which DETR degrades faster at low severities but YOLO collapses more at maximum noise levels. Mild blur slightly improves YOLO, suggesting an implicit denoising effect in underwater imagery.*

4.4. RQ3: Ablation Study of Proposed Method

We evaluate our proposed consistency training with noise-aware weighting through controlled ablation studies on cross-domain transfer. We ablate three components independently and in combination: the feature consistency loss L_{cons} , the noise-aware weight w , and their combination (the full proposed method).

For DETR, we train four variants for 30 epochs on DUO starting from `facebook/detr-resnet-50` pretrained weights, and evaluate cross-domain on RUOD using the 4 shared classes. For YOLO, we train four variants for 50 epochs on DUO using the Ultralytics trainer with the standard augmentation stack (mosaic, mixup, HSV jitter), evaluating cross-domain on RUOD using the 4 shared classes. The four ablation variants are: (1) **Baseline**, which uses standard supervised training with L_{det} only; (2) **Cons. only**, which adds consistency loss L_{cons} without weighting ($w = 1$); (3) **Weight only**, which applies noise-aware weighting w to L_{det} without consistency loss; and (4) **Full**, which combines both L_{cons} and w .

Table 3. **Ablation study on cross-domain transfer performance.** We evaluate components on the DUO↔RUOD shift (4 shared classes). ✓ indicates the component is enabled. Best results are in **bold**.

Variant	L_{cons}	w	AP ₅₀	mAP _{50:95}
<i>DETR (DUO → RUOD, 30 epochs)</i>				
Baseline	–	–	0.608	0.285
Cons. only	✓	–	0.608	0.285
Weight only	–	✓	0.572	0.278
Full	✓	✓	0.614	0.294
<i>DETR (RUOD → DUO, 30 epochs)</i>				
Baseline	–	–	0.245	0.105
Cons. only	✓	–	0.239	0.086
Weight only	–	✓	0.242	0.098
Full	✓	✓	0.236	0.084
<i>YOLO (DUO → RUOD, 50 epochs)</i>				
Baseline	–	–	0.629	0.361
Cons. only	✓	–	0.628	0.363
Weight only	–	✓	0.647	0.372
Full	✓	✓	0.608	0.352
<i>YOLO (RUOD → DUO, 50 epochs)</i>				
Baseline	–	–	0.367	0.232
Cons. only	✓	–	0.360	0.230
Weight only	–	✓	0.358	0.229
Full	✓	✓	0.374	0.238

The ablations show that the proposed components help differently across detector families and transfer directions. For DETR, the full method is best on DUO→RUOD, improving over the baseline from 0.608 to 0.614 AP@0.5 and from 0.285 to 0.294 mAP, but not on RUOD→DUO. YOLO shows a clearer direction-dependent pattern: on DUO→RUOD, *weight only* performs best (0.647 AP@0.5, 0.372 mAP), whereas on RUOD→DUO the *full* method performs best (0.374 AP@0.5, 0.238 mAP). Overall, the results suggest that noise-aware weighting is broadly useful, while the benefit of adding consistency depends on both detector family and transfer direction.

Why consistency loss alone does not help (DETR).

The `cons` only variant matches the baseline exactly (0.608/0.285). At the start of training, raw and CLAHE-enhanced features are nearly identical, so $\mathcal{L}_{\text{cons}} \approx 0$ and contributes negligible gradient signal. Without the weighting component, $\mathcal{L}_{\text{cons}}$ alone provides too little task-relevant signal to alter the decision boundary.

Why weight only slightly hurts (DETR). With batch size 2, the within-batch uncertainty normalisation in Eq. 4 has only two samples to normalise over, producing near-constant $w \approx 1.0$ throughout training. The result is functionally close to the baseline, but with added normalisation noise. We therefore expect weighting to become more effective at larger batch sizes.

Why the full method works for DETR. The two components are complementary. $\mathcal{L}_{\text{cons}}$ provides the disagreement signal between raw and enhanced views, while the weighting component uses that signal to down-weight unreliable supervision. Without $\mathcal{L}_{\text{cons}}$, the weighting mechanism has no stable uncertainty signal; without weighting, consistency regularization treats all samples equally, including noisy ones.

