Delving into Cascaded Instability: A Lipschitz Continuity View on Image Restoration and Object Detection Synergy

Qing Zhao[1] **Weijian Deng**[2] **Pengxu Wei**[1,5†]
Ziyi Dong[1] **Hannan Lu**[3] **Xiangyang Ji**[4] **Lin Liang**[1,5]
[1]Sun Yat-sen University [2]Australian National University
[3]Harbin Institute of Technology [4]Tsinghua University [5]Peng Cheng Laboratory
{zhaoq78, dongzy6}@mail2.sysu.edu.cn, dengwj16@gmail.com,
weipx3@mail.sysu.edu.cn, luhannan@hit.edu.cn, xyji@tsinghua.edu.cn, linliang@ieee.org

Abstract

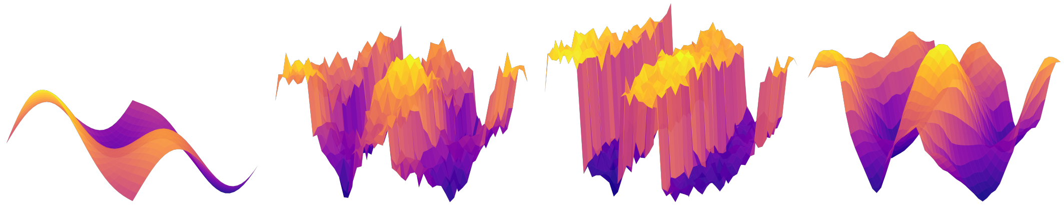
To improve detection robustness in adverse conditions (e.g., haze and low light), image restoration is commonly applied as a pre-processing step to enhance image quality for the detector. However, the functional mismatch between restoration and detection networks can introduce instability and hinder effective integration—an issue that remains underexplored. We revisit this limitation through the lens of Lipschitz continuity, analyzing the functional differences between restoration and detection networks in both the input space and the parameter space. Our analysis shows that restoration networks perform smooth, continuous transformations, while object detectors operate with discontinuous decision boundaries, making them highly sensitive to minor perturbations. This mismatch introduces instability in traditional cascade frameworks, where even imperceptible noise from restoration is amplified during detection, disrupting gradient flow and hindering optimization. To address this, we propose Lipschitz-regularized object detection (LROD), a simple yet effective framework that integrates image restoration directly into the detector's feature learning, harmonizing the Lipschitz continuity of both tasks during training. We implement this framework as Lipschitz-regularized YOLO (LR-YOLO), extending seamlessly to existing YOLO detectors. Extensive experiments on haze and low-light benchmarks demonstrate that LR-YOLO consistently improves detection stability, optimization smoothness, and overall accuracy.

1 Introduction

Adverse imaging conditions introduce challenges for object detection by causing various image degradations, including reduced contrast, blurred edges, and obscured object boundaries. A typical way to alleviate this issue is to employ image restoration as a pre-processing step, aiming to improve image quality before detection [1, 2, 3]. However, its effectiveness is limited by the functional mismatch between restoration and detection networks. This inconsistency can introduce instability, where imperceptible noise introduced during restoration is amplified during detection, leading to unreliable predictions [4, 5]. Moreover, the underlying differences between these tasks remain underexplored, hindering opportunities for better integration and enhanced robustness. To bridge this gap, understanding their functional behaviors is crucial for achieving effective synergy. To this end, we analyze the conventional Image Restoration→Object Detection cascade framework through the lens of Lipschitz continuity, focusing on two aspects: the input space and the parameter space.

†Corresponding Author. Code available: `https://github.com/diasuki/LR-YOLO`

39th Conference on Neural Information Processing Systems (NeurIPS 2025).



(a) Image Restoration (b) Object Detection (c) Cascade (d) Ours

Figure 1: **Visualization of network functional behaviors under input perturbations.** (a) Image Restoration networks exhibit smooth, continuous mappings, where input changes lead to gradual adjustments. (b) Object Detection networks display sharp discontinuities due to abrupt decision boundaries in classification and bounding box regression. (c) Cascade frameworks (Image Restoration $\rightarrow$ Object Detection) magnify instability, resulting in fragmented and non-smooth behavior. (d) Our method integrates low-Lipschitz image restoration into the feature learning of high-Lipschitz object detection, promoting smoother transitions and enhanced stability.

From the input space perspective, we leverage the concept of Lipschitz continuity, which characterizes the sensitivity of a model's output to input perturbations [6]. Networks with lower Lipschitz constants exhibit smoother, more predictable changes, while higher constants indicate heightened sensitivity and instability. By computing the Jacobian norm [7] with respect to haze density variations, we observe that the Lipschitz constant of the object detection network is nearly an order of magnitude larger than that of the restoration network, highlighting its substantially lower smoothness. This disparity highlights the differences in their functional behaviors. Restoration networks exhibit smooth, continuous mappings, where small input perturbations result in gradual and predictable adjustments to the restored image. This smoothness stems from pixel-wise processing that consistently enhances local regions and propagates changes smoothly across the image. In contrast, object detection networks are inherently discontinuous, characterized by sharp decision boundaries in classification and bounding box regression. Even minor pixel-level changes can cause abrupt shifts in class predictions or bounding box coordinates, reflecting non-smooth, step-like transitions in the output. This sharp contrast in behavior contributes to instability when the two networks are cascaded. To further illustrate this disparity, we visualize the functional behaviors in Figure 1 (a) and (b), where the smooth transitions of restoration sharply contrast with the abrupt shifts observed in detection. This inconsistency introduces instability when the two networks are cascaded: imperceptible noise introduced during restoration can be amplified during detection, resulting in overall non-smooth behavior in the cascade framework, as shown in Figure 1 (c).

To further understand the instability observed in conventional Image Restoration$\rightarrow$Object Detection cascade framework, we extend our analysis to the parameter space of the networks, where Lipschitz continuity characterizes the sensitivity of a model's output to changes in its parameters. Our findings reveal that image restoration networks maintain relatively low Lipschitz constants, resulting in smooth and stable optimization trajectories during training. In contrast, object detection networks exhibit substantially higher Lipschitz constants, leading to sharp gradient transitions and erratic convergence paths. This imbalance disrupts gradient flow, introduces mutual interference, and destabilizes joint optimization, further compounding the instability of traditional cascade frameworks.

Given the importance of network stability in adverse conditions, a key challenge lies in harmonizing image restoration and object detection to address the inherent differences in Lipschitz continuity. To address this, we propose Lipschitz-regularized object detection (LROD), a simple yet effective framework that integrates image restoration directly into the detector's feature learning. Unlike conventional cascades, LROD harmonizes the Lipschitz continuity of both tasks during training, smoothing out perturbations before they propagate through the detector's discontinuous layers. This coupling mitigates noise amplification, enhancing stability in challenging environments. Furthermore, LROD introduces a parameter-space regularization term to stabilize gradient flows, ensuring smoother optimization dynamics and improved robustness under varying degradation intensities.

We implement LROD into existing YOLO detectors, taking advantage of their real-time performance, resource efficiency, and suitability for edge deployment. This integration yields an efficient model, called Lipschitz-regularized YOLO (LR-YOLO), which can be seamlessly applied to YOLO series detectors (*e.g.*, YOLOv10 [8] and YOLOv8 [9]). As shown in Figure 1 (d), our Lipschitz-regularized object detection achieves a smoother Lipschitz continuity compared to the cascade framework. Extensive experiments on image dehazing and low-light enhancement benchmarks demonstrate that LR-YOLO improves detection stability and robustness compared to traditional cascade frameworks. In summary, our contributions are as follows:

- **Lipschitz Continuity Analysis**: we perform a detailed analysis of Lipschitz continuity in both the input space and the parameter space of image restoration and object detection networks. Our analysis uncovers a critical mismatch in smoothness between these tasks, which potentially introduces instability and impedes effective integration. To our knowledge, this is the early work to provide a detailed Lipschitz continuity analysis aimed at understanding the instability challenges in cascade-based detection pipelines.
- **Lipschitz-Regularized Framework**: motivated by our analysis, we propose a simple and effective object detection framework that integrates image restoration directly into the detector's feature learning, harmonizing the Lipschitz continuity of both tasks during training. This design enhances smoothness and mitigates the instability inherent in traditional cascade-based methods.

2 Related Work

Object Detection Under Adverse Conditions. Existing research primarily focuses on cascade frameworks, where image restoration techniques such as image dehazing [10, 11], low-light enhancement [12, 13], and all-in-one restoration [14] are used as pre-processing steps to improve image quality and enhance human trust in detection results compared to domain adaptation approaches [15]. ReForDe [2] uses adversarial training to generate detection-friendly labels for fine-tuning restoration networks. SR4IR [16] introduces a training framework where image restoration is constrained by object detection, and detection training utilizes restoration outputs Image-adaptive techniques [1, 3] integrate differentiable image processing filters into the detection pipeline. FeatEnHancer [17] applies hierarchical feature enhancement to improve detection performance. Despite these advancements, the functional mismatch between restoration and detection networks is underexplored. Our work reports that the large disparity in Lipschitz continuity between the two tasks exacerbates non-smoothness when they are cascaded, leading to instability under varying degradation intensities. To address this, we propose a Lipschitz-regularized framework that enhances the Lipschitz continuity of the detection network, facilitating better harmonization between these two tasks.

Lipschitz Continuity Analysis. Lipschitz continuity is useful in analyzing the stability and robustness of deep neural networks [18, 19, 20]. Models with lower Lipschitz constants tend to exhibit better generalization performance, especially under adversarial conditions [21]. This has motivated further research on regularization techniques that constrain the Lipschitz constant to enhance model robustness. For instance, SN-GAN [22] controls the Lipschitz constant by restricting the spectral norm of network parameters, while other Lipschitz-based regularization techniques have been proposed to improve model stability [23]. Several studies have extended these ideas to network design [24], highlighting the critical role of Lipschitz continuity in controlling the smoothness and stability of neural networks. In our work, we analyze object detection stability under adverse conditions from both the input and parameter spaces using Lipschitz continuity as the lens of investigation. We demonstrate that the disparity in Lipschitz continuity between image restoration and object detection networks is a primary source of non-smoothness and instability in cascade frameworks.

3 Lipschitz Continuity Perspective

3.1 Input Space Analysis: Model Stability in Adverse Conditions

Object detection in adverse conditions, such as haze or low light, is highly sensitive to variations in degradation intensity, including changes in haze density and luminance fluctuations. Traditional Image Restoration$\rightarrow$Object Detection cascade framework struggles with such variations, leading to unstable detection results. As shown in Figure 2 (a), even when partially mitigated by restoration, minor perturbations still cause significant shifts in detector features, exposing the framework's instability. To understand this, we analyze the problem through the lens of Lipschitz continuity, which quantifies a model's sensitivity to input changes. Our findings reveal that the Lipschitz constant of

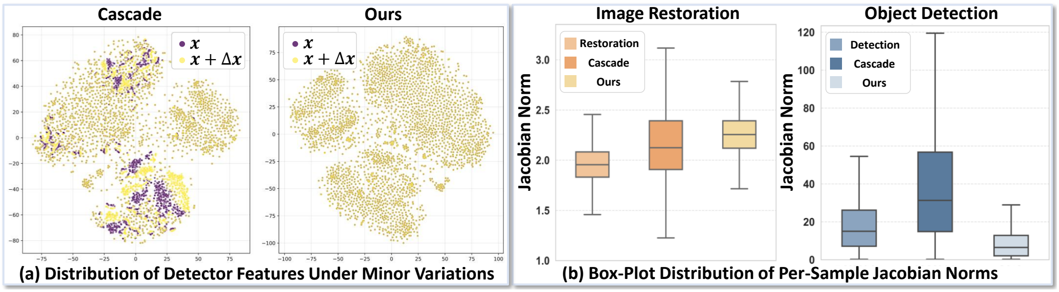


Figure 2: **Impact of haze density variations on feature stability and Lipschitz continuity.** (a) Distribution of the detector backbone's features between two haze inputs $\boldsymbol{x}$ and $\boldsymbol{x} + \Delta\boldsymbol{x}$ under minor haze density variations $\Delta\boldsymbol{x}$ on Pascal VOC [26] with synthetic haze. In the cascade framework, nearly half of the features shift under slight haze density variations, while our Lipschitz-regularized object detection remains stable. (b) Box-plot distribution of Jacobian norms $\|\nabla_{\boldsymbol{x}} f_\theta(\boldsymbol{x})\|$ at each sample $\boldsymbol{x}$ between image restoration and object detection task on Pascal VOC [26] with synthetic haze. The Lipschitz constant of the object detection network is nearly an order of magnitude larger than that of the restoration network. This large disparity in Lipschitz continuity between the two tasks exacerbates the non-smoothness in the cascade framework. Our method constrains the Lipschitz constant of object detection to harmonize these two tasks better. ConvIR [11] and YOLOv8 [9] are taken as restoration and detection methods, respectively.

the detection network is nearly an order of magnitude larger than that of the restoration network, amplifying noise and disrupting stability under adverse conditions.

