GaRA-SAM: Robustifying Segment Anything Model with Gated-Rank Adaptation

Sohyun Lee[1] **Yeho Gwon**[1] **Lukas Hoyer**[2] **Suha Kwak**[1]

[1]POSTECH [2]Google

Abstract

Improving robustness of the Segment Anything Model (SAM) to input degradations is critical for its deployment in high-stakes applications such as autonomous driving and robotics. Our approach to this challenge prioritizes three key aspects: first, parameter efficiency to maintain the inherent generalization capability of SAM; second, fine-grained and input-aware robustification to precisely address the input corruption; and third, adherence to standard training protocols for ease of training. To this end, we propose gated-rank adaptation (GaRA). GaRA introduces lightweight adapters into intermediate layers of the frozen SAM, where each adapter dynamically adjusts the effective rank of its weight matrix based on the input by selectively activating (rank-1) components of the matrix using a learned gating module. This adjustment enables fine-grained and input-aware robustification without compromising the generalization capability of SAM. Our model, GaRA-SAM, significantly outperforms prior work on all robust segmentation benchmarks. In particular, it surpasses the previous best IoU score by up to 21.3%p on ACDC, a challenging real corrupted image dataset.

1 Introduction

The Segment Anything Model (SAM) [28] has proven to be a powerful tool for zero-shot image segmentation, exhibiting impressive generalization across unseen objects and images without additional training. Nevertheless, its performance deteriorates considerably when faced with degraded input due to noise, blur, low illumination, and adverse weather [8], as shown in Figure 1(b). This limitation significantly restricts its use in high-stakes applications such as autonomous driving and robotics.

A seemingly straightforward approach to improving the robustness of SAM is to attach an existing image restoration module to the front of SAM. However, this typically introduces significant computational overhead, and often yields suboptimal segmentation performance since image restoration is optimized to enhance the perceptual quality of images, rather than to improve performance of segmentation models like SAM [7, 12, 30, 33, 52, 57]. An alternative strategy involves fine-tuning SAM entirely on degraded inputs, which, however, demands substantial computational resources and diminishes its inherent zero-shot generalization capability [8].

Chen *et al.* [8] addressed these limitations by introducing anti-degradation modules into SAM; these modules are trained to refine degraded features by promoting feature-level consistency between a pair of clean and corrupted images of the same content. Despite effectively enhancing robustness, their method, dubbed RobustSAM, has a couple of limitations. First, its requirement for paired clean and degraded images, which cannot be captured by typical cameras, necessitates the use of synthetic degradations in training, hindering its generalization to real-world degradations. Second, since RobustSAM aims at learning representations invariant to various degradations, it struggles to produce representations adapted to the specific degradation affecting the input at hand, restricting

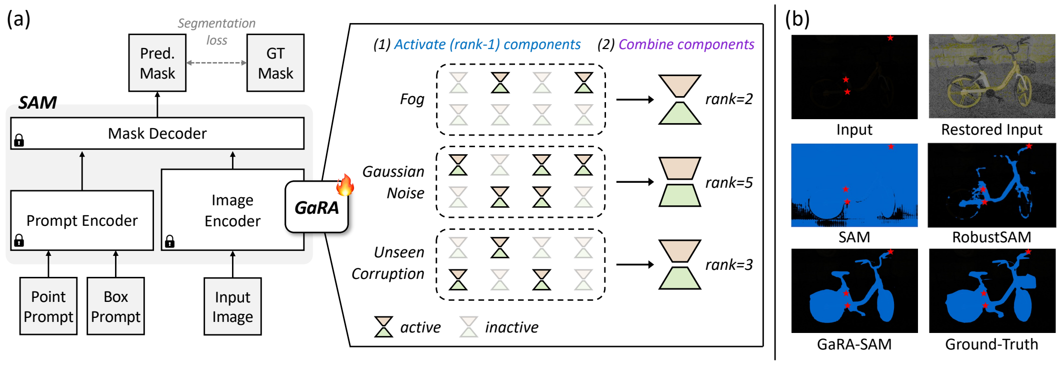

Figure 1: Overview and example results of GaRA-SAM. (a) Conceptual illustration of GaRA-SAM. (b) Example results on a real low-light image [5]. GaRA-SAM produces an accurate mask while the original SAM fails. Note that the restored input is provided for illustrative purposes only; GaRA-SAM does not perform image restoration.

further performance improvement. These limitations reduce the effectiveness of RobustSAM in real-world scenarios, where inputs can be degraded by real and previously unseen corruptions.

We found that low-rank adaptation (LoRA) [20] offers a promising foundation for overcoming many of the previously discussed limitations. By introducing and training lightweight adapters within the frozen SAM, LoRA improves the robustness parameter-efficiently while preserving the generalization ability of SAM. Our empirical analysis confirms that SAM incorporating LoRA achieves impressive robustness without using paired clean and degraded images for training, thus establishing a strong baseline as is. Nevertheless, our analysis also reveals a key limitation: the fixed rank of the adapters hinders its effective adaptation to a wide range of corruptions. We found a significant variation in the optimal rank of the adapters between different corruption types and inputs, underscoring the need for input-adaptive rank modulation.

Based on this observation, we introduce *gated-rank adaptation* (GaRA), which is illustrated in Figure 1(a), a novel method designed to enhance the robustness of SAM while addressing the aforementioned limitations. GaRA dynamically adjusts the effective rank of the weight matrix of each adapter, while being parameter-efficient and not demanding clean-degraded image pairs for training. Specifically, GaRA decomposes the weight matrix of an adapter into (rank-1) components and chooses a proper subset of them dynamically according to the input. To achieve this, we introduce a gating module that predicts a binary vector selectively activating the most appropriate components for the input. This mechanism allows GaRA to flexibly control both the number and combination of active components based on the input without any test-time optimization. Consequently, our zero-shot segmentation model integrating GaRA with SAM, named GaRA-SAM, achieves fine-grained and input-aware robustification while remaining parameter-efficient.

GaRA-SAM achieves the state of the art on multiple robust segmentation benchmarks [5, 9, 15, 40, 47, 51, 55, 61], including both synthetic and real-world corruption datasets. Importantly, and in contrast to prior work, the design of GaRA-SAM enables training using real-world degraded images lacking clean counterparts, leading to notable performance improvement on real-world corruption benchmarks. The main contribution of this work is three-fold:

- Our extensive analysis reveals the surprising effectiveness of LoRA in robustifying SAM, with its optimal rank varying significantly across different corruption types and individual images. These findings suggest a new research avenue for improving robustness of vision foundation models.
- We propose GaRA, a novel method to achieve robust SAM. At its core lies a lightweight and input-dependent adapter that enables fine-grained and parameter-efficient robustification without compromising the generalization capability of SAM. Also, GaRA does not require paired clean and degraded images for training and thus, unlike previous work [8], it can be learned using real degraded images without their clean references.