YOLO ablation analysis. YOLO behaves differently from DETR. On DUO→RUOD, the best result comes from `weight only`, while the full method drops below the baseline. On RUOD→DUO, however, the full method gives the best result, improving over the baseline on both AP@0.5 and mAP. This suggests that YOLO already benefits from the weighting signal, while the added consistency term becomes useful mainly in the harder RUOD→DUO direction rather than uniformly across settings.

RQ3 answer: *Consistency training with noise-aware weighting can improve cross-domain robustness without re-annotation, but the effect is detector- and direction-dependent. For DETR, the full method helps on DUO→RUOD but not on RUOD→DUO. For YOLO, noise-aware weighting is strongest on DUO→RUOD, while the full combination is best on RUOD→DUO.*

4.5. Qualitative Results and Label Noise Audit

Figure 4 shows representative detection outputs across four conditions: clean in-domain (Row 1), severe haze corruption (Row 2), cross-domain transfer (Row 3), and a GT mislabel example (Row 4).

Rows 1–2: In-domain and corruption. In-domain, DETR successfully localises dense echinus clusters with confidence scores near 1.0, consistent with its in-domain AP@0.5 of 0.694. Under severity-5 haze (Row 2), the model produces almost no detections (mAP: 0.053, −88%), confirming that the scattering veil pushes the colour distri-

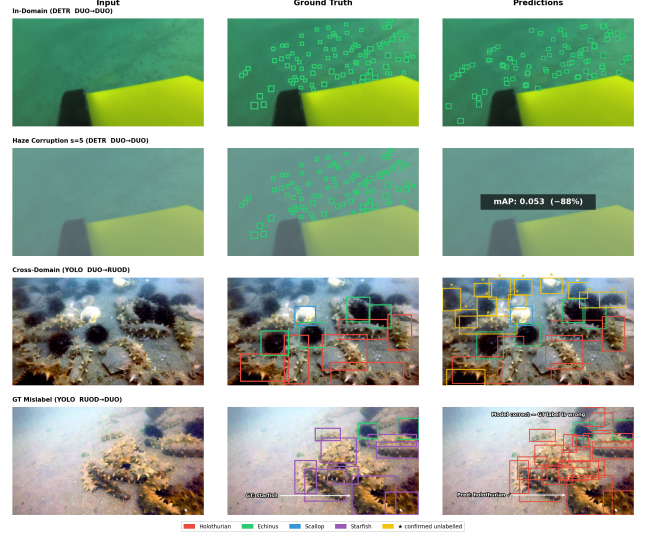


Figure 4. Qualitative detection results across four conditions. **Row 1** (In-domain, DETR DUO→DUO): dense echinus clusters correctly localised at high confidence. **Row 2** (Haze $s=5$, DETR DUO→DUO): model nearly fails entirely (mAP 0.053, −88%), confirming haze as the most catastrophic corruption. **Row 3** (Cross-domain, YOLO `weight_only` DUO→RUOD): yellow ★ boxes mark high-confidence detections with no ground-truth annotation — real objects missed by the human annotator and incorrectly penalised as false positives by COCO evaluation. **Row 4** (GT mismatch, YOLO RUOD→DUO): objects visually identifiable as holothurians are labelled *starfish* in the ground truth; the model predicts the correct class but is doubly penalised. Best viewed in colour.

bution entirely outside the training domain.

Row 3: Cross-domain transfer and annotation quality. When the YOLO `weight_only` variant (DUO→RUOD) is applied to image 005935.jpg, the model produces several high-confidence detections with no corresponding ground-truth box (shown in yellow ★ in Figure 4). Figure 5 shows a zoomed view of this panel.

Visual inspection confirms these are real objects missed by the annotator, not model errors — a direct instance of the annotation-quality confounder described by Lucas et al. [5]. Under COCO evaluation every yellow box is counted as a false positive, artificially depressing reported mAP.

Row 4: GT mislabel. Image 005286.jpg shows a more severe annotation failure: objects visually identifiable as holothurians are labelled as *starfish* in the ground truth. The YOLO model (RUOD→DUO) correctly predicts *holothurian* at these locations, yet COCO penalises each prediction as both a false positive (wrong class) and a false negative (GT box unmatched) — the model is doubly penalised for being right.