We begin by recalling the definition of Lipschitz continuity: A network $f(\cdot;\theta) : \mathbb{R}^D \mapsto \mathbb{R}^K$, defined on some domain $dom(f) \subseteq \mathbb{R}^D$ with parameters θ, is called C-Lipschitz continuous if there exists a real constant $C > 0$ such that $\forall \boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathrm{dom}(f) : \|f(\boldsymbol{x}_1;\theta) - f(\boldsymbol{x}_2;\theta)\|_p \leq C\|\boldsymbol{x}_1 - \boldsymbol{x}_2\|_p$. For simplicity, we will compute the 2-norm, denoted as $\|\cdot\|$, throughout the rest of the paper, which can be easily generalized to other norms. Using Theorem 1 in [25], we know that for a differentiable, C-Lipschitz continuous network $f(\cdot;\theta) : \mathbb{R}^D \mapsto \mathbb{R}^K$, the Lipschitz constant of $f(\cdot;\theta)$ can be expressed as $C_{\boldsymbol{x}}(f(\boldsymbol{x};\theta)) = \sup_{\boldsymbol{x}\in dom(f)} \|\nabla_{\boldsymbol{x}} f(\boldsymbol{x};\theta)\|_* = \sup_{\boldsymbol{x}\in dom(f)} \|\nabla_{\boldsymbol{x}} f(\boldsymbol{x};\theta)\|$, where $\nabla_{\boldsymbol{x}} f(\boldsymbol{x};\theta)$ is Jacobian of f w.r.t. input $\boldsymbol{x}$ and $\|\cdot\|_*$ denotes the dual norm (The dual norm of the 2-norm is itself).

To quantitatively assess the Lipschitz constant of the image restoration and object detection network, we compute the above Jacobian norm for each sample $\boldsymbol{x}$ in the Pascal VOC dataset [26], considering variations in haze density. As shown in Figure 2 (b), we observe that the Jacobian norm of the image restoration network ranges from 1 to 3.5 per sample, while the Jacobian norm of the object detection network is nearly an order of magnitude larger than that of the image restoration network. This indicates that object detection has a higher Lipschitz constant compared to image restoration. Therefore, the large disparity in Lipschitz continuity between the two tasks leads to an unstable framework when they are directly cascaded. Specifically, even slight variations will inevitably be amplified by the restoration network since its Jacobian norm per sample exceeds 1, and further destabilized by the high-Lipschitz constant of the detection network within the cascade framework.

Remark 1. *Image restoration networks exhibit smooth and continuous mappings, while object detection networks are more non-smooth from the perspective of Lipschitz continuity. This large disparity in Lipschitz continuity between the two tasks exacerbates the non-smoothness when they are directly cascaded, leading to instability under variations in degradation intensities.*

3.2 Parameter Space Analysis: Training Stability

The disparity in Lipschitz continuity between restoration and detection networks extends beyond the input space, impacting their training stability. To understand this, we analyze the parameter space of the networks to capture how gradient updates influence model stability during optimization. Our analysis shows that restoration networks, with lower Lipschitz constants, maintain smooth optimization trajectories, while detection networks, with substantially higher Lipschitz constants, experience sharp gradient transitions and unstable convergence. This imbalance disrupts gradient flow, contributing to training instability in cascade-based designs.

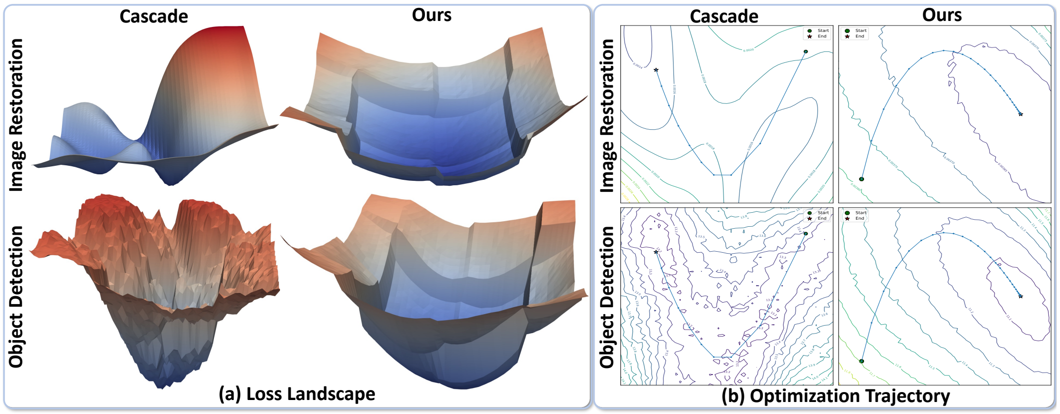


Figure 3: **Parameter-space smoothness and optimization stability comparison between the cascade framework and our Lipschitz-regularized object detection.** (a) Loss landscapes of restoration and detection tasks: restoration networks demonstrate smooth trajectories, while detection networks encounter sharp gradient transitions, indicating instability. (b) The cascade framework amplifies this imbalance, leading to inefficient convergence and oscillatory optimization paths. (c) Our method introduces Lipschitz regularization to smooth the parameter space of object detection, enhancing stability and harmonizing it with restoration. ConvIR [11] and YOLOv8 [9] are used as representative restoration and detection methods, respectively.

We extend the Lipschitz continuity analysis to the parameter space: A network $f(\boldsymbol{x};\theta)$ defined on some parameter space $\boldsymbol{\Theta}$ is called Lipschitz continuous in the parameter space if there exists $C_\theta(f(\boldsymbol{x};\theta)) > 0$ such that $\forall\theta_1, \theta_2 \in \boldsymbol{\Theta}$, $\|f(\boldsymbol{x};\theta_1) - f(\boldsymbol{x};\theta_2)\| \leq C_\theta(f(\boldsymbol{x};\theta))\|\theta_1 - \theta_2\|$. Due to the symmetry between $\boldsymbol{x}$ and θ, an analogous result holds when the two variables are interchanged: The Lipschitz constant in the parameter space of $f(\boldsymbol{x};\theta)$, defined on the parameter space $\boldsymbol{\Theta}$, can be expressed as $C_\theta(f(\boldsymbol{x};\theta)) = \sup_{\theta\in\boldsymbol{\Theta}} \|\nabla_\theta f(\boldsymbol{x};\theta)\|$, where $\nabla_\theta f(\boldsymbol{x};\theta)$ represents the gradient of network parameters in the parameter space.

Given that the network is trained using the gradient descent optimization algorithm, expressed as $\theta \leftarrow \theta - \mu \cdot \nabla_\theta f(\boldsymbol{x};\theta)$ (μ denotes the learning rate), the Lipschitz constant in the parameter space is crucial for ensuring training stability. This is because the Lipschitz constant in the parameter space acts as an upper bound for the gradients of the network parameters during training.

The Lipschitz continuity in parameter space reflects the sensitivity of the model's output to variations in its parameters. To illustrate this, we visualize the loss landscape by perturbing parameters along two directions, revealing their impact on optimization smoothness and stability. Specifically, we use the visualization method in [27]: let θ represent the fixed model parameters, we select two normalized direction vectors δ and η in the parameter space, and plot the function $f(\alpha, \beta) = \mathcal{L}(\theta + \alpha\delta + \beta\eta)$ on the surface, where $\mathcal{L}$ is the loss function, and α and β are the coordinates on the surface.

As shown in Figure 3 (a), the loss landscape of the image restoration network exhibits a smooth loss function, while the loss landscape of the object detection network is notably rough. This reflects differences in their Lipschitz constants in parameter space: restoration networks tend to have lower Lipschitz constants, giving smoother gradients, while detection networks exhibit higher Lipschitz constants, showing sharper transitions, and increased sensitivity to parameter changes. Figure 3 (b) further illustrates the optimization trajectories, where restoration follows stable paths, while detection experiences frequent shifts, indicating instability. This imbalance disrupts gradient flow during joint training, resulting in unstable convergence and reduced optimization efficiency.

Remark 2. *Image restoration networks with lower Lipschitz constants exhibit smooth optimization trajectories, while object detection networks with higher Lipschitz constants experience sharp gradient transitions and unstable convergence. This imbalance in the parameter space between these two tasks results in training instability and reduced optimization efficiency in cascade-based designs.*

4 Lipschitz-Regularized Object Detection

The analysis in Section 3 reveals the disparities in Lipschitz continuity between restoration and detection networks, manifesting in both the input space and the parameter space. Driven by this, we propose lipschitz-regularized object detection (LROD), a simple and effective framework that harmonizes restoration and detection through targeted Lipschitz regularization. Specifically, LCOD introduces two core mechanisms: *1) Lipschitz regularization via low-Lipschitz restoration* to constrain the Lipschitz constant of object detection in the input space, and *2) Lipschitz regularization via parameter-space smoothing* to constrain the Lipschitz constant in the parameter space.

4.1 Lipschitz Regularization via Low-Lipschitz Restoration

Lipschitz continuity analysis in input space (Section 3.1) shows that image restoration networks exhibit smooth, continuous mappings, while object detection networks are more non-smooth. By leveraging the low-Lipschitz properties of the restoration task, we integrate restoration learning into the detector backbone's feature learning, constraining the Lipschitz constant of the object detection task in the input space. This better harmonizes the detection task with the low-Lipschitz restoration.