- Our final model, GaRA-SAM, achieves the state of the art across multiple robust segmentation benchmarks. Notably, it significantly surpasses the previous best IoU score by up to 21.3%p on ACDC, a challenging real-world corrupted image dataset.

2 Related Work

Robust Segment Anything. SAM [28] accepts free-form prompts along with an image to produce relevant masks, showing superior zero-shot generalizability. Despite its success, its robustness against visual corruptions is questionable [8, 48]. To improve the robustness of SAM, RobustSAM [8] adopts anti-degradation modules to approximate features of clean images. However, it requires clean-degraded image pairs for training, which are often unavailable in real-world scenarios. Improving the robustness of vision models with image restoration [1, 32, 43, 46, 52] and degradation-specific techniques [3, 5, 30, 44] has also been investigated. AirNet [32] and URIE [52] are universal image restoration models, but they introduce heavy computational overhead and AirNet targets a better image quality, not improving performance of visual perception models. LoRA-IR [1], DA-CLIP [43], and PromptIR [46] tackle this by encoding degradation-specific information into prompts. However, they rely on additional information such as degradation types and textual descriptions, and suffer from their complex modules and training schemes. Meanwhile, FIFO [30] and FreD [3] focus on extracting fog-invariant features, making their models robust to only foggy scenes. In contrast to these prior arts, GaRA is an efficient yet effective approach to fine-tuning SAM for improving its robustness without demanding clean references of degraded images for training.

Low-rank Adaptation. Fine-tuning large-scale pre-trained models introduces intensive overheads in space and time. To overcome this, various parameter-efficient learning schemes such as prompt tuning [23, 31, 35, 50] and adapter-based fine-tuning [6, 11, 16, 18, 19, 26, 36, 45, 60] have emerged. Among these, LoRA [19] leverages trainable low-rank matrices to catch up with full fine-tuning while introducing marginal extra learnable parameters. However, fixed ranks of LoRA yield limitations such as poor generalization in certain tasks [56, 59] and inefficient parameter allocation [4, 63]. Despite the efforts of previous work to update the ranks during training dynamically [39, 56, 63], they still use fixed ranks at test time and require handcrafted rank selection. In contrast, GaRA learns a small module that selects and combines appropriate (rank-1) components of a LoRA block, adapting to individual samples with various visual corruptions even during inference.

Mixture of Experts. While model scaling has been known as an effective and promising way of constructing a powerful model, training such a model on large-scale datasets [2, 17, 24, 29, 54, 62] is challenging due to high computational requirements [14, 49]. To this end, mixture of experts (MoE), which adopts multiple submodules and considers each as an expert, has gained prominence [14, 21, 38, 49, 58, 64]. These submodules are active or inactive using a learnable gating module at both training and test time, resulting in efficient use of resources and improved training stability. Switch Transformer [14] sparsely simplifies the MoE paradigm and proposes to select experts sparsely. AdaMix [58] injects an MoE adapter consisting of various up- and down-sampling layers into each transformer layer, both improving parameter-efficiency and performance. Motivated by the prior work, GaRA first decomposes the low-rank matrices of LoRA into (rank-1) components and considers each as an expert. Then, a learnable gating module sparsely selects a combination of (rank-1) components in an input-adaptive manner, enabling adaptation on a per-input basis.

3 Proposed Method

We first describe LoRA as a strong baseline for robustifying SAM in Sec. 3.1, and analyze the impact of the rank of its adapters on the robustness of SAM to verify our motivation of input-adaptive rank modulation in Sec. 3.2. Finally, Sec. 3.3 details GaRA that we integrate into SAM as GaRA-SAM.

3.1 Foundation: Low-rank Adaptation for Robustifying SAM

LoRA [20] is a parameter-efficient adaptation method that imbues a frozen pretrained model with adaptability using a handful of trainable parameters. It freezes a pretrained weight matrix $\mathbf{W}_0 \in \mathbb{R}^{D\times K}$ and introduces trainable low-rank update $\Delta\mathbf{W} = \mathbf{B}\mathbf{A}$, where $\mathbf{B} \in \mathbb{R}^{D\times R}$ and $\mathbf{A} \in \mathbb{R}^{R\times K}$ with rank $R \ll \min(D, K)$. The adapted forward pass becomes:

$$\mathbf{h} = \mathbf{W}_0\mathbf{x} + \mathbf{B}\mathbf{A}\mathbf{x}, \tag{1}$$

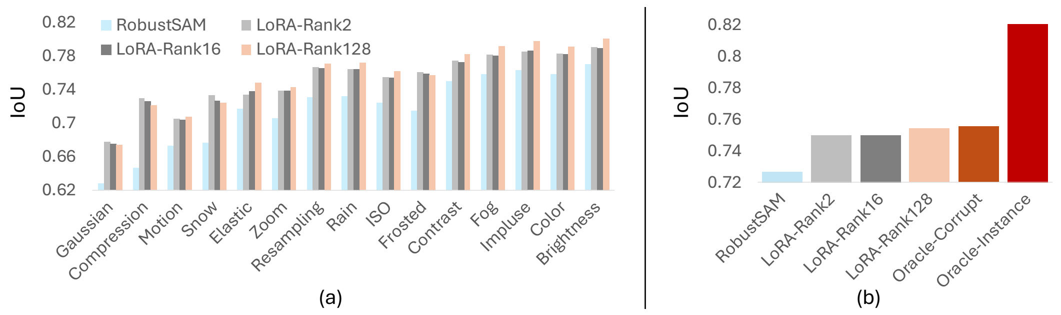

Figure 2: Impact of the rank in LoRA integrated with SAM. The models were evaluated on LVIS [15] using point prompts. (a) Performance versus rank under various corruption types. The best rank varies depending on the corruption type. (b) Comparisons of rank selection strategies. Oracle-Corrupt chooses the best rank per corruption type, while Oracle-Instance selects the best rank per image. The outstanding performance of Oracle-Instance suggests the need for input-adaptive rank manipulation.

where $\mathbf{x} \in \mathbb{R}^K$ is the input and $\mathbf{h} \in \mathbb{R}^D$ is the output.

Given its original purpose of parameter-efficient adaptation without compromising generalization, we argue that LoRA offers a reasonable baseline for improving robustness of SAM against input degradation. To validate this potential empirically, we evaluated SAM integrated with LoRA for robust segmentation using point prompts following an evaluation protocol of RobustSAM [8]. Specifically, we froze the original weights of SAM, attached low-rank adapters $\Delta\mathbf{W}$ to the key, query, and value projection layers of its image encoder, and trained the adapters with the standard segmentation loss on degraded images. As demonstrated in Figure 2, SAM with LoRA clearly outperformed RobustSAM [8] on the LVIS dataset [15], although it does not require clean reference images and the auxiliary loss the RobustSAM demands.