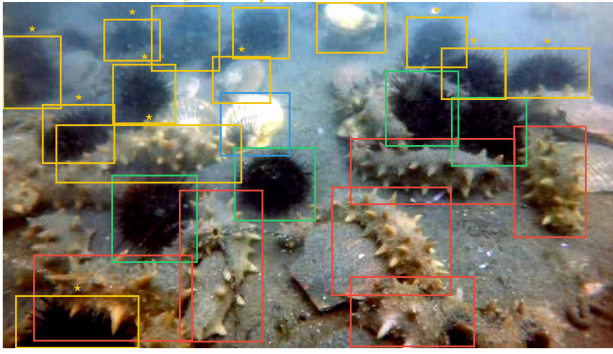


Figure 5. **Zoomed qualitative example of missing ground-truth annotations.** Cross-domain YOLO `weight_only` predictions on RUOD (trained on DUO; image 005935.jpg). Yellow $\star$ markers follow the convention of Fig. 4. Under standard COCO evaluation, these detections are counted as false positives.

Label noise audit. To quantify these effects systematically, we ranked all RUOD test images by number of high-confidence unmatched predictions (score ≥ 0.3 , IoU < 0.1 with all same-class GT boxes) and manually reviewed the top 80 images across two reviewers, with a third completing a further 40 for a total of 120 images. Each unmatched prediction was classified as CONFIRMED_UNLABELLED (real object, no GT box), GT_MISLABEL (GT box with wrong class), CLASS_CONFUSION (correct localisation, class mismatch from domain shift), FALSE_POSITIVE, or AMBIGUOUS.

Across 1212 logged predictions, reviewers confirmed 853 unlabelled objects, 13 GT mislabels, and 28 class confusions — 881 predictions (confirmed unlabelled + class confusions) penalised by COCO that the model localised correctly. Table 4 reports per-class missing-annotation rates. In the full RUOD test set rates are low (0.4–1.6%); in the audited high-noise subset they rise to 25–44%, confirming that annotation errors are *concentrated* in specific images rather than uniformly distributed.

Table 4. Estimated label-noise rates by class for the full dataset and the 120-image audited subset.

Class	Full		Subset	
	GT	Miss. %	GT	Miss. %
Holothurian	15,480	0.43	268	25.0
Echinus	63,584	0.70	1,009	44.1
Scallop	14,704	1.63	540	44.3
Starfish	19,400	0.53	339	30.1

The concentration effect is also visible in adjusted mAP. For the YOLO baseline on DUO→RUOD, correcting the 157 confirmed missing annotations on the 21 most af-

fected images raises mAP from 0.430 to 0.516 (+0.086) on that subset alone, versus only +0.011 on the full test set (0.358→0.369). In-domain, the same correction yields a negligible Δ mAP of < 0.001 , confirming that label noise is a *cross-domain* problem: annotators missed objects that are harder to see in RUOD’s varied lighting conditions but that a model trained on DUO’s cleaner images can nonetheless detect. These results motivate re-annotating the RUOD test set using enhanced images as a reference, which would give a cleaner upper bound on true cross-domain performance.

5. Conclusion

In conclusion, this study answers the three research questions. First, YOLO achieves higher absolute performance than DETR in both in-domain and cross-domain underwater detection. Relative robustness, however, is direction-dependent: DETR retains more of its in-domain mAP under DUO→RUOD shift, whereas YOLO retains more under RUOD→DUO. Second, corruption experiments show that haze and turbidity are the most damaging degradations, while blur is the least harmful and can even slightly improve YOLO at low severities. Third, the ablation results show that consistency training with noise-aware weighting is beneficial, but not uniformly across models and directions: the full method improves DETR on DUO→RUOD, whereas for YOLO the best variant depends on transfer direction. Finally, the qualitative audit shows that annotation noise materially suppresses reported cross-domain performance, with missing labels concentrated in a relatively small subset of images.

This study is limited by time constraints, as DETR was trained for only 30 epochs and may require longer training to fully converge. In future work, training DETR for more epochs, evaluating stronger transformer variants such as RT-DETR, incorporating physics-informed enhancement models, applying domain-specific augmentation to reduce CNN feature bias, and expanding evaluation to more shared classes and larger batch sizes may further improve robustness.