Remark 3 (Lipschitz Regularization via Low-Lipschitz Restoration)**.** *Let: $f_{\theta_b,\theta_d} = f_{\theta_d} \circ f_{\theta_b}$ denote the object detection model, where $f_{\theta_b}(\cdot;\theta_b)$ is the backbone network parameterized by θ_b, and $f_{\theta_d}(\cdot;\theta_d)$ is the detection head parameterized by θ_d. Similarly, let: $g_{\theta_b,\theta_r} = f_{\theta_r} \circ f_{\theta_b}$ represent the image restoration model, where $f_{\theta_r}(\cdot;\theta_r)$ is the restoration head parameterized by θ_r, sharing the same backbone f_{θ_b}. Given a weighted combination of the detection loss and the restoration loss:*

$$\mathcal{L}(\theta_b, \theta_d, \theta_r) = \mathcal{L}_{\text{det}}(f_{\theta_b,\theta_d}) + \lambda \cdot \mathcal{L}_{\text{res}}(g_{\theta_b,\theta_r}), \quad \lambda > 0$$

Let $\text{Lip}(f_{\theta_b}) := \sup_{\boldsymbol{x}} \|J_{f_{\theta_b}}(\boldsymbol{x})\|$ be the Lipschitz constant of f_{θ_b} defined by jacobian norm. If:

1. *$\mathcal{L}_{\text{res}}$ is Lipschitz continuous and $\|\nabla_{\theta_b}\mathcal{L}_{\text{res}}(g_{\theta_b,\theta_r})\| \leq G$ for $G < \|\nabla_{\theta_b}\mathcal{L}_{\text{det}}(f_{\theta_b,\theta_d})\|$;*
2. *There exists a training sample $\boldsymbol{x}^\star$ and $\gamma > 0$ such that: $\left\langle \nabla_{\theta_b}\|J_{f_{\theta_b}}(\boldsymbol{x}^\star)\|, \nabla_{\theta_b}\mathcal{L}_{\text{res}}(g_{\theta_b,\theta_r}) \right\rangle \geq \gamma$,*

then under continuous-time gradient descent ${\theta_b}^{(t+1)} \leftarrow {\theta_b}^{(t)} - \mu \cdot \nabla_{\theta_b}\mathcal{L}(\theta_b, \theta_d, \theta_r)$ (μ denotes the learning rate), the evolution of the Lipschitz constant satisfies:

$$\frac{d}{dt}\left[\text{Lip}(f_{\theta_b})\right] \leq -\lambda \cdot \gamma + \xi(t)$$

where $\xi(t) := \left\langle \nabla_{\theta_b}\|J_{f_{\theta_b}}(\boldsymbol{x}^\star)\|, \nabla_{\theta_b}\mathcal{L}_{\text{det}}(f_{\theta_b,\theta_d}) \right\rangle$ is the unconstrained change induced by the detection loss and γ is the regularization via the restoration task.

This suggests that integrating the image restoration task directly into the detector's feature learning by sharing the detector's backbone helps suppress the model's sensitivity to input perturbations during training, effectively acting as a Lipschitz regularization. The detailed proof is in Appendix A.

Specifically, we extract low-level features from the first three stages of the detector backbone, which preserve essential spatial and textural information for image restoration. These features are then passed through a restoration-specific head to obtain the restored images. By leveraging the inherently smoother Lipschitz continuity of the image restoration task, this restoration loss implicitly regularizes the feature representations used for object detection during training, thereby constraining the Lipschitz constant of the detection network in the input space. As shown in Figure 2 (a) and (b), our Lipschitz-regularized object detection exhibits smoother Lipschitz continuity compared to both the original object detection and the cascade framework, with lower Lipschitz constants and more stable detector features under varying degradation intensities.

4.2 Lipschitz Regularization via Parameter-Space Smoothing

Lipschitz continuity analysis in parameter space (Section 3.2) shows that low-Lipschitz restoration networks maintain smooth optimization trajectories, while high-Lipschitz detection networks experience sharp gradient transitions and unstable convergence. To improve harmony between these tasks and ensure training stability, we constrain the Lipschitz constant of the detection networks in the parameter space. We introduce a parameter-space regularization term to stabilize gradient flows, promoting smoother optimization dynamics.

Remark 4 (Lipschitz Regularization via Parameter-Space Smoothing)**.** *Let $\theta = \theta_b \cup \theta_d$ is the full parameter set of the detection model. The parameter-space regularization term is defined as the gradient norm with respect to the model parameters, denoted by $\|\nabla_\theta f_\theta(\boldsymbol{x})\|$.*

Full framework. The Lipschitz-regularized object detection (LROD) framework incorporates the above two regularizations to ensure stable and efficient training. The total loss function is defined as:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{det}} + \lambda \cdot \mathcal{L}_{\text{res}} + \lambda_p \cdot \|\nabla_\theta f_\theta(\boldsymbol{x})\|,$$

where $\mathcal{L}_{\text{det}}$ is the detection loss, $\mathcal{L}_{\text{res}}$ is the restoration loss, computed as a Charbonnier loss [28] between the restored image and the ground truth clean image, and $\|\nabla_\theta f_\theta(\boldsymbol{x})\|$ is the regularization term. The weights λ and λ_p are used to balance the restoration and regularization terms, respectively.

We implement this framework as Lipschitz-regularized YOLO (LR-YOLO), which builds upon YOLO detectors. As illustrated in Figure 3, ConvIR [11] and YOLOv8 [9] are employed as representative restoration and detection methods, respectively. LR-YOLO smooths the loss landscape of object detection compared to traditional cascade frameworks, better aligning with image restoration during training. This results in smooth gradient flow, improved stability, and more efficient optimization.

5 Experiments

5.1 Experimental Settings

Dataset. Datasets cover two challenging conditions: *hazy weather* and *low-light environments*. For both settings, we use Pascal VOC [26] and COCO [29] datasets for training and validation following the degradation setting from [1, 2, 5], and real-world datasets for out-of-domain evaluation.

1) Training and Validation Data: a) VOC_Haze_Train and *VOC_Haze_Val* consist of $8,111$ and $2,734$ images respectively. Haze is synthesized *online* during training using the atmospheric scattering model with $\beta \in [0.5, 1.5]$, while validation images are synthesized *offline* once for reproducibility; *b) VOC_Dark_Train* and *VOC_Dark_Val* consist of $12,334$ and $3,760$ images respectively. Low-light is simulated *online* during training and *offline* for validation via gamma correction with $\gamma \in [1.5, 5]$. The classes in both the training and validation datasets for haze and low-light conditions align with those in the real-world datasets; *c) COCO_Haze_Train* and *COCO_Dark_Train* consist of $118,287$ training images, and *COCO_Haze_Val* and *COCO_Dark_Val* contain $5,000$ validation images.

2) Real-world Test Data. We adopt **two** benchmark datasets for the out-of-domain evaluation: *a) RTTS* [30] contains $4,322$ real-world hazy images annotated with 5 object categories, *i.e.*, *Person*, *Car*, *Bus*, *Bicycle*, and *Motorbike*; *b) ExDark* [31] contains $2,563$ real-world low-light images labeled with 10 categories, *i.e.*, *People*, *Car*, *Bus*, *Bicycle*, *Motorbike*, *Boat*, *Bottle*, *Chair*, *Dog*, and *Cat*.

Evaluation Metrics. We evaluate object detection performance using mean Average Precision (mAP) at an IoU threshold of 50%, which excludes difficult objects by default. Additionally, we report **mAP$_{\text{difficult}}$**, which includes all objects, including challenging cases (*e.g.*, occluded targets) on the Pascal VOC [26] and RTTS [30] datasets. For the COCO dataset [29], we adopt standard COCO-style metrics, including mAP averaged over IoU thresholds from 0.5 to 0.95 (in 0.05 increments), along with AP$_{50}$, AP$_{75}$, and scale-specific scores: AP$_S$ (small), AP$_M$ (medium), and AP$_L$ (large).

Implementation Details: We adopt YOLOv10-s and YOLOv8-s as the baseline detectors. For training, the loss weights are set to $\lambda = 10$ and $\lambda_p = 0.01$. We use the SGD optimizer with an initial learning rate of 1×10^2 and a weight decay of 5×10^{-4}. The model is trained on an RTX 4090 GPU for 100 epochs with a batch size of 16, requiring approximately 8 hours. Input images are resized to 640×640, and standard YOLO data augmentation techniques (*e.g.*, random flipping and affine transformation) are applied. For experiments on the COCO dataset, we use 8 RTX 4090 GPUs with a batch size of 16 per GPU. Training is conducted for 300 epochs and takes approximately 48 hours.

5.2 Object Detection under Adverse Conditions

Table 1 presents a method comparison of object detection under two adverse conditions: *hazy weather*, evaluated on the *VOC_Haze_Val* and *RTTS* [30] datasets, and *low-light environments*, evaluated on the *VOC_Dark_Val* and *ExDark* [31] datasets. We compare various image restoration methods, including SFNet [10], ConvIR [11], LLFormer [12], and RetinexFormer [13], all of which are trained

Methods	*Datasets (Haze Weather)*			
	VOC_Haze_Val		*RTTS* [30]	
	mAP	**mAP$_{\text{difficult}}$**	**mAP**	**mAP$_{\text{difficult}}$**
YOLOv10 [8]	50.5	44.7	42.6	33.8
SFNet [10]→YOLOv10	77.9	70.1	45.5	35.9
SFNet [10]→YOLOv10† [2]	79.1	72.1	46.6	37.1
SFNet [10]→YOLOv10‡ [16]	79.3	71.7	45.8	36.0
ConvIR [11]→YOLOv10	79.9	72.2	46.1	36.0
ConvIR [11]→YOLOv10† [2]	80.1	72.9	46.6	37.2
ConvIR [11]→YOLOv10‡ [16]	80.5	72.6	46.5	36.5
IA [1]→YOLOv10	79.9	72.0	45.4	35.8
GDIP [3]→YOLOv10	79.2	70.9	47.2	37.0
FeatEnHancer [17]→YOLOv10	79.8	71.6	46.7	36.2
LR-YOLOv10 (Ours)	**82.5**	**74.4**	**49.2**	**38.5**
YOLOv8 [9]	54.3	48.3	45.3	36.2
SFNet [10]→YOLOv8	79.2	71.1	48.9	38.4
SFNet [10]→YOLOv8† [2]	80.8	73.8	49.1	39.3
SFNet [10]→YOLOv8‡ [16]	80.3	72.8	49.3	39.2
ConvIR [11]→YOLOv8	80.5	72.8	49.3	38.7
ConvIR [11]→YOLOv8† [2]	80.9	74.1	49.5	39.0
ConvIR [11]→YOLOv8‡ [16]	81.4	74.0	50.1	39.9
IA [1]→YOLOv8	80.6	73.0	47.7	37.3
GDIP [3]→YOLOv8	81.0	73.1	50.3	39.8
FeatEnHancer [17]→YOLOv8	81.2	73.4	48.4	38.8
LR-YOLOv8 (Ours)	**83.3**	**76.5**	**53.2**	**42.4**

Methods	*Datasets (Low-Light Environment)*		
	VOC_Dark_Val		*ExDark* [31]
	mAP	**mAP$_{\text{difficult}}$**	**mAP**
YOLOv10 [8]	62.1	55.0	49.2
LLFormer [12]→YOLOv10	65.6	58.0	46.3
LLFormer [12]→YOLOv10† [2]	64.7	57.5	47.0
LLFormer [12]→YOLOv10‡ [16]	66.3	59.2	49.5
Retinexformer [13]→YOLOv10	66.3	58.6	47.6
Retinexformer [13]→YOLOv10† [2]	66.0	58.4	45.8
Retinexformer [13]→YOLOv10‡ [16]	66.9	59.2	47.5
IA [1]→YOLOv10	66.0	58.7	50.4
GDIP [3]→YOLOv10	65.8	58.5	48.9
FeatEnHancer [17]→YOLOv10	67.6	59.9	50.9
LR-YOLOv10 (Ours)	**70.6**	**62.7**	**53.8**
YOLOv8 [9]	63.4	55.8	50.0
LLFormer [12]→YOLOv8	66.2	58.7	46.6
LLFormer [12]→YOLOv8† [2]	66.2	58.8	47.9
LLFormer [12]→YOLOv8‡ [16]	66.2	59.2	48.6
Retinexformer [13]→YOLOv8	67.8	59.5	47.6
Retinexformer [13]→YOLOv8† [2]	67.7	60.0	49.5
Retinexformer [13]→YOLOv8‡ [16]	68.6	61.0	49.5
IA [1]→YOLOv8	66.5	59.2	49.6
GDIP [3]→YOLOv8	68.9	61.2	51.2
FeatEnHancer [17]→YOLOv8	68.7	60.8	51.8
LR-YOLOv8 (Ours)	**71.7**	**63.9**	**54.5**

Table 1: **Comparison under two adverse conditions: *haze weather* and *low-light environment*.** Left: Results on *VOC_Haze_Val* and *RTTS* [30], with models trained on *VOC_Haze_Train*. Right: Results on *VOC_Dark_Val* and *ExDark* [31], with models trained on *VOC_Dark_Train*. In the cascade framework, † indicates adversarial training [2], and ‡ denotes alternating training [16].