3.2 Analysis on the Impact of Rank on Robustness

Since different degradations distort different aspects of the input, the level of representation capacity needed for robustifying SAM varies accordingly, motivating the investigation into the role of LoRA's rank. To analyze this, we first evaluated its performance across various input corruptions while varying the rank. The results in Figure 2(a) suggest that the optimal rank varies depending on the corruption type. We conjecture that more pronounced corruptions lead to more substantial contamination of the semantic content of the input, and lower-rank adapters are more effective in such conditions since their restricted capacity forces them to prioritize more essential features for segmentation [42].

To further verify this conclusion, we evaluated SAM with LoRA under two *oracle* rank modulation scenarios: selecting the best rank per corruption type (Oracle-Corrupt) and per image (Oracle-Instance) using ground-truth. Specifically, we first computed IoU scores for all candidate ranks and selected only the top-performing rank for each corruption type or each image. While Oracle-Corrupt yielded moderate gains over fixed ranks, Oracle-Instance achieved significantly higher performance, highlighting the substantial potential of fine-grained, input-adaptive rank selection. These findings motivate our design of GaRA, which dynamically adjusts the rank per input.

3.3 Gated-rank Adaptation

Our solution for robustifying SAM, termed gated-rank adaptation (GaRA), is designed with three key properties in mind: parameter efficiency to preserve the generalization ability of SAM, fine-grained and input-aware robustification, and adherence to standard segmentation learning protocols for ease of training. These properties are realized by a novel, lightweight adapter that dynamically adjusts itself based on the input corruption; the architecture of the adapter is depicted in Figure 3. Our final model, GaRA-SAM, is constructed by integrating these adapters into the key, query, and value projection layers of the image encoder of SAM, keeping the original SAM weights frozen. Also, GaRA-SAM is trained solely with the standard segmentation loss computed from degraded images, without auxiliary learning objectives or paired clean references.

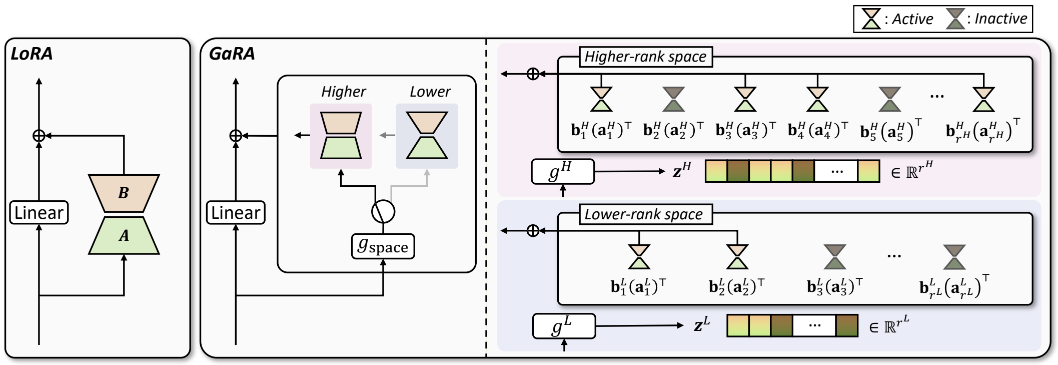
: Active
: Inactive
LoRA
Linear
B
A
GaRA
Linear
Higher
Lower
g_{space}
Higher-rank space
$\mathbf{b}_1^H(\mathbf{a}_1^H)^\top$ $\mathbf{b}_2^H(\mathbf{a}_2^H)^\top$ $\mathbf{b}_3^H(\mathbf{a}_3^H)^\top$ $\mathbf{b}_4^H(\mathbf{a}_4^H)^\top$ $\mathbf{b}_5^H(\mathbf{a}_5^H)^\top$ $\mathbf{b}_{r^H}^H(\mathbf{a}_{r^H}^H)^\top$
g^H
$\mathbf{z}^H$
$\in \mathbb{R}^{r^H}$
Lower-rank space
$\mathbf{b}_1^L(\mathbf{a}_1^L)^\top$ $\mathbf{b}_2^L(\mathbf{a}_2^L)^\top$ $\mathbf{b}_3^L(\mathbf{a}_3^L)^\top$ $\mathbf{b}_{r^L}^L(\mathbf{a}_{r^L}^L)^\top$
g^L
$\mathbf{z}^L$
$\in \mathbb{R}^{r^L}$

Figure 3: Adapter architecture of GaRA and comparison to LoRA. GaRA leverages hierarchical gating for both coarse and fine control over the adaptation process. First, g_{space} selects between the higher-rank and lower-rank spaces based on the input. Then, the corresponding gating module g^H or g^L predicts a binary vector $\mathbf{z}^H$ or $\mathbf{z}^L$ to activate a subset of (rank-1) components tailored to the input. These active components are composed to form the final update matrix for adaptation.

GaRA introduces a gating strategy to modulate the adapter's rank based on the input. To this end, we reinterprete the update matrix of LoRA, *i.e.*, $\Delta\mathbf{W} = \mathbf{BA}$, as a composition of R (rank-1) components: $\Delta\mathbf{W} = \sum_{i=1}^{R} \mathbf{b}_i\mathbf{a}_i^\top$, where $\mathbf{b}_i \in \mathbb{R}^D$ and $\mathbf{a}_i \in \mathbb{R}^K$ are the i-th column and row of $\mathbf{B}$ and $\mathbf{A}$, respectively. We assume that R is large enough so that the model has access to a rich set of (rank-1) components, enabling GaRA to flexibly choose a relevant subset depending on the input.

GaRA employs a hierarchical gating strategy that provides both coarse and fine control over the adaptation process. The adapter first coarsely selects between lower- and higher-rank spaces based on the input, and then, within the selected rank space, the gating module finely determines the rank of the adapter by activating a subset of the (rank-1) components appropriate for the input. The coarse rank space selection separates the adapter's components into two exclusive sets, tailored to different degraded inputs demanding different representation capacities, to mitigate potential conflicts between the sets and further improve performance. Meanwhile, the fine-grained gating enables flexible and input-specific composition of the update matrix, supporting efficient and expressive adaptation to diverse corruptions. Below we elaborate on this gating process.