6. AI Use Acknowledgement

AI was used for guidance in the project, debugging errors in the code, as well as proofreading and revising report and code.

The following AI tools were used:

- **Claude, Chatgpt:** for code help/assistance and debugging errors
- **Grammarly, Claude:** for report writing help/assistance and revision

Prompts used:

- Why am I getting this error?
- Help me solve this error
- Is this section written correctly?
- Without modifying the code, add in comments for documentation/explanation.
- Make this chart(YOLO vs DETR corruption curves) match the colors in the slides
- Do not remove what is currently written in the README.md file. Instead, add proper headers and organize it in a way that is helpful for those who are curious about the project.
- Help me finalize a results visualization notebook using the `src/utlis/*.py` scripts I wrote. Add any missing plots or tables, but confirm with me before modifying existing functions.
- Help me build a label noise audit notebook with 4 sections based on my design: (1) auto-scan evaluation results for high-confidence unmatched predictions ($score \geq 0.30$, $IoU < 0.10$), (2) split flagged images across 3 reviewers, (3) an interactive ipywidgets review widget for per-box verdict labelling, and (4) merge findings and compute adjusted mAP and label noise rates.

7. Individual Contributions

Buthaina AlMulla led the project end-to-end. She identified both datasets, discovered the cross-dataset image overlap, and built the pHash deduplication pipeline (RUOD-filtered.v2). She architected the experiment codebase, wrote all notebook drafts (01–08), and built the shared results pipeline used by all members. She ran the YOLO method ablations on both directions, identified and fixed the YOLO ablation training setup (notebook 06), and ran the corrected experiments. She identified the label noise problem during qualitative inspection, designed and ran the audit (notebook 08), and computed the adjusted mAP metrics. She wrote the Method, Experiments, and Conclusion sections, managed the repository and documentation, merged all results, and coordinated the team throughout.

Abdullah Alshehhi proposed the project idea and the dual-view consistency training methodology. He designed and implemented the DETR training pipeline, resolving convergence issues that had blocked training, and ran all DETR experiments: in-domain baselines, cross-domain evaluations, method ablations, and corruption curves. He wrote the initial draft of the experimental setup and results subsections and participated in the label noise audit.

Shamma Yaqoob created the initial YOLO ablation notebook, trained YOLO on DUO, and ran the in-domain and DUO→RUOD cross-domain evaluations, providing a first

set of results that informed the final experimental design. She ran the YOLO corruption evaluation (all four corruptions, five severities) and merged her results with Abdullah’s DETR curves to produce comparative corruption figures. She wrote the Related Work section, refined the conclusion section, contributed to report writing and proofreading, designed and presented the final slides, and participated in the label noise audit.

References

- [1] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European Conference on Computer Vision (ECCV)*, 2020. 4
- [2] Chenping Fu, Risheng Liu, Xin Fan, Puyang Chen, Hao Fu, Wanqi Yuan, Ming Zhu, and Zhongxuan Luo. Rethinking general underwater object detection: Datasets, challenges, and solutions. *Neurocomputing*, 517:243–256, 2023. 4
- [3] M. Jian, N. Yang, C. Tao, H. Zhi, and H. Luo. Underwater object detection and datasets: A survey. *Intelligent Marine Technology and Systems*, 2024. 1, 2
- [4] C. Liu et al. A dataset and benchmark of underwater object detection for robot picking. *arXiv preprint arXiv:2106.05681*, 2021. 1, 3
- [5] A. Lucas et al. Underwater image enhancement and object detection: Are poor object detection benchmarks. In *Proceedings of the WACV 2025 Workshops (MaCVi)*, 2025. 1, 2, 7
- [6] Ultralytics. YOLOv8 documentation. <https://docs.ultralytics.com>, 2023. 2
- [7] F. Fan Y. Wu X. Wang, K. Li and Y. Zhang. Optimized yolo8s framework with deformable convolution for underwater object detection. *Scientific Reports*, vol. 15, no. 1, 2025. 2
- [8] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. In *International Conference on Learning Representations (ICLR)*, 2021. 2
- [9] Karel Zuiderveld. Contrast limited adaptive histogram equalization. In *Graphics Gems IV*, pages 474–485. Academic Press, 1994. 2