Methods	*COCO_Haze_Val*						*COCO_Dark_Val*					
	mAP	**AP$_{50}$**	**AP$_{75}$**	**AP$_{S}$**	**AP$_{M}$**	**AP$_{L}$**	**mAP**	**AP$_{50}$**	**AP$_{75}$**	**AP$_{S}$**	**AP$_{M}$**	**AP$_{L}$**
YOLOv8 [9]	20.3	28.8	22.0	9.1	22.8	29.1	31.3	45.2	33.5	16.5	34.2	45.1
InstructIR [14]→YOLOv8	33.8	47.7	36.6	15.6	37.2	50.3	31.1	44.8	33.3	15.0	33.7	46.5
InstructIR [14]→YOLOv8† [2]	33.4	47.7	36.1	15.2	36.8	49.0	30.2	43.8	32.3	14.3	32.9	44.5
InstructIR [14]→YOLOv8‡ [16]	35.0	49.7	37.9	17.0	38.9	51.1	30.4	43.8	32.7	15.5	32.9	45.2
IA [1]→YOLOv8	36.4	51.4	39.5	18.0	39.8	51.6	33.3	47.8	36.0	17.7	36.1	48.2
GDIP [3]→YOLOv8	36.6	51.8	39.4	18.1	40.6	51.5	33.4	47.6	36.0	17.5	35.9	47.6
FeatEnHancer [17]→YOLOv8	36.7	52.2	39.7	18.3	40.5	52.1	34.1	49.2	37.1	17.9	37.0	48.5
LR-YOLOv8 (Ours)	**37.7**	53.3	40.6	19.5	41.6	52.7	**35.3**	50.5	37.9	19.0	38.3	49.7

Table 2: **Comparison on *COCO_Haze_Val* and *COCO_Dark_Val* datasets under haze and low-light conditions.** All models are trained from scratch on *COCO_Haze_Train* and *COCO_Dark_Train*, respectively. The † indicates adversarial training [2] and ‡ denotes alternating training [16].

Methods	SFNet [10]	ConvIR [11]	LLFormer [12]	Retinexformer [13]	InstructIR [14]	IA [1]	GDIP [3]	FeatEnHancer [17]	**Ours**
Params (M)	13.27	5.53	24.55	1.61	31.15	0.17	138.24	0.14	0.52
Flops (G)	775.17	42.10	22.52	15.57	123.90	12.32	40.37	44.29	11.32

Table 3: **Computational complexity comparison.** Our method shows lower computational complexity in terms of the number of parameters (Params) and floating point operations (FLOPs).

on degraded images and used to restore inputs before detection. We further consider two joint training strategies: *1) Adversarial training* [2], where restoration networks are fine-tuned to generate detection-friendly images; *2) Alternating training* [16], where restoration is supervised using detection-driven perceptual losses and detection is trained on restored outputs. Furthermore, we include end-to-end methods for comparison, including IA [1], GDIP [3], and FeatEnHancer [17]. They are trained directly on degraded inputs. All models are trained from scratch on the *VOC_Haze_Train* and *VOC_Dark_Train* datasets, respectively. Our method outperforms other methods when using both YOLOv10 and YOLOv8 as object detectors, achieving mAP improvements of 2.0 and 2.9 on *RTTS*, and 2.9 and 2.7 on *ExDark*, respectively. Table 2 shows a comparison on *COCO_Haze_Val* and *COCO_Dark_Val* datasets, trained on *COCO_Haze_Train* and *COCO_Dark_Train*, respectively. We compare the all-in-one restoration method InstructIR [14]. Our method consistently improves performance across all evaluation metrics, achieving mAP improvements of 1.0 and 1.2, respectively.

5.3 Evaluation and Analysis

Computational Complexity Evaluation. Table 3 presents a comparison of parameters (Params) and floating point operations (FLOPs), showcasing the inference efficiency of our framework.

Lipschitz Regularization Ablation Study. We evaluate the impact of two Lipschitz regularization parts (low-Lipschitz restoration learning $\mathcal{L}_{\text{res}}$ and parameter-space smoothing $\|\nabla_\theta f_\theta(\boldsymbol{x})\|$). The evaluation is conducted on the *RTTS* and *ExDark* for out-of-domain performance, as presented in

	$\mathcal{L}_{\text{res}}$	$\|\nabla_\theta f_\theta(\boldsymbol{x})\|$	*RTTS* [30]	*ExDark* [31]
YOLOv10			46.0	50.6
	✓		48.1	52.7
		✓	47.2	51.5
	✓	✓	**49.2**	**53.8**
YOLOv8			49.3	51.6
	✓		51.3	53.6
		✓	50.1	52.4
	✓	✓	**53.2**	**54.5**

Table 4: **Lipschitz regularization ablation study.** We evaluate the effect of two Lipschitz regularization parts $\mathcal{L}_{\text{res}}$ and $\|\nabla_\theta f_\theta(\boldsymbol{x})\|$.

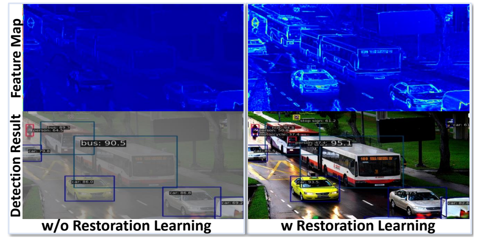


Figure 4: **Feature visualization.** We visualize the feature maps in the backbone of our model trained without and with $\mathcal{L}_{\text{res}}$.

Table 4. Incorporating both restoration learning and parameter space smoothing during training improves synergy between detection and restoration, leading to mAP gains of 3.2 for YOLOv10 and 3.9 for YOLOv8 in *RTTS*, and 3.2 and 2.9 on *ExDark*, respectively, compared to baseline methods.

Restoration Learning Analysis. We visualize the backbone features of our model trained with and without restoration learning $\mathcal{L}_{\text{res}}$ as shown in Figure 4. Integrating restoration learning into the detector's feature learning facilitates the enhancement of degraded image features in the backbone, resulting in improved detection (*e.g.*, complete detections of objects like the stop sign and motorcycle).

Alternative Regularization Ablation Study. We compare our method with two alternative regularization strategies—Spectral Norm Regularization (SNR [22]) and adversarial training via PGD [32]—by training on *VOC_Haze_Train* and evaluating out-of-domain on *RTTS*. As presented in Table 5, our approach attains the best *RTTS* performance, surpassing both SNR and PGD. Unlike SNR, which constrains weights globally, our method penalizes $\|\nabla_\theta f_\theta(\boldsymbol{x})\|$, reducing output sensitivity to parameter changes and enabling input-aware smoothness. Compared to PGD-based adversarial training, which requires generating perturbed inputs and increases training cost, our approach achieves implicit robustness without adversarial examples, resulting in more stable and efficient training and no observed degradation on clean inputs.

Method	Baseline	SNR [22]	PGD [32]	Ours
RTTS	49.3	50.1	40.8	**53.2**

Table 5: **Alternative regularization ablation study.**

Architectural Variants Ablation Study. We ablate how deeply to share the encoder between detection and restoration by varying the number of shared stages, training on *VOC_Haze_Train* and evaluating out-of-domain on *RTTS*. As summarized in Table 6, shallower sharing (F1–F2) provides insufficient regularization, while deeper sharing (F1–F4) introduces task interference, supporting our design choice of the first three stages (F1–F3).

Sharing	Baseline	F1–F2	F1–F3 (Ours)	F1–F4
RTTS	49.3	51.9	**53.2**	52.8

Table 6: **Alternative variants ablation study.**

Regularization Coefficients Ablation Study. We ablate the input-space and parameter-space regularization strengths by sweeping $\lambda \in \{0, 5, 10, 20\}$ and $\lambda_p \in \{0, 0.005, 0.01, 0.02\}$, training on *VOC_Haze_Train* and evaluating out-of-domain on *RTTS*. As reported in Table 7, our method consistently outperforms the baseline across a range of coefficient values, with only minor performance variation over the sweep, demonstrating robustness to the choice of regularization magnitudes.

λ	0	10	10	10	20	5
λ_p	0	0.005	0.02	0.01	0.01	0.01
RTTS	49.3	52.9	53.0	**53.2**	53.1	52.8

Table 7: **Regularization coefficients ablation study.**

Lipschitz Continuity Analysis. We analyze changes in Lipschitz continuity in both the input and parameter spaces during training. Specifically, we monitor the upper bound of the Jacobian norm $\sup_{\boldsymbol{x}\in dom(f)} \|\nabla_{\boldsymbol{x}} f_\theta(\boldsymbol{x})\|$ in the input space, where $dom(f)$ represents the domain of input images from Pascal VOC. Additionally, we track the gradient norm $\|\nabla_\theta f_\theta(\boldsymbol{x})\|$ in the parameter space. As shown in Figure 5, LR-YOLOv8 trained with $\mathcal{L}_{\text{res}}$ reduces the Lipschitz constant in both the input and parameter spaces compared to ConvIR→YOLOv8 and YOLOv8 during training. Training with $\|\nabla_\theta f_\theta(\boldsymbol{x})\|$ further promotes Lipschitz continuity in the input and parameter spaces.

Generalization on Other Detection Paradigm. We further validate the generalizability of our LROD by integrating it into the shared backbones of a transformer-based detector (RT-DETR [33])

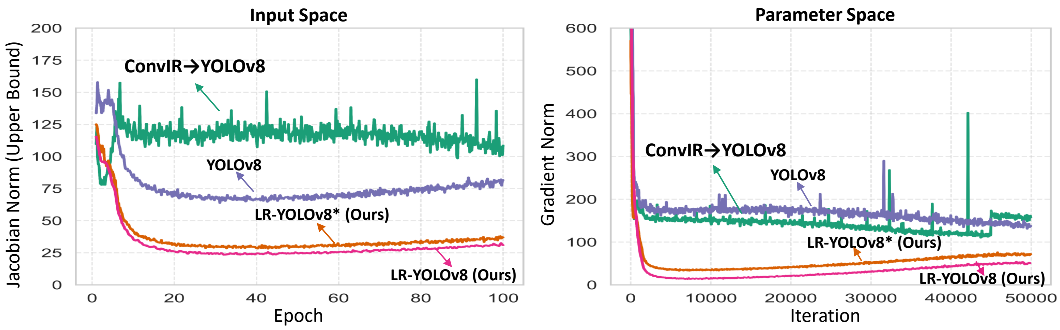


Figure 5: **Comparison of changes in Lipschitz continuity in both the input space and parameter space during training.** Methods include ConvIR→YOLOv8 (Cascade), YOLOv8 (Baseline), LR-YOLOv8* (only trained with $\mathcal{L}_{\text{res}}$), and LR-YOLOv8 (trained with both $\mathcal{L}_{\text{res}}$ and $\|\nabla_\theta f_\theta(\boldsymbol{x})\|$).

Method	RT-DETR [33]		Faster R-CNN [34]	
	VOC_Haze_Val	*RTTS*	*VOC_Haze_Val*	*RTTS*
Baseline	51.5	43.7	69.1	43.2
ConvIR [11]	76.0	43.7	78.5	44.1
IA [1]	76.8	43.6	78.6	41.3
GDIP [3]	72.6	43.5	76.6	44.5
FeatEnHancer [17]	73.3	42.4	77.7	39.4
LROD (Ours)	**78.9**	**45.1**	**80.2**	**45.9**

Table 8: **Generalization on other detection paradigm.** We integrate our LROD into RT-DETR [33] and Faster-RCNN [34].