Gating 1: Rank Space Selection. We explicitly divide the adapter into two distinct rank spaces, a lower-rank set $\{\mathbf{a}_i^L, \mathbf{b}_i^L\}_{i=1}^{r_L}$ and a higher-rank set $\{\mathbf{a}_j^H, \mathbf{b}_j^H\}_{j=1}^{r_H}$ with $r_L < r_H \ll K$, where K denotes the input feature dimension, and r_L and r_H indicate the maximal ranks of the lower-rank and higher-rank spaces, respectively.[1] To choose between these rank spaces, we employ a binary gating module g_{space}, which takes as input the intermediate feature $f(\mathbf{x})$ computed from $\mathbf{x}$ and outputs a binary gating variable $z_{\text{space}} \in \{0, 1\}$, where 0 and 1 indicate the lower- and higher-rank space, respectively. This module consists of a two-layer MLP followed by the Gumbel-Sigmoid, allowing differentiable binary decisions during training. The gating process is formally expressed as:

$$\alpha_{\text{space}} = \text{MLP}_{\text{space}}(f(\mathbf{x})), \tag{2}$$

where α_{space} is the gating logit computed from the input feature. To enable backpropagation despite the binary nature of the binary gating variable z_{space}, we apply the Gumbel-Sigmoid relaxation [22]:

$$\tilde{z}_{\text{space}} = \sigma\left(\frac{1}{\tau}(\alpha_{\text{space}} + G)\right), \tag{3}$$

where $G \sim \text{Gumbel}(0, 1)$ is a noise sampled during training, σ is the sigmoid function, and τ is a temperature parameter controlling the sharpness of the sigmoid. During training, we apply hard thresholding in the forward pass as $z_{\text{space}} = \mathbb{I}[\tilde{z}_{\text{space}} > 0.5]$ while using the continuous value $\tilde{z}_{\text{space}}$ to compute gradients in the backward pass. At test time, we discard the noise and compute the gate: $z_{\text{space}} = \mathbb{I}[\sigma(\alpha_{\text{space}}) > 0.5]$.

[1]We use the terms 'lower-rank' and 'higher-rank' in a relative sense. Both r_L and r_H are substantially smaller than the full feature dimension K, and thus still lie in the low-rank regime.

Gating 2: (rank-1) Component Selection. Within the selected rank space, the associated gating module predicts a binary vector identifying (rank-1) components suitable for the input:

$$\mathbf{z}^L = g^L(f(\mathbf{x})) \in \{0,1\}^{r_L}, \quad \mathbf{z}^H = g^H(f(\mathbf{x})) \in \{0,1\}^{r_H}. \tag{4}$$

where $g^L(\cdot)$ and $g^H(\cdot)$ denote the gating modules for the lower-rank and higher-rank spaces, respectively. Each gating module consists of a three-layer MLP whose final output is a real-valued logit vector, followed by Gumbel-Sigmoid operations to obtain binary masks:

$$\boldsymbol{\alpha}^L = \text{MLP}_L(f(\mathbf{x})), \quad \boldsymbol{\alpha}^H = \text{MLP}_H(f(\mathbf{x})). \tag{5}$$

To enable gradient-based learning, we apply the Gumbel-Sigmoid relaxation to each element of the logit vectors [22]:

$$\tilde{z}_i = \sigma\left(\frac{1}{\tau}(\alpha_i + G_i)\right), \quad i = 1, \ldots, r_L \text{ or } r_H, \tag{6}$$

where α_i is the i-th logit from $\boldsymbol{\alpha}^L$ or $\boldsymbol{\alpha}^H$, $G_i \sim \text{Gumbel}(0,1)$ is the corresponding noise sample, τ is a temperature parameter, and σ is the sigmoid function. During training, hard thresholding is applied in the forward pass to produce binary decisions ($z_i = \mathbb{I}[\tilde{z}_i > 0.5]$), while the continuous $\tilde{z}_i$ is used for backpropagation. At inference time, the gating becomes deterministic: $z_i = \mathbb{I}[\sigma(\alpha_i) > 0.5]$. We apply this relaxation to the gating functions g^L and g^H to enable learnable and input-adaptive binary decisions for the selection of (rank-1) components. This gating mechanism results in a dynamic, input-dependent adapter update:

$$\Delta\mathbf{W} = (1 - z_{\text{space}}) \cdot \sum_{i=1}^{r_L} z_i^L \mathbf{b}_i^L (\mathbf{a}_i^L)^\top + z_{\text{space}} \cdot \sum_{j=1}^{r_H} z_j^H \mathbf{b}_j^H (\mathbf{a}_j^H)^\top. \tag{7}$$

This design of GaRA enables flexible and conflict-free adaptation to a wide range of corruptions by coarse-to-fine rank modulation.

4 Experiments

4.1 Experimental Setting

Dataset. For training and validation, we utilize the Robust-Seg dataset [8], which is constructed by applying 15 types of synthetic corruptions to three semantic segmentation benchmarks: LVIS [15], MSRA-10K [9], and ThinObjects-5K [37], comprising a total of 26,000 masks. For evaluation, we use five clear-condition image segmentation benchmarks: LVIS, MSRA-10K, STREETS [51], NDD20 [55], COCO [40]. Also, we test on a real-world corrupted benchmark including BDD-100K [61] and LIS [5]. For training GaRA-SAM on real-world data, we use BDD-100K and LIS for training, and evaluate on BDD-100K and LIS, and ACDC [47]. Note that all these datasets are free from licensing issues. For brevity, we refer to the union of BDD-100K and LIS as BDD+LIS, and the union of STREETS and NDD20 as STREETS+NDD. More details are given in the appendix.

Experimental Details. We adopt the ViT-B and ViT-L variants of SAM [13], and freeze their parameters during training the GaRA modules. The models are optimized by Adam [27] with a learning rate of 1×10^{-4} for ViT-B and 1×10^{-5} for ViT-L, a weight decay of 1×10^{-5}, and input batches of size 8, using both point and box prompts. The gating modules are trained with the same learning rate. We set the lower- and higher-rank dimensions as $r_L = 16$ and $r_H = 256$, respectively, and use a Gumbel-Sigmoid temperature of 0.5. Since no official evaluation code is provided by previous work, we reproduce and evaluate its models using the same protocol. All training and evaluation experiments were conducted at POSTECH.

4.2 Comparison on Seen Dataset

To assess the robustness of GaRA-SAM, we first evaluate its performance on synthetic corruptions applied to seen datasets such as LVIS and MSRA-10K. The results in Table 1 show that GaRA-SAM outperforms prior methods, including RobustSAM [8], HQ-SAM [25], and restoration-based methods (*e.g.*, AirNet [32] and URIE [52]), when using both point and box prompts in both datasets. Notably, our model demonstrates significant improvements in the degraded setting, *e.g.*, surpassing the previous best by up to 4.3%p on LVIS without compromising performance on clean images.

Table 1: Segmentation results on LVIS and MSRA using point and box prompts.