Method	Motion Blur	Rain	Snow	Haze + Rain
YOLOv8 [9]	50.8	53.1	60.8	50.1
ConvIR [11]→YOLOv8 [9]	80.1	79.9	80.5	79.2
IA [1]→YOLOv8 [9]	79.6	79.9	80.3	78.0
GDIP [3]→YOLOv8 [9]	80.1	79.6	80.4	78.3
FeatEnHancer [17]→YOLOv8 [9]	80.2	79.6	79.6	78.9
LR-YOLOv8 (Ours)	**82.3**	**82.5**	**83.0**	**81.6**

Table 9: **Generalization on other degradation.** We assess the robustness of our method under motion blur, rain, snow, and a haze–rain mixture.

and a two-stage detector (Faster R-CNN [34]). All models are trained on *VOC_Haze_Train* and evaluated on both the synthetic *VOC_Haze_Val* and real-world *RTTS* dataset. As shown in Table 8, LROD consistently outperforms other methods when using other detection paradigms, achieving mAP improvements of 1.4 on *RTTS*, supporting LROD as a plug-and-play regularization framework.

Generalization on Other Degradation. We further assess robustness under additional adverse conditions—motion blur, rain, snow, and a haze–rain mixture—by constructing matched train/validation splits for each degradation and retraining all methods per setting. As reported in Table 9, our method consistently outperforms existing methods, achieving mAP gains of 2.1, 2.6, 2.5, and 2.4, respectively. These results highlight the versatility and robustness of our method across diverse degradation.

6 Conclusion

In this paper, we revisit the integration of image restoration and object detection under adverse conditions through the lens of Lipschitz continuity in both the input and parameter spaces. Our analysis reveals that the inherent mismatch in Lipschitz continuity between these tasks introduces instability and non-smoothness when directly cascaded. To address this, we propose a Lipschitz-regularized framework that harmonizes the two tasks by constraining the Lipschitz continuity of object detection. This is achieved through low-Lipschitz restoration learning to smooth perturbations before detection, alongside parameter-space regularization to stabilize gradient flows during training. We implement this approach as Lipschitz-Regularized YOLO (LR-YOLO), which extends to existing YOLO detectors with minimal overhead. Extensive experiments on haze and low-light benchmarks show that our method improves detection stability and optimization smoothness, contributing to more robust performance in challenging environments.

Limitation and Future Direction. While our method has been validated across a range of adverse conditions—including haze, rain, snow, low-light, and mixed weather—a current limitation is that each input is assumed to contain only a single type of degradation. A valuable future direction would be to extend our framework to handle inputs affected by multiple, concurrent degradations. Another promising direction is to extend our Lipschitz-continuity analysis to camouflaged object detection (COD [35, 36]), as COD involves detecting objects with ambiguous, low-contrast boundaries, posing challenges similar to those in cascaded systems under adverse conditions.

Acknowledgements

This work is supported in part by the National Natural Science Foundation of China (NSFC) under Grant Nos. 62376292, 62325605, and U21A20470; the Guangdong Provincial General Fund under Grant No. 2024A1515010208; and the Guangzhou Science and Technology Program Project under Grant Nos. 2025A04J5465 and 2024A04J6365.

References

[1] W. Liu, G. Ren, R. Yu, S. Guo, J. Zhu, and L. Zhang, "Image-adaptive yolo for object detection in adverse weather conditions," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022.

[2] S. Sun, W. Ren, T. Wang, and X. Cao, "Rethinking image restoration for object detection," *Advances in Neural Information Processing Systems*, 2022.

[3] S. Kalwar, D. Patel, A. Aanegola, K. R. Konda, S. Garg, and K. M. Krishna, "Gdip: Gated differentiable image processing for object detection in adverse conditions," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2023.

[4] C. Li, H. Zhou, Y. Liu, C. Yang, Y. Xie, Z. Li, and L. Zhu, "Detection-friendly dehazing: Object detection in real-world hazy scenes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.

[5] J. Wu and Z. Jin, "Unsupervised variational translator for bridging image restoration and high-level vision tasks," in *European Conference on Computer Vision*, 2024.

[6] S. Bubeck and M. Sellke, "A universal law of robustness via isoperimetry," *Advances in Neural Information Processing Systems*, 2021.

[7] G. Khromov and S. P. Singh, "Some fundamental aspects about lipschitz continuity of neural networks," *International Conference on Learning Representations*, 2024.

[8] A. Wang, H. Chen, L. Liu, K. Chen, Z. Lin, J. Han *et al.*, "Yolov10: Real-time end-to-end object detection," *Advances in Neural Information Processing Systems*, 2024.

[9] G. Jocher, A. Chaurasia, and J. Qiu, "Ultralytics yolov8," https://github.com/ultralytics/ultralytics, 2023.

[10] Y. Cui, Y. Tao, Z. Bing, W. Ren, X. Gao, X. Cao, K. Huang, and A. Knoll, "Selective frequency network for image restoration," in *International Conference on Learning Representations*, 2023.

[11] Y. Cui, W. Ren, X. Cao, and A. Knoll, "Revitalizing convolutional network for image restoration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[12] T. Wang, K. Zhang, T. Shen, W. Luo, B. Stenger, and T. Lu, "Ultra-high-definition low-light image enhancement: A benchmark and transformer-based method," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023.

[13] Y. Cai, H. Bian, J. Lin, H. Wang, R. Timofte, and Y. Zhang, "Retinexformer: One-stage retinex-based transformer for low-light image enhancement," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

[14] M. V. Conde, G. Geigle, and R. Timofte, "Instructir: High-quality image restoration following human instructions," in *European Conference on Computer Vision*, 2024.

[15] R. Zhao, H. Yan, and S. Wang, "Revisiting domain-adaptive object detection in adverse weather by the generation and composition of high-quality pseudo-labels," in *European Conference on Computer Vision*, 2024.

[16] J. Kim, J. Oh, and K. M. Lee, "Beyond image super-resolution for image recognition with task-driven perceptual loss," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

[17] K. A. Hashmi, G. Kallempudi, D. Stricker, and M. Z. Afzal, "Featenhancer: Enhancing hierarchical features for object detection and beyond under low-light vision," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

[18] L. Meunier, B. J. Delattre, A. Araujo, and A. Allauzen, "A dynamical system perspective for lipschitz neural networks," in *International Conference on Machine Learning*, 2022.

[19] A. Araujo, A. Havens, B. Delattre, A. Allauzen, and B. Hu, “A unified algebraic perspective on lipschitz neural networks,” *International Conference on Learning Representations*, 2023.

[20] R. Wang and I. Manchester, “Direct parameterization of lipschitz-bounded deep networks,” in *International Conference on Machine Learning*, 2023.

[21] S. Arora, R. Ge, B. Neyshabur, and Y. Zhang, “Stronger generalization bounds for deep nets via a compression approach,” in *International Conference on Machine Learning*, 2018.

[22] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, “Spectral normalization for generative adversarial networks,” *International Conference on Learning Representations*, 2018.

[23] K. Leino, Z. Wang, and M. Fredrikson, “Globally-robust neural networks,” in *International Conference on Machine Learning*, 2021.

[24] X. Qi, J. Wang, Y. Chen, Y. Shi, and L. Zhang, “Lipsformer: Introducing lipschitz continuity to vision transformers,” *International Conference on Learning Representations*, 2023.

[25] F. Latorre, P. Rolland, and V. Cevher, “Lipschitz constant estimation of neural networks via sparse polynomial optimization,” *International Conference on Learning Representations*, 2020.

[26] M. Everingham, S. A. Eslami, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, “The pascal visual object classes challenge: A retrospective,” *International Journal of Computer Vision*, 2015.

[27] H. Li, Z. Xu, G. Taylor, C. Studer, and T. Goldstein, “Visualizing the loss landscape of neural nets,” *Advances in Neural Information Processing Systems*, 2018.

[28] A. Bruhn, J. Weickert, and C. Schnörr, “Lucas/kanade meets horn/schunck: Combining local and global optic flow methods,” *International Journal of Computer Vision*, 2005.

[29] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, “Microsoft coco: Common objects in context,” in *European Conference on Computer Vision*. Springer, 2014.

[30] B. Li, W. Ren, D. Fu, D. Tao, D. Feng, W. Zeng, and Z. Wang, “Benchmarking single-image dehazing and beyond,” *IEEE Transactions on Image Processing*, 2018.

[31] Y. P. Loh and C. S. Chan, “Getting to know low-light images with the exclusively dark dataset,” *Computer Vision and Image Understanding*, 2019.

[32] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, “Towards deep learning models resistant to adversarial attacks,” *International Conference on Learning Representations*, 2018.

[33] Y. Zhao, W. Lv, S. Xu, J. Wei, G. Wang, Q. Dang, Y. Liu, and J. Chen, “Detrs beat yolos on real-time object detection,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

[34] S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” *Advances in Neural Information Processing Systems*, 2015.

[35] D.-P. Fan, G.-P. Ji, G. Sun, M.-M. Cheng, J. Shen, and L. Shao, “Camouflaged object detection,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020.

[36] X. Hu, S. Wang, X. Qin, H. Dai, W. Ren, D. Luo, Y. Tai, and L. Shao, “High-resolution iterative feedback network for camouflaged object detection,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023.

[37] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, “The unreasonable effectiveness of deep features as a perceptual metric,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018.

Appendix

A Detailed Proofs

Remark 3 (Lipschitz Regularization via Low-Lipschitz Restoration). *Let: $f_{\theta_b,\theta_d} = f_{\theta_d} \circ f_{\theta_b}$ denote the object detection model, where $f_{\theta_b}(\cdot;\theta_b)$ is the backbone network parameterized by θ_b, and $f_{\theta_d}(\cdot;\theta_d)$ is the detection head parameterized by θ_d. Similarly, let: $g_{\theta_b,\theta_r} = f_{\theta_r} \circ f_{\theta_b}$ represent the image restoration model, where $f_{\theta_r}(\cdot;\theta_r)$ is the restoration head parameterized by θ_r, sharing the same backbone f_{θ_b}. Given a weighted combination of the detection loss and the restoration loss:*

$$\mathcal{L}(\theta_b,\theta_d,\theta_r) = \mathcal{L}_{\mathrm{det}}(f_{\theta_b,\theta_d}) + \lambda \cdot \mathcal{L}_{\mathrm{res}}(g_{\theta_b,\theta_r}), \quad \lambda > 0$$

Let $\mathrm{Lip}(f_{\theta_b}) := \sup_{\boldsymbol{x}} \|J_{f_{\theta_b}}(\boldsymbol{x})\|$ be the Lipschitz constant of f_{θ_b} defined by jacobian norm. If:

1. *$\mathcal{L}_{\mathrm{res}}$ is Lipschitz continuous and $\|\nabla_{\theta_b}\mathcal{L}_{\mathrm{res}}(g_{\theta_b,\theta_r})\| \leq G$ for $G < \|\nabla_{\theta_b}\mathcal{L}_{\mathrm{det}}(f_{\theta_b,\theta_d})\|$;*
2. *There exists a training sample $\boldsymbol{x}^\star$ and $\gamma > 0$ such that: $\left\langle \nabla_{\theta_b}\|J_{f_{\theta_b}}(\boldsymbol{x}^\star)\|, \nabla_{\theta_b}\mathcal{L}_{\mathrm{res}}(g_{\theta_b,\theta_r}) \right\rangle \geq \gamma$,*

then under continuous-time gradient descent ${\theta_b}^{(t+1)} \leftarrow {\theta_b}^{(t)} - \mu \cdot \nabla_{\theta_b}\mathcal{L}(\theta_b,\theta_d,\theta_r)$ (μ denotes the learning rate), the evolution of the Lipschitz constant satisfies:

$$\frac{d}{dt}\left[\mathrm{Lip}(f_{\theta_b})\right] \leq -\lambda \cdot \gamma + \xi(t)$$

where $\xi(t) := \left\langle \nabla_{\theta_b}\|J_{f_{\theta_b}}(\boldsymbol{x}^\star)\|, \nabla_{\theta_b}\mathcal{L}_{\mathrm{det}}(f_{\theta_b,\theta_d}) \right\rangle$ is the unconstrained change induced by the detection loss and γ is the regularization via the restoration task.