Backbone	Method	LVIS								MSRA							
		Point Prompts				Box Prompts				Point Prompts				Box Prompts			
		Degrade		Clear		Degrade		Clear		Degrade		Clear		Degrade		Clear	
		IoU	Dice	IoU	Dice	IoU	Dice	IoU	Dice	IoU	Dice	IoU	Dice	IoU	Dice	IoU	Dice
ViT-B	SAM	65.0	76.3	70.1	80.1	75.5	84.5	78.4	86.1	75.9	84.5	79.1	86.6	86.6	92.2	88.7	93.4
	HQ-SAM	69.5	80.1	76.0	84.8	79.0	87.1	83.1	89.7	82.8	89.6	86.6	92.0	89.7	94.3	92.4	95.9
	AirNet+SAM	64.8	76.1	70.1	80.1	75.4	84.4	78.3	86.0	75.7	84.3	79.1	86.6	86.4	92.1	88.8	93.4
	URIE+SAM	64.8	76.2	70.0	80.0	74.5	83.7	78.0	85.8	74.6	83.5	77.6	85.6	85.9	91.8	88.8	93.5
	RobustSAM	72.7	82.6	77.2	85.8	81.5	89.0	84.3	90.7	86.6	92.3	89.6	94.2	89.5	94.2	92.1	95.7
	GaRA-SAM	**77.0**	**85.7**	**81.3**	**88.7**	**83.7**	**90.5**	**86.1**	**91.9**	**89.4**	**94.0**	**91.4**	**95.2**	**92.3**	**95.8**	**93.9**	**96.8**
ViT-L	SAM	66.2	76.0	75.0	83.0	79.9	87.7	82.8	89.4	77.3	84.6	82.0	88.1	87.6	92.9	88.9	93.5
	HQ-SAM	72.6	82.1	79.0	86.7	81.1	88.6	84.7	90.8	85.0	91.0	87.6	92.6	89.4	94.1	91.4	95.2
	AirNet+SAM	66.0	75.7	74.8	82.8	79.7	87.6	82.7	89.4	76.9	84.3	81.7	87.8	87.5	92.8	89.0	93.6
	URIE+SAM	66.4	76.3	74.8	83.0	79.4	87.4	82.7	89.4	78.1	85.4	83.0	88.9	87.6	92.8	89.6	94.0
	RobustSAM	75.6	84.5	80.0	87.5	83.6	90.3	85.8	91.6	87.6	92.9	90.1	94.4	91.6	95.4	93.6	96.5
	GaRA-SAM	**78.5**	**86.6**	**82.6**	**89.4**	**84.4**	**90.8**	**86.7**	**92.2**	**89.6**	**94.1**	**91.6**	**95.3**	**92.6**	**96.0**	**94.0**	**96.8**

Table 2: Zero-shot segmentation results on COCO and STREETS+NDD using point and box prompts.

Backbone	Method	COCO								STREETS+NDD							
		Point Prompts				Box Prompts				Point Prompts				Box Prompts			
		Degrade		Clear		Degrade		Clear		Degrade		Clear		Degrade		Clear	
		IoU	Dice	IoU	Dice	IoU	Dice	IoU	Dice	IoU	Dice	IoU	Dice	IoU	Dice	IoU	Dice
ViT-B	SAM	65.2	76.3	69.6	79.6	76.0	84.8	78.9	86.4	74.4	83.5	81.8	89.1	80.2	88.3	85.4	91.8
	HQ-SAM	70.2	80.6	76.0	84.8	79.6	87.5	83.5	90.1	75.4	84.5	82.2	89.5	80.8	88.7	86.2	92.3
	AirNet+SAM	65.0	76.1	69.5	79.5	75.9	84.7	78.8	86.3	74.3	83.4	81.8	89.1	80.1	88.2	85.4	91.8
	URIE+SAM	65.0	76.2	69.6	79.6	75.0	84.0	78.2	85.8	74.1	83.4	81.0	88.6	79.8	88.0	84.9	91.5
	RobustSAM	72.7	82.6	77.1	85.7	81.6	89.1	84.5	90.9	74.5	84.1	81.0	88.8	81.8	89.4	86.2	92.4
	GaRA-SAM	**77.2**	**85.8**	**81.0**	**88.4**	**84.2**	**90.8**	**86.4**	**92.1**	**77.6**	**86.4**	**82.9**	**90.1**	**84.1**	**91.0**	**87.7**	**93.3**
ViT-L	SAM	66.9	76.5	74.6	82.5	80.4	88.0	82.9	89.4	72.9	81.3	82.1	89.0	81.5	89.2	86.4	92.5
	HQ-SAM	73.2	82.5	79.0	86.7	81.7	89.0	85.1	91.1	76.8	85.4	83.9	90.6	81.7	89.3	86.7	92.6
	AirNet+SAM	66.6	76.3	74.4	82.4	80.3	87.9	82.8	89.3	72.5	81.0	81.9	88.7	81.4	89.1	86.4	92.5
	URIE+SAM	66.6	76.4	74.3	82.4	79.9	87.7	82.8	89.4	72.6	81.4	81.2	88.4	81.1	88.9	86.1	92.2
	RobustSAM	75.3	84.3	79.9	87.6	83.8	90.5	86.2	91.9	75.7	84.8	82.7	89.9	83.0	90.2	87.5	93.1
	GaRA-SAM	**78.9**	**86.9**	**82.3**	**89.1**	**84.9**	**91.2**	**86.9**	**92.4**	**80.0**	**88.0**	**85.3**	**91.7**	**85.3**	**91.7**	**88.7**	**93.9**

4.3 Zero-shot Segmentation Comparison

To further assess the generalization capability of GaRA-SAM, we evaluate it under both synthetic and real corruptions on datasets not seen during training. Specifically, we include COCO and STREETS+NDD with synthetic degradations, and BDD+LIS, which exhibits real corruptions such as fog, rain, motion blur, and low-light. As reported in Table 2, GaRA-SAM consistently achieves the best across all prompt types and metrics in both COCO and STREETS+NDD. Table 3 shows that it also outperforms previous work on real corrupted images, improving over RobustSAM by more than 3.4%p IoU with point prompts in ViT-L, despite being trained only on unpaired corrupted images. It demonstrates that GaRA-SAM generalizes well to both synthetic and real-world corruptions, highlighting the effectiveness of its input-adaptive design.

4.4 Training on Real Corruption Datasets

GaRA-SAM enables training on real corruption datasets without requiring access to clean references. To demonstrate this, we train GaRA-SAM solely on the real-world dataset, BDD+LIS, and evaluate its performance on both the seen dataset (BDD+LIS) and an unseen real-world dataset, ACDC. As shown in Table 4, training directly on real corrupted images significantly improves the performance on all benchmarks, outperforming the previous best by up to 21.3%p IoU on ACDC. In addition, we compare GaRA-SAM trained on BDD+LIS (GaRA-SAM-Real) and that trained on the synthetic Robust-Seg dataset (GaRA-SAM-Syn), using box prompts and ViT-L in Table 3; the results suggest the clear benefit of training directly on real corrupted images.

Table 3: Evaluation results on a real degraded image dataset, BDD+LIS, using point and box prompts.