This suggests that integrating the image restoration task directly into the detector's feature learning by sharing the detector's backbone helps suppress the model's sensitivity to input perturbations during training, effectively acting as a Lipschitz regularization.

Proof. Using Theorem 1 in [25], the Lipschitz constant of f_{θ_b} is:

$$\mathrm{Lip}(f_{\theta_b}) = \sup_{\boldsymbol{x}} \|J_{f_{\theta_b}}(\boldsymbol{x})\|$$

Let $\boldsymbol{x}^\star$ be the input that attains or approximates this supremum. Then, during continuous-time gradient descent:

$$\frac{d}{dt}\left[\mathrm{Lip}(f_{\theta_b})\right] = \frac{d}{dt}\|J_{f_{\theta_b}}(\boldsymbol{x}^\star)\| = \left\langle \nabla_{\theta}\|J_{f_{\theta_b}}(\boldsymbol{x}^\star)\|_2, -\nabla_{\theta_b}\mathcal{L}(\theta_b,\theta_d,\theta_r) \right\rangle$$

Substituting the joint loss:

$$\frac{d}{dt}\left[\mathrm{Lip}(f_{\theta_b})\right] = -\left\langle \nabla_{\theta_b}\|J_{f_{\theta_b}}(\boldsymbol{x}^\star)\|, \nabla_{\theta_b}\mathcal{L}_{\mathrm{det}}(f_{\theta_b,\theta_d}) + \lambda \cdot \nabla_{\theta_b}\mathcal{L}_{\mathrm{res}}(g_{\theta_b,\theta_r}) \right\rangle$$

Breaking into two components:

$$\frac{d}{dt}\left[\mathrm{Lip}(f_{\theta_b})\right] = -\left\langle \nabla_{\theta_b}\|J_{f_{\theta_b}}(\boldsymbol{x}^\star)\|, \nabla_{\theta_b}\mathcal{L}_{\mathrm{det}}(f_{\theta_b,\theta_d}) \right\rangle - \lambda \cdot \left\langle \nabla_{\theta_b}\|J_{f_{\theta_b}}(\boldsymbol{x}^\star)\|, \nabla_{\theta_b}\mathcal{L}_{\mathrm{res}}(g_{\theta_b,\theta_r}) \right\rangle$$

By Assumption 2, the second term is lower bounded:

$$\langle \nabla_{\theta}\|J_{f_{\theta}}(\boldsymbol{x}^\star)\|_2, \nabla_{\theta}\mathcal{L}_{\mathrm{res}}(g_{\theta_b,\theta_r}) \rangle \geq \gamma$$

Define the first term as $\xi(t)$, then:

$$\frac{d}{dt}\left[\mathrm{Lip}(f_{\theta})\right] \leq -\lambda\gamma + \xi(t)$$

which completes the proof. $\square$

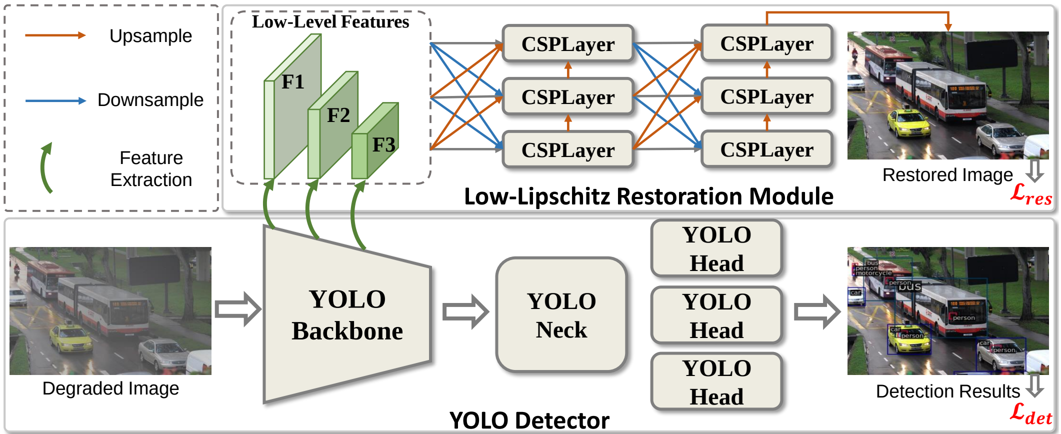


Figure 6: The overall architecture of Lipschitz-Regularized YOLO (LR-YOLO).

Dataset	image	person	bicycle	car	bus	motorbike	Total
VOC_Haze_Train	8111	13256	1064	3267	822	1052	19561
VOC_Haze_Val	2734	4528	337	1201	213	325	6604
VOC_Haze_Val (all objects)	2734	5136	389	1528	254	367	7674
RTTS [30]	4322	7950	534	18413	1838	862	29577
RTTS (all objects)	4322	11366	698	25317	2590	1232	41203

Table 10: Statistics of haze datasets in terms of image count and object annotations per class.

B Detailed Model

To efficiently and effectively harmonize image restoration and object detection, we integrate image restoration learning into the feature extraction process of the object detection backbone. This integration implicitly enforces Lipschitz continuity during training, thereby enhancing the stability of the detector under varying degradation intensities. As illustrated in Figure 6, we extend existing YOLO detectors by extracting low-level features from the first three stages of the backbone *without modifying the original network architecture*. These features are then processed by a lightweight restoration-aware module, which reconstructs a clean version of the input image and facilitates the learning of smoother and more stable representations within the detection network.

YOLO Detector. The YOLO architecture is a one-stage object detection framework that performs detection in a single forward pass, achieving high efficiency and speed. It consists of three main components, *i.e.*, the Backbone, which extracts visual features from the input image; the Neck, which aggregates multi-scale features to enhance representation; and the Head, which predicts bounding boxes, class scores, and objectness. Due to its high computational efficiency and ease of deployment on edge devices, YOLO is widely used in real-time detection applications.

Low-Lipschitz Restoration Module. To improve the stability of object detection under adverse imaging conditions, we introduce a *Restoration-Aware Module* integrated into the YOLO framework. As illustrated in Figure 6, we extract low-level features from the first three stages of the YOLO backbone (denoted as F1, F2, and F3), which preserve rich spatial and textural information essential for image restoration. Inspired by the design of the YOLO Neck and Head, these features are passed through a restoration-specific neck and decoder composed of multiple Cross Stage Partial layers (CSPLayer). The module adopts a densely connected architecture that facilitates multi-scale feature fusion, which is crucial for effective restoration learning. By progressively refining the low-level representations, the module reconstructs a restored version of the input image that is less affected by visual degradation. This restoration-aware module not only contributes to the stability of YOLO detectors during training due to the inherently smoothness of restoration, but also enhances the low-level features used by the detector for downstream detection tasks.

Dataset	image	person	bicycle	car	bus	motorbike	boat	bottle	cat	chair	dog	Total
VOC_Dark_Train	12334	13256	1064	3267	822	1052	1140	1764	1593	3152	2025	29135
VOC_Dark_Val	3760	4528	337	1201	213	325	263	469	358	756	489	8939
VOC_Dark_Val (all objects)	3760	5183	389	1533	254	367	393	646	368	1268	529	10930
ExDark	2563	2235	418	919	164	242	515	433	425	609	490	6450

Table 11: Statistics of low-light datasets in terms of image count and object annotations per class.

C Detailed Datasets

Datasets cover two challenging conditions: *hazy weather* and *low-light environments*. For both settings, we use real-world datasets for out-of-domain evaluation and construct synthetic training/validation sets based on PASCAL VOC [26], following IA-YOLO [1], ReForDe [2] and Vat [5]:

1) Training and Validation Data: We construct synthetic datasets by selecting PASCAL VOC [26] images containing the relevant object categories:

- *VOC_Haze_Train* and *VOC_Haze_Val* consist of $8,111$ and $2,734$ images respectively. Haze is synthesized *online* during training using the atmospheric scattering model with $\beta \in [0.5, 1.5]$, while validation images are synthesized *offline* once for reproducibility.
- *VOC_Dark_Train* and *VOC_Dark_Val* consist of $12,334$ and $3,760$ images respectively. Low-light is simulated *online* during training and *offline* in validation via gamma correction with $\gamma \in [1.5, 5]$.

2) Real-world Test Data. We adopt **two** benchmark datasets for the out-of-domain evaluation:

- ***RTTS*** [30]: contains $4,322$ real-world hazy images annotated with 5 object categories, *i.e.*, *Person*, *Car*, *Bus*, *Bicycle*, and *Motorbike*.
- ***ExDark*** [31]: contains $2,563$ real-world low-light images labeled with 10 categories, *i.e.*, *People*, *Car*, *Bus*, *Bicycle*, *Motorbike*, *Boat*, *Bottle*, *Chair*, *Dog*, and *Cat*.

The object annotations per class in the above datasets are presented in Table 10 and Table 11.

D Detailed Detection Results

We report the average precision for each category on the *VOC_Haze_Val* and *RTTS* datasets, as shown in Table 12, and on the *VOC_Dark_Val* and *ExDark* datasets, as presented in Table 14.

E Image Restoration Evaluation

We evaluate the restoration performance of our LR-YOLO under haze and low-light conditions using peak signal-to-noise ratio (PSNR) for pixel-level fidelity and learned perceptual image patch similarity (LPIPS) [37] for perceptual similarity. Table 13 presents the restoration results on *VOC_Haze_Val* and *VOC_Dark_Val*. We compare our method with the image dehazing technique ConvIR [11], low-light enhancement method Retinexformer [13], and image adaptive methods IA [1] and GDIP [3]. Our full model achieves the best LPIPS score and competitive PSNR on both *VOC_Haze_Val* and *VOC_Dark_Val*, outperforming the baseline YOLOv8 and image adaptive pipelines. Additionally, our restoration-only variant (LR-YOLOv8 (Restoration-Only) trained only with restoration learning) achieves a balanced improvement in reconstruction fidelity.

F Qualitative Comparison

Figure 2 (a) illustrates that the detector features of the cascade method are highly sensitive to minor haze density variations $\Delta\boldsymbol{x}$, while our Lipschitz-regularized framework maintains stability. A qualitative comparison of detection result stability is presented in Figure 7. When two haze inputs $\boldsymbol{x}$ and $\boldsymbol{x} + \Delta\boldsymbol{x}$ with slight haze density variations $\Delta\boldsymbol{x}$ are fed into the model, the detection results of the cascade method (ConvIR→YOLOv8) exhibits significant instability even though those haze can be mitigated by the image dehazing method. For example, a car is detected in one case but

Methods	*VOC_Haze_Val*						*RTTS* [30]					
	mAP	Person	Car	Bus	Bicycle	Motorbike	**mAP**	Person	Car	Bus	Bicycle	Motorbike
YOLOv10	50.5	55.3	63.5	46.3	48.1	39.5	42.6	70.7	48.9	20.8	41.0	31.5
SFNet→YOLOv10	77.9	79.2	82.4	76.5	76.1	75.3	45.5	72.2	53.2	23.9	42.6	35.4
SFNet→YOLOv10†	79.1	80.5	83.6	77.4	76.9	77.3	46.6	73.4	54.8	24.9	42.1	38.0
SFNet→YOLOv10‡	79.3	80.4	83.2	77.8	77.5	77.7	45.8	72.4	54.5	24.1	41.7	36.2
ConvIR→YOLOv10	79.9	80.5	83.7	79.2	77.8	78.1	46.1	72.6	53.9	24.3	43.7	36.3
ConvIR→YOLOv10†	80.1	80.9	84.4	79.2	77.3	78.6	46.6	73.2	55.3	25.0	42.4	37.0
ConvIR→YOLOv10‡	80.5	81.1	83.9	80.1	78.3	79.0	46.5	72.5	54.7	24.4	43.7	37.2
IA→YOLOv10	79.9	79.9	83.8	76.3	81.4	78.1	45.4	72.0	55.4	22.5	42.3	34.6
GDIP→YOLOv10	79.2	79.3	83.4	76.8	79.4	77.4	47.2	73.2	54.6	24.6	45.3	38.1
FeatEnHancer→YOLOv10	79.8	79.2	82.9	78.4	80.0	78.6	46.7	72.0	53.4	22.9	43.8	36.6
LR-YOLOv10 (Ours)	**82.5**	82.0	86.6	79.7	83.4	81.8	**49.2**	74.1	58.5	27.7	44.9	42.0
YOLOv8	54.3	58.2	65.6	51.8	52.3	43.7	45.3	73.3	53.4	21.1	45.6	33.3
SFNet→YOLOv8	79.2	80.3	83.8	75.9	78.9	77.1	48.9	75.2	59.0	23.9	48.7	37.8
SFNet→YOLOv8†	80.8	82.0	85.2	78.2	79.4	78.9	49.1	75.6	59.2	25.2	47.3	38.0
SFNet→YOLOv8‡	80.3	81.6	84.8	77.5	79.4	78.0	49.3	75.6	59.8	24.4	48.3	38.4
ConvIR→YOLOv8	80.5	81.3	84.8	78.2	79.7	78.5	49.3	75.3	59.5	24.2	48.9	38.5
ConvIR→YOLOv8†	80.9	82.1	85.6	77.3	79.7	79.6	49.5	75.3	59.9	25.7	47.7	39.0
ConvIR→YOLOv8‡	81.4	82.4	85.9	77.9	81.3	79.9	50.1	75.6	60.1	25.8	49.4	39.5
IA→YOLOv8	80.6	81.6	85.5	77.2	80.8	78.0	47.7	73.3	58.7	25.6	44.6	36.5
GDIP→YOLOv8	81.0	81.7	85.9	78.1	79.9	79.3	50.3	75.8	59.7	27.8	49.9	38.0
FeatEnHancer→YOLOv8	81.2	81.4	85.3	77.0	81.8	80.4	48.4	74.8	57.9	26.1	47.1	40.7
LR-YOLOv8 (Ours)	**83.3**	83.7	87.6	79.3	84.4	82.8	**53.2**	76.0	62.9	30.0	50.2	46.9
Methods	*VOC_Haze_Val (all objects)*						*RTTS (all objects)* [30]					
	mAP	Person	Car	Bus	Bicycle	Motorbike	**mAP**	Person	Car	Bus	Bicycle	Motorbike
YOLOv10	48.3	52.6	54.3	45.9	47.9	40.6	36.2	56.3	43.3	17.2	37.5	26.5
SFNet→YOLOv10	71.1	73.9	73.6	67.5	70.5	70.1	38.4	58.1	46.1	19.5	39.1	29.1
SFNet→YOLOv10†	73.8	77.1	76.7	70.4	71.5	73.3	39.3	60.4	48.3	20.7	38.0	28.9
SFNet→YOLOv10‡	72.8	76.7	75.2	69.1	71.3	71.6	39.2	60.3	48.7	19.8	38.4	29.1
ConvIR→YOLOv10	72.8	76.4	75.0	69.3	71.5	71.6	38.7	58.1	46.4	19.9	40.0	29.4
ConvIR→YOLOv10†	74.1	77.5	76.9	70.0	71.9	74.1	39.0	59.7	48.9	20.6	37.6	28.3
ConvIR→YOLOv10‡	74.0	77.3	77.0	69.3	72.3	74.2	39.9	60.2	49.0	20.2	39.8	30.2
IA→YOLOv10	73.0	76.5	75.7	69.4	72.6	70.9	37.3	56.5	45.8	19.8	35.6	28.7
GDIP→YOLOv10	73.1	76.4	75.4	69.5	71.0	73.0	39.8	60.5	46.6	22.6	39.8	29.5
FeatEnHancer→YOLOv10	73.4	76.4	75.5	68.7	72.9	73.4	38.8	59.3	45.3	20.7	37.7	31.0
LR-YOLOv10 (Ours)	**76.5**	79.2	79.1	71.1	75.5	77.4	**42.4**	60.9	51.4	23.9	40.0	36.0
YOLOv8	44.7	50.2	53.8	41.7	42.1	35.6	33.8	54.2	40.8	15.1	33.5	23.4
SFNet→YOLOv8	70.1	72.7	71.7	68.2	69.4	68.6	35.9	55.7	41.6	18.8	35.7	27.9
SFNet→YOLOv8†	72.1	75.5	74.4	69.0	69.5	71.9	37.1	58.0	44.6	19.9	33.9	28.9
SFNet→YOLOv8‡	71.7	75.1	73.2	68.9	69.5	71.4	36.0	55.9	42.6	19.1	34.4	28.2
ConvIR→YOLOv8	72.2	75.3	73.3	70.5	70.8	70.9	36.0	55.7	42.0	19.3	35.8	27.3
ConvIR→YOLOv8†	72.9	76.2	75.0	70.6	70.2	72.5	37.2	57.8	45.1	20.2	34.6	28.4
ConvIR→YOLOv8‡	72.6	76.0	73.7	70.2	70.8	72.5	36.5	55.9	42.8	19.4	35.3	29.0
IA→YOLOv8	72.0	74.5	74.0	68.1	72.8	70.8	35.8	55.5	43.4	18.3	34.3	27.8
GDIP→YOLOv8	70.9	72.7	72.6	68.1	70.2	70.7	37.0	56.1	42.7	19.7	37.0	29.2
FeatEnHancer→YOLOv8	71.6	72.7	72.7	69.8	71.6	71.2	35.8	55.2	41.7	18.3	35.6	28.1
LR-YOLOv8 (Ours)	**74.4**	77.5	76.7	71.3	75.0	74.6	**38.5**	59.1	45.9	21.3	37.3	33.0

Table 12: Detailed results on *VOC_Haze_Val* and *RTTS* [30], with models trained on *VOC_Haze_Train*.

Methods	*VOC_Haze_Val*	
	LPIPS ↓	PSNR ↑
ConvIR [11]→YOLOv8	0.180	**25.44**
IA [1]→YOLOv8	0.270	13.12
GDIP [3]→YOLOv8	0.234	15.75
YOLOv8 (Baseline)	0.382	13.51
LR-YOLOv8 (Restoration-Only)	0.195	23.72
LR-YOLOv8 (Ours)	**0.133**	22.72

Methods	*VOC_Dark_Val*	
	LPIPS ↓	PSNR ↑
Retinexformer [13]→YOLOv8	0.293	**21.46**
IA [1]→YOLOv8	0.195	20.73
GDIP [3]→YOLOv8	0.189	18.82
YOLOv8 (Baseline)	0.315	12.00
LR-YOLOv8 (Restoration-Only)	0.245	21.19
LR-YOLOv8 (Ours)	**0.179**	21.05

Table 13: Restoration results on *VOC_Haze_Val* and *VOC_Dark_Val*.

not in another, and a person is suddenly undetected. This highlights the instability inherent in the cascade framework. Visual examples in Figure 8 (haze condition) and Figure 9 (low-light condition) qualitatively illustrate the effectiveness of our method in improving detection accuracy and perceptual quality, thereby enhancing human trust in detection results.

Methods	*VOC_Dark_Val*										
	mAP	Person	Car	Bus	Bicycle	Motorbike	Boat	Bottle	Chair	Dog	Cat
YOLOv10	62.1	68.7	72.9	69.0	70.2	67.1	56.2	47.0	43.0	61.7	65.3
LLFormer→YOLOv10	65.6	72.6	78.0	71.8	74.0	70.7	59.0	46.9	46.2	66.1	70.3
LLFormer→YOLOv10†	64.7	73.9	77.4	69.7	73.5	71.7	56.4	47.2	43.1	65.0	68.6
LLFormer→YOLOv10‡	66.3	73.2	78.2	71.8	73.7	71.7	60.1	50.0	46.0	67.5	70.6
RetinexFormer→YOLOv10	66.3	73.3	78.8	73.6	72.9	70.7	59.8	49.7	46.0	66.4	71.7
RetinexFormer→YOLOv10†	66.0	74.8	78.5	72.7	75.4	72.7	57.2	49.5	45.2	64.9	69.2
RetinexFormer→YOLOv10‡	66.9	74.9	78.6	72.4	75.2	72.6	60.3	49.6	46.5	67.4	71.8
IA→YOLOv10	66.0	73.1	78.5	68.2	74.5	69.4	60.6	52.0	48.2	67.1	68.0
GDIP→YOLOv10	65.8	74.3	77.9	71.2	77.4	72.5	56.5	49.4	45.4	67.0	66.8
FeatEnHancer→YOLOv10	67.6	75.3	79.1	73.6	76.7	73.8	59.6	52.3	49.2	68.3	68.6
LR-YOLOv10 (Ours)	**70.6**	77.6	82.5	76.3	78.3	75.9	65.6	56.3	52.1	69.3	72.5
YOLOv8	63.4	70.2	76.1	69.3	72.1	67.4	59.9	45.9	44.2	63.4	65.9
LLFormer→YOLOv8	66.2	74.5	80.0	71.5	74.3	71.7	59.0	45.4	47.2	67.9	70.3
LLFormer→YOLOv8†	66.2	76.3	80.6	71.9	75.1	71.7	59.2	45.8	46.2	65.3	70.0
LLFormer→YOLOv8‡	66.2	75.0	80.7	71.6	75.7	72.7	59.5	47.4	45.3	65.5	68.4
RetinexFormer→YOLOv8	67.8	76.5	81.1	73.5	73.8	71.0	62.9	49.8	47.7	69.4	72.2
RetinexFormer→YOLOv8†	67.7	77.5	80.9	72.8	74.8	73.6	61.6	48.6	49.6	67.3	70.2
RetinexFormer→YOLOv8‡	68.6	77.4	81.6	75.2	75.2	74.3	61.3	51.0	49.4	68.5	71.8
IA→YOLOv8	66.5	73.2	79.8	70.7	73.1	73.0	61.6	50.1	48.6	66.5	68.4
GDIP→YOLOv8	68.9	77.1	81.6	73.1	76.2	75.0	61.9	51.4	49.9	69.2	73.3
FeatEnHancer→YOLOv8	68.7	75.8	79.7	73.2	76.9	76.7	60.6	52.5	51.0	69.6	71.3
LR-YOLOv8 (Ours)	**71.7**	78.5	82.9	77.2	79.7	78.3	62.9	60.1	52.6	70.3	74.3