Backbone	Method	Point Prompts		Box Prompts	
		IoU	Dice	IoU	Dice
ViT-B	SAM	65.2	75.4	74.2	82.7
	HQ-SAM	67.6	77.9	69.4	78.0
	AirNet+SAM	64.1	74.5	73.4	82.1
	URIE+SAM	65.7	76.1	74.1	82.8
	RobustSAM	<u>69.5</u>	<u>79.5</u>	<u>75.7</u>	<u>84.1</u>
	GaRA-SAM	**71.3**	**80.9**	**76.8**	**85.3**
ViT-L	SAM	68.9	77.8	74.3	82.0
	HQ-SAM	<u>72.7</u>	<u>81.7</u>	76.3	84.3
	AirNet+SAM	67.2	76.4	73.1	81.3
	URIE+SAM	69.7	78.8	74.4	82.1
	RobustSAM	71.4	80.9	<u>78.1</u>	<u>86.1</u>
	GaRA-SAM	**74.8**	**83.4**	**80.0**	**87.4**

Table 4: Evaluation results on BDD+LIS and ACDC, using a box prompt with ViT-L. GaRA-SAM-Syn and -Real are trained on Robust-Seg and BDD+LIS, respectively.

Method	BDD+LIS		ACDC	
	IoU	Dice	IoU	Dice
SAM	74.3	82.0	65.9	72.6
RobustSAM	78.1	86.1	67.5	77.0
GaRA-SAM-Syn	<u>80.0</u>	<u>87.4</u>	<u>71.6</u>	<u>80.4</u>
GaRA-SAM-Real	**89.7**	**93.9**	**88.8**	**92.9**

Table 5: Comparison of our GaRA and LoRA on LVIS with ViT-B, using point prompts.

Method		Degraded		Clear	
		IoU	Dice	IoU	Dice
LoRA	Rank 16	75.0	84.3	79.4	87.3
	Rank 128	<u>75.6</u>	<u>84.6</u>	<u>80.6</u>	<u>88.1</u>
	Rank 256	73.4	83.1	78.5	86.6
GaRA-SAM		**77.0**	**85.7**	**81.3**	**88.7**

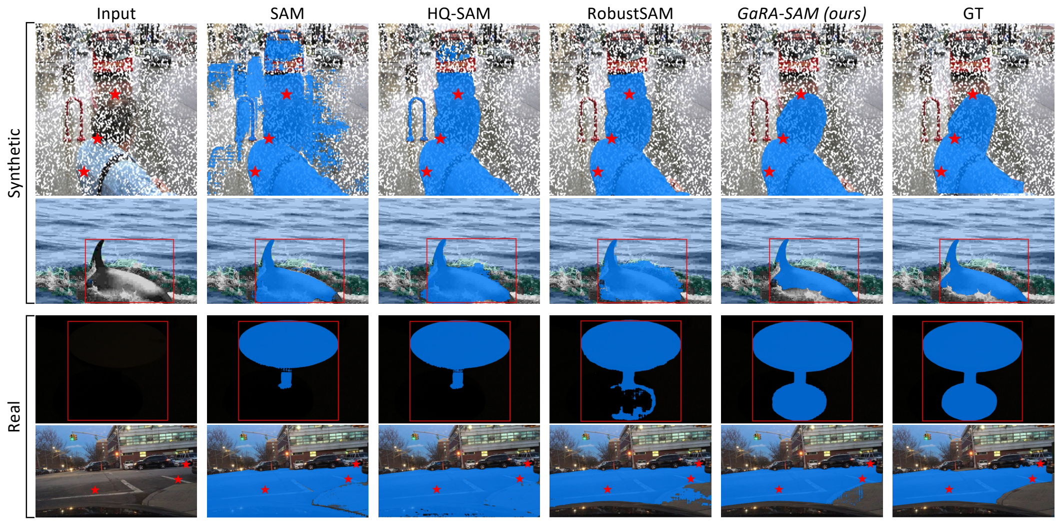

Figure 4: Results on synthetic (COCO, STREETS+NDD) and real corruption (BDD+LIS) datasets.

4.5 Qualitative results

Figure 4 presents qualitative results under synthetic and real-world corruptions. GaRA-SAM delivers more accurate and complete masks than SAM, HQ-SAM, and RobustSAM, especially under severe degradations: it better preserves object boundaries in synthetic cases and is more reliable in low-light and adverse weather conditions.

4.6 In-depth analysis

We conduct a comprehensive ablation study to assess the contribution of each design choice in GaRA-SAM. All experiments are performed on the LVIS dataset, using point prompts for consistency.

LoRA vs. GaRA. We first compare GaRA with the standard LoRA with fixed ranks. As shown in Table 5, GaRA-SAM consistently outperforms all fixed-rank LoRA. While LoRA establishes a strong baseline, its fixed-rank design limits flexibility. In contrast, GaRA-SAM dynamically composes (rank-1) components based on the input, resulting in improved robustness. These results validate the benefit of input-adaptive rank modulation over static alternatives.

Table 6: Comparison of GaRA and MoE LoRA.

Method	Degraded IoU	Degraded Dice	Clear IoU	Clear Dice
MoE LoRA (2, 16, 128, 256)	76.4	85.3	79.7	87.5
MoE LoRA (2, 16, 128, 1024)	75.4	84.5	78.6	86.7
GaRA-SAM	**77.0**	**85.7**	**81.3**	**88.7**

Table 7: Effect of the rank space separation.

Method		Degraded IoU	Degraded Dice	Clear IoU	Clear Dice
w/o Gating 1	Rank 16	76.2	85.2	79.9	87.7
	Rank 256	73.3	82.8	77.3	85.5
w/ Gating 1		**77.0**	**85.7**	**81.3**	**88.7**

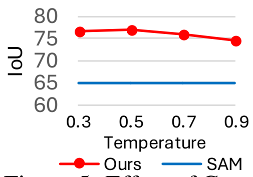

Figure 5: Effect of Gumbel temperature.

Table 8: Comparison of computational efficiency.

Method	Training: Learnable Params	Training: # GPU	Inference: GPU Memory	Inference: FPS
SAM	1250M	256	3.36GB	2.9
RobustSAM	403M	8	5.41GB	2.8
GaRA-SAM	343M	8	5.39GB	2.6

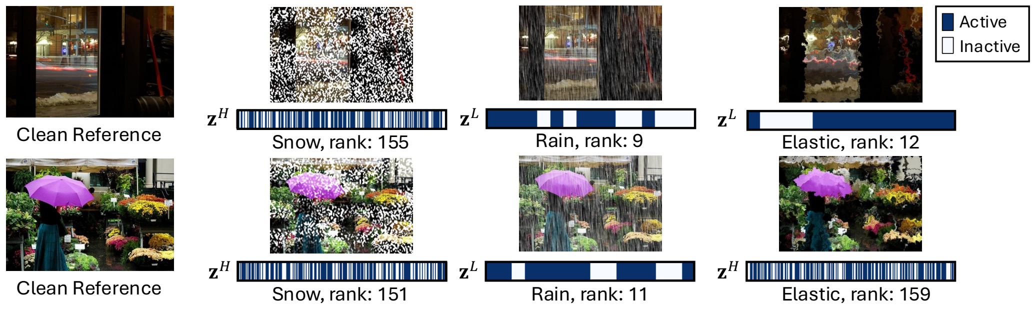

Figure 6: Visualization of the binary gating vectors and ranks under various images under corruptions.