Methods	*VOC_Dark_Val (all objects)*										
	mAP	Person	Car	Bus	Bicycle	Motorbike	Boat	Bottle	Chair	Dog	Cat
YOLOv10	55.8	63.8	66.2	61.5	64.0	61.5	46.1	36.7	33.9	60.6	64.3
LLFormer→YOLOv10	58.7	69.6	70.1	64.1	67.3	65.5	46.3	36.1	36.6	63.9	67.8
LLFormer→YOLOv10†	58.8	70.3	71.0	64.6	67.2	66.1	45.7	36.9	36.6	61.9	67.6
LLFormer→YOLOv10‡	59.2	70.3	71.1	64.4	67.7	67.2	46.9	37.8	36.5	63.0	67.4
RetinexFormer→YOLOv10	59.5	70.2	70.9	66.0	65.3	64.5	48.4	38.6	37.2	64.5	69.4
RetinexFormer→YOLOv10†	60.0	71.5	71.5	65.1	67.4	67.8	47.5	38.7	38.5	62.8	69.0
RetinexFormer→YOLOv10‡	61.0	71.4	72.2	67.1	69.1	68.8	48.3	40.9	38.6	64.6	69.4
IA→YOLOv10	59.2	68.5	70.5	64.7	65.4	66.9	48.0	40.1	38.4	62.9	66.9
GDIP→YOLOv10	61.2	71.3	71.9	65.1	68.6	68.4	49.2	41.9	39.5	65.0	71.5
FeatEnHancer→YOLOv10	60.8	69.7	69.9	65.6	69.3	70.0	47.0	41.9	40.7	65.4	68.9
LR-YOLOv10 (Ours)	**63.9**	72.3	73.3	70.1	72.3	71.6	49.0	48.2	42.3	67.1	72.7
YOLOv8	55.0	62.3	64.2	61.7	62.9	60.9	42.9	37.5	34.4	58.7	64.2
LLFormer→YOLOv8	58.0	67.3	68.1	64.7	66.3	64.1	45.3	37.5	36.9	62.1	67.6
LLFormer→YOLOv8†	57.5	68.0	67.9	63.8	65.3	64.9	43.4	37.9	34.7	61.5	67.4
LLFormer→YOLOv8‡	59.2	70.3	71.1	64.4	67.7	67.2	46.9	37.8	63.5	63.0	67.4
RetinexFormer→YOLOv8	58.6	68.3	68.5	65.7	64.9	63.3	45.9	40.0	36.8	62.9	70.1
RetinexFormer→YOLOv8†	58.4	68.9	68.3	65.0	66.8	66.5	43.7	40.1	36.9	61.0	67.1
RetinexFormer→YOLOv8‡	59.2	68.8	68.7	65.5	67.8	65.8	46.6	39.5	37.3	63.4	69.1
IA→YOLOv8	58.7	68.5	68.8	62.0	67.8	64.1	47.7	41.9	67.9	63.2	64.9
GDIP→YOLOv8	58.5	68.5	68.0	64.1	69.1	67.1	44.4	39.3	36.5	62.9	65.2
FeatEnHancer→YOLOv8	59.9	69.1	69.0	65.5	68.5	68.2	45.7	41.1	40.0	64.2	67.3
LR-YOLOv8 (Ours)	**62.7**	71.5	72.4	68.1	71.1	70.1	50.1	45.6	41.3	66.2	71.1

Methods	***ExDark*** [31]										
	mAP	Person	Car	Bus	Bicycle	Motorbike	Boat	Bottle	Chair	Dog	Cat
YOLOv10	50.0	55.5	52.4	63.3	58.9	31.8	42.0	48.4	42.2	58.0	47.5
LLFormer→YOLOv10	46.6	50.7	47.6	55.2	58.0	30.5	41.0	48.2	38.8	52.5	43.7
LLFormer→YOLOv10†	47.9	53.5	50.6	57.9	60.7	32.7	39.6	49.3	38.4	52.2	43.8
LLFormer→YOLOv10‡	48.6	53.9	49.6	57.9	60.9	33.3	41.7	50.0	40.0	52.6	45.9
RetinexFormer→YOLOv10	47.6	51.3	49.4	58.1	57.3	31.4	41.6	50.6	39.2	54.1	43.2
RetinexFormer→YOLOv10†	49.5	54.5	51.6	60.6	60.8	34.6	42.5	50.6	40.0	53.3	46.5
RetinexFormer→YOLOv10‡	49.5	55.1	50.2	58.5	60.7	34.4	41.9	51.0	42.6	55.0	45.8
IA→YOLOv10	49.6	56.5	52.0	60.0	60.8	32.1	43.0	48.9	41.5	55.8	45.1
GDIP→YOLOv10	51.2	57.1	51.0	61.7	65.1	36.0	44.8	47.7	43.6	57.1	48.4
FeatEnHancer→YOLOv10	51.8	55.5	50.0	63.6	61.7	36.5	43.5	51.1	45.7	59.5	50.9
LR-YOLOv10 (Ours)	**54.5**	60.2	56.3	66.9	66.6	37.9	43.7	54.9	44.8	61.6	52.1
YOLOv8	49.2	53.1	50.0	63.8	58.9	34.0	43.8	46.3	43.6	53.8	44.8
LLFormer→YOLOv8	46.3	49.3	45.6	56.2	60.5	34.2	41.9	42.3	40.3	50.9	42.0
LLFormer→YOLOv8†	47.0	51.7	46.8	58.1	62.6	37.4	40.5	43.2	38.9	50.2	41.0
LLFormer→YOLOv8‡	49.5	54.3	47.9	60.0	63.9	37.0	44.3	46.0	42.0	53.9	45.6
RetinexFormer→YOLOv8	47.6	50.4	46.8	59.3	58.5	34.4	43.6	44.2	42.0	52.7	44.4
RetinexFormer→YOLOv8†	45.8	49.8	46.9	55.6	60.9	35.3	41.6	42.9	39.0	47.7	38.2
RetinexFormer→YOLOv8‡	47.5	52.2	46.7	57.0	61.4	35.1	43.4	44.0	40.8	51.6	42.9
IA→YOLOv8	50.4	55.9	50.1	64.2	61.9	34.5	45.5	48.5	43.1	54.1	46.2
GDIP→YOLOv8	48.9	51.7	47.8	63.4	61.1	36.7	42.3	43.7	39.1	55.5	47.9
FeatEnHancer→YOLOv8	50.9	54.0	48.9	65.9	61.8	37.1	45.4	45.4	43.0	58.3	49.4
LR-YOLOv8 (Ours)	**53.8**	57.2	53.7	69.0	66.3	40.3	44.3	53.3	44.7	59.6	49.4

Table 14: Detailed results on *VOC_Dark_Val* and *ExDark* [31], with models trained on *VOC_Dark_Train*.

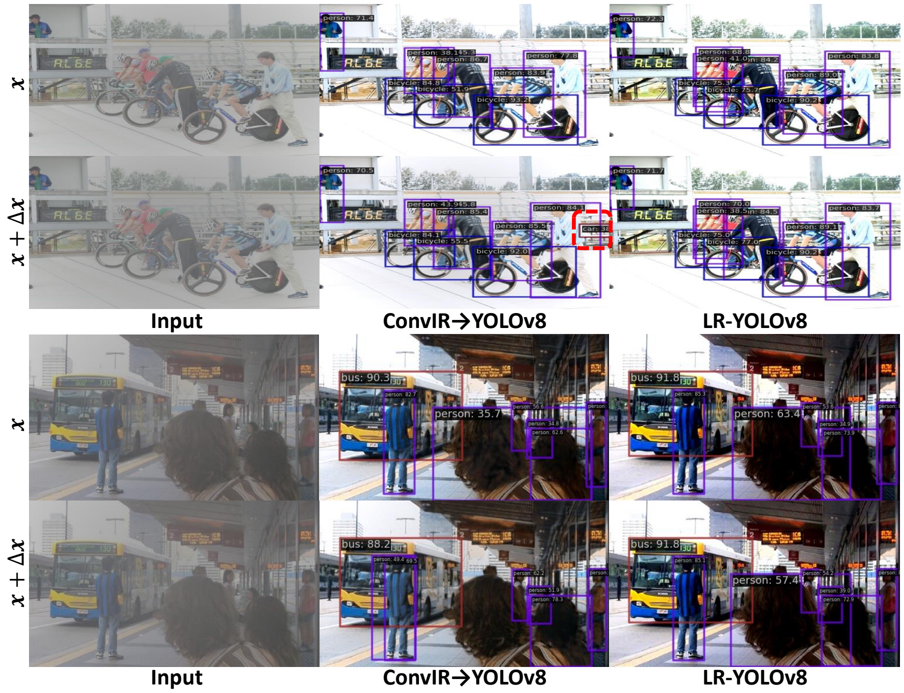


Figure 7: Qualitative comparison of detection result stability between the cascade method (ConvIR→YOLOv8) and our LR-YOLOv8.

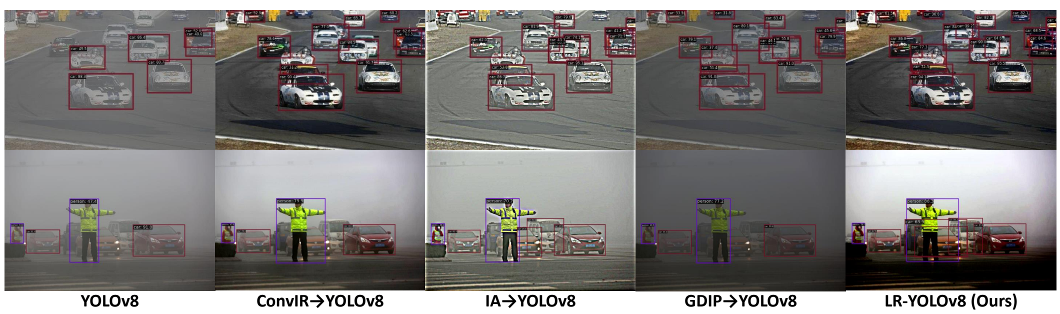


Figure 8: Qualitative comparisons on both *VOC_Haze_Val* and *RTTS* between our LR-YOLOv8 and other methods.

Figure 9: Qualitative comparisons on both *VOC_Dark_Val* and *ExDark* between our LR-YOLOv8 and other methods.

G Broader Impacts

Improving object detection in adverse weather and low-light environments has significant implications for safety-critical applications, such as autonomous driving, traffic surveillance, and search-and-rescue missions. In particular, autonomous vehicles often operate under unpredictable environmental conditions. Failure to accurately detect pedestrians, vehicles, or obstacles in foggy or nighttime scenarios can lead to life-threatening consequences. Our method aims to fill this gap by jointly enhancing visual clarity and detection accuracy, offering a potential safety upgrade to existing perception pipelines.

Nevertheless, this line of research also entails broader considerations. First, the deployment of advanced visual detection systems could increase surveillance capabilities in urban and rural areas. While this may improve public security, it also raises concerns about privacy and the potential for misuse by authoritarian entities.

Second, performance across different demographic and geographic contexts should be evaluated. Adverse weather conditions may vary significantly between regions (e.g., smog vs. marine fog), and ensuring that models generalize fairly across different environments and communities is crucial to avoid biased deployment outcomes.

Lastly, we acknowledge that improved detection in low-visibility environments might be repurposed for military or security applications. While the proposed method is designed for civilian safety and transportation enhancement, dual-use risks exist.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes] , [No] , or [NA] .
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes] " is generally preferable to "[No] ", it is perfectly acceptable to answer "[No] " provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No] " or "[NA] " is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

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

   Answer: [Yes]

   Justification: The abstract and introduction clearly state the claims made.

   Guidelines:

   - The answer NA 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 NA 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: The paper includes a section discussing limitations.

Guidelines:

- The answer NA 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: [Yes]

   Justification: This paper provides the proof in the appendix.

   Guidelines:

   - The answer NA 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: This paper provides PyTorch-like codes for reproducibility in the appendix.

   Guidelines:

   - The answer NA 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: [No]

   Justification: This paper does not provide open access to data and code.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.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://nips.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, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: This paper specifies all relevant training and test details.

   Guidelines:

   - The answer NA 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: [No]

   Justification: This paper does not report error bars.

   Guidelines:

   - The answer NA 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: This paper provides details on the type of compute workers (GPUs), memory requirements, and time of execution for each experiment, ensuring reproducibility.

   Guidelines:

   - The answer NA 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 adheres to the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA 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 paper includes a section discussing potential positive impacts.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA 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., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: The paper poses no such risks.

    Guidelines:

    - The answer NA 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: The paper properly credits the creators and original owners of the assets used, and the licenses and terms of use are explicitly mentioned and respected.

    Guidelines:

    - The answer NA 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: [NA]

    Justification: This paper does not release new assets

    Guidelines:

    - The answer NA 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: [NA]

    Justification: The paper does not include experiments and research with human subjects.

Guidelines:

- The answer NA 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: [Yes]

    Justification: The paper describes the potential risks to study participants, confirms that these risks were disclosed, and states that IRB approval was obtained for the user study.

    Guidelines:

    - The answer NA 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 rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

    Guidelines:

    - The answer NA 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 (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.