MoE LoRA vs. GaRA. We also investigate an alternative approach to rank selection using a mixture-of-experts (MoE) [64] variant of LoRA, where multiple fixed rank LoRA blocks are instantiated, and a shared gating module selects one for each input. As reported in Table 6, both MoE variants perform reasonably well, but fall short of GaRA-SAM in both clean and degraded conditions. This gap is not trivial regarding the fewer number of parameters of GaRA-SAM.

Impact of Rank Space Selection. To evaluate the effectiveness of rank space selection (Gating 1), we compare GaRA-SAM with a variant that removes the separation between the lower- and higher-rank spaces, using a single unified set of (rank-1) components with a fixed maximum rank (*e.g.*, 16 or 256). As shown in Table 7, eliminating rank space selection results in performance degradation, suggesting that separating the low- and high-rank components allows more effective specialization.

Impact of Temperature in Gumbel-Sigmoid. We investigate the effect of the temperature of the Gumbel-Sigmoid by varying its value. Figure 5 summarizes the performance of variants of GaRA-SAM trained using different values of temperature; the optimal value for τ is 0.5. The results suggest that GaRA-SAM remains stable across a wide range of temperature values, suggesting that GaRA-SAM is insensitive to temperature settings.

Computational Cost Analysis. We compare the computational efficiency of GaRA-SAM in terms of learnable parameters, GPU resources, memory consumption, and inference speed (FPS). As summarized in Table 8, GaRA-SAM achieves the lowest number of learnable parameters and GPUs, owing to its parameter-efficient design. At inference time, it requires less GPU memory than RobustSAM, while maintaining comparable FPS to both SAM and RobustSAM.

Impact of Maximum Rank Configuration. We demonstrate the effect of our chosen maximum lower-rank and higher-rank configuration. Table 9 summarizes the performance of variants of GaRA-SAM using different maximum rank configurations. The optimal rank combination is {16, 256}, achieving the best performance.

Table 9: Effect of rank configuration.

Rank Config.	IoU
{8, 256}	76.3
{16, 128}	76.9
{16, 256}	**77.0**
{16, 512}	76.6

Table 10: Effect of gating strategy.

Gating Strategy	IoU
Random	73.9
Ours	**77.0**

Table 11: Effect of rank space selection.

Gating 1	Gating 2	IoU
		73.4
✓		76.0
✓	✓	**77.0**

Table 12: Effect of hierarchy depth.

Hierarchy Depth	IoU
2-level (Ours)	**77.0**
3-level	76.4
4-level	74.0

Analysis on Gating Vectors. Figure 6 demonstrates how the gating module responds to different combinations of corruption type and image. For each corrupted input, we present the corresponding binary activation vector $\mathbf{z}^H$ or $\mathbf{z}^L$ according to their selected rank space, along with the resulting number of active (rank-1) components. We observe that GaRA activates different (rank-1) components depending on the image contents as well as the corruption types. Even when the selected rank is similar, the constituent components often vary, highlighting the input-adaptive and fine-grained gating mechanism of GaRA.

Analysis on Gating in GaRA-SAM. To empirically demonstrate that the gating modules are learned to provide desirable gating paths even with such indirect supervision, we compared GaRA-SAM and its variant with random gating. Table 10 indicates that GaRA-SAM significantly outperforms the random gating variant, supporting the effectiveness of our training strategy.

Isolating Effect of Rank Space Selection. To isolate the effect of rank space selection, we experimented with rank space selection alone with fixed rank settings, disabling (rank-1) component selection. Table 11 demonstrate that rank space separation alone significantly improves performance. We believe this is because it enables each rank space to specialize in handling different degraded inputs, allowing the gating to select the most suitable representational capacity for each. Furthermore, enabling (rank-1) component selection on top further boosts the performance, highlighting its complementary benefit.

Impact of Depth of Hierachical Gating. We conducted additional experiments to assess deeper hierarchical gating structures (*i.e.*, 3-level and 4-level). As shown in Table. 12, increasing the depth introduces optimization challenges due to more complex gating decisions, ultimately degrading performance. Notably, our current 2-level hierarchy yields the best performance and offers the most favorable capacity and trainability.

5 Conclusion

In this paper, we introduce GaRA-SAM, a novel approach for robustifying Segment Anything Model (SAM) under diverse image degradations. Through extensive empirical analysis, we observed that the optimal LoRA rank varies significantly across corruption types and individual inputs, motivating our design of Gated-Rank Adaptation (GaRA), a lightweight and input-adaptive module that dynamically modulates the effective rank of LoRA adapters. GaRA operates without requiring paired clean and degraded images, enabling fine-grained and parameter-efficient adaptation while preserving SAM's inherent zero-shot generalization capabilities. GaRA-SAM achieves state-of-the-art performance across all robustness benchmarks and, notably, supports training directly on real-world corruption datasets without clean references. This leads to substantial gains in real-world scenarios, highlighting the practical utility and broad applicability of our method.

Limitations. While GaRA adaptively determines the appropriate rank for each rank space, several design choices, such as the split between lower- and higher-rank spaces, the predefined maximum ranks, and the fixed 2-level gating hierarchy, are set manually and thus limit flexibility. More generalized automated designs are conceptually desirable but practically challenging due to unstable optimization. Developing automated methods remains an important future direction.

Acknowledgement. This work was supported by Samsung Research Funding & Incubation Center of Samsung Electronics (SRFC-IT1801-52), AI Graduate School Program at POSTECH (RS-2019-II191906), and the IITP grants (RS-2024-00457882, RS-2024-00509258) funded by Ministry of Science and ICT, Korea.

References

[1] Yuang Ai, Huaibo Huang, and Ran He. Lora-ir: Taming low-rank experts for efficient all-in-one image restoration. *arXiv preprint arXiv:2410.15385*, 2024.

[2] Ibrahim Alabdulmohsin, Behnam Neyshabur, and Xiaohua Zhai. Revisiting neural scaling laws in language and vision. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL `https://openreview.net/forum?id=h3RYh6IBBS`.

[3] Qi Bi, Shaodi You, and Theo Gevers. Generalized foggy-scene semantic segmentation by frequency decoupling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 1389–1399, June 2024.

[4] Huandong Chang, Zicheng Ma, Mingyuan Ma, Zhenting Qi, Andrew Sabot, Hong Jiang, and HT Kung. Elalora: Elastic & learnable low-rank adaptation for efficient model fine-tuning. *arXiv preprint arXiv:2504.00254*, 2025.

[5] Linwei Chen, Ying Fu, Kaixuan Wei, Dezhi Zheng, and Felix Heide. Instance segmentation in the dark. *International Journal of Computer Vision*, 2023. URL `https://github.com/Linwei-Chen/LIS`.

[6] Shoufa Chen, Chongjian Ge, Zhan Tong, Jiangliu Wang, Yibing Song, Jue Wang, and Ping Luo. Adaptformer: Adapting vision transformers for scalable visual recognition. In *Proc. Neural Information Processing Systems (NeurIPS)*, 2022.

[7] Wei-Ting Chen, I-Hsiang Chen, Chih-Yuan Yeh, Hao-Hsiang Yang, Hua-En Chang, Jian-Jiun Ding, and Sy-Yen Kuo. Rvsl: Robust vehicle similarity learning in real hazy scenes based on semi-supervised learning. In *Proc. European Conference on Computer Vision (ECCV)*. Springer, 2022.

[8] Wei-Ting Chen, Yu-Jiet Vong, Sy-Yen Kuo, Sizhou Ma, and Jian Wang. RobustSAM: segment anything robustly on degraded images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4081–4091, 2024.

[9] Ming-Ming Cheng, Niloy J. Mitra, Xiaolei Huang, Philip H. S. Torr, and Shi-Min Hu. Global contrast based salient region detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2015. URL `https://mmcheng.net/msra10k/`.

[10] Noel CF Codella, David Gutman, M Emre Celebi, Brian Helba, Michael A Marchetti, Stephen W Dusza, Aadi Kalloo, Konstantinos Liopyris, Nabin Mishra, Harald Kittler, et al. Skin lesion analysis toward melanoma detection: A challenge at the 2017 international symposium on biomedical imaging (isbi), hosted by the international skin imaging collaboration (isic). In *2018 IEEE 15th international symposium on biomedical imaging (ISBI 2018)*, 2018.

[11] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. *arXiv preprint arXiv:2305.14314*, 2023.

[12] Steven Diamond, Vincent Sitzmann, Frank Julca-Aguilar, Stephen Boyd, Gordon Wetzstein, and Felix Heide. Dirty pixels: Towards end-to-end image processing and perception. *ACM Transactions on Graphics (TOG)*, 2021.

[13] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *Proc. International Conference on Learning Representations (ICLR)*, 2021.

[14] William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23 (120):1–39, 2022.

[15] Agrim Gupta, Piotr Dollar, and Ross Girshick. Lvis: A dataset for large vocabulary instance segmentation. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. URL `https://www.lvisdataset.org/`.

[16] Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. Towards a unified view of parameter-efficient transfer learning. *arXiv preprint arXiv:2110.04366*, 2021.

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[18] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *Proc. International Conference on Machine Learning (ICML)*. PMLR, 2019.

[19] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[20] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

[21] Changho Hwang, Wei Cui, Yifan Xiong, Ziyue Yang, Ze Liu, Han Hu, Zilong Wang, Rafael Salas, Jithin Jose, Prabhat Ram, HoYuen Chau, Peng Cheng, Fan Yang, Mao Yang, and Yongqiang Xiong. Tutel: Adaptive mixture-of-experts at scale. In D. Song, M. Carbin, and T. Chen, editors, *Proceedings of Machine Learning and Systems*, 2023.

[22] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

[23] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. In *European Conference on Computer Vision (ECCV)*, 2022.

[24] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

[25] Lei Ke, Mingqiao Ye, Martin Danelljan, Yifan Liu, Yu-Wing Tai, Chi-Keung Tang, and Fisher Yu. Segment anything in high quality. In *NeurIPS*, 2023.

[26] Sungyeon Kim, Boseung Jeong, Donghyun Kim, and Suha Kwak. Efficient and versatile robust fine-tuning of zero-shot models. In *European Conference on Computer Vision (ECCV)*, 2024.

[27] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[28] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4015–4026, 2023.

[29] Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. Big transfer (bit): General visual representation learning. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part V 16*, pages 491–507. Springer, 2020.

[30] Sohyun Lee, Taeyoung Son, and Suha Kwak. Fifo: Learning fog-invariant features for foggy scene segmentation. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[31] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 2021.

[32] Boyun Li, Xiao Liu, Peng Hu, Zhongqin Wu, Jiancheng Lv, and Xi Peng. All-In-One Image Restoration for Unknown Corruption. In *IEEE Conference on Computer Vision and Pattern Recognition*, New Orleans, LA, June 2022.

[33] Siyuan Li, Iago Breno Araujo, Wenqi Ren, Zhangyang Wang, Eric K Tokuda, Roberto Hirata Junior, Roberto Cesar-Junior, Jiawan Zhang, Xiaojie Guo, and Xiaochun Cao. Single image deraining: A comprehensive benchmark analysis. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[34] Xiang Li, Tianhan Wei, Yau Pun Chen, Yu-Wing Tai, and Chi-Keung Tang. Fss-1000: A 1000-class dataset for few-shot segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020.

[35] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021.

[36] Dongze Lian, Daquan Zhou, Jiashi Feng, and Xinchao Wang. Scaling & shifting your features: A new baseline for efficient model tuning. *Advances in Neural Information Processing Systems*, 35:109–123, 2022.

[37] Jun Hao Liew, Scott Cohen, Brian Price, Long Mai, and Jiashi Feng. Deep interactive thin object selection. In *Winter Conference on Applications of Computer Vision (WACV)*, 2021. URL `https://github.com/liewjunhao/thin-object-selection`.

[38] Bin Lin, Zhenyu Tang, Yang Ye, Jiaxi Cui, Bin Zhu, Peng Jin, Junwu Zhang, Munan Ning, and Li Yuan. Moe-llava: Mixture of experts for large vision-language models. *arXiv preprint arXiv:2401.15947*, 2024.

[39] Cheng Lin, Lujun Li, Dezhi Li, Jie Zou, Wei Xue, and Yike Guo. Nora: Nested low-rank adaptation for efficient fine-tuning large models. *arXiv preprint arXiv:2408.10280*, 2024.

[40] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Proc. European Conference on Computer Vision (ECCV)*. Springer, 2014. URL `https://cocodataset.org`.

[41] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, 2017.

[42] Jiaming Liu, Senqiao Yang, Peidong Jia, Ming Lu, Yandong Guo, Wei Xue, and Shanghang Zhang. Vida: Homeostatic visual domain adapter for continual test time adaptation. *arXiv preprint arXiv:2306.04344*, 2023.

[43] Ziwei Luo, Fredrik K. Gustafsson, Zheng Zhao, Jens Sjölund, and Thomas B. Schön. Controlling vision-language models for multi-task image restoration. In *The Twelfth International Conference on Learning Representations*, 2024. URL `https://openreview.net/forum?id=t3vnnLeajU`.

[44] Xianzheng Ma, Zhixiang Wang, Yacheng Zhan, Yinqiang Zheng, Zheng Wang, Dengxin Dai, and Chia-Wen Lin. Both style and fog matter: Cumulative domain adaptation for semantic foggy scene understanding. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 18922–18931, 2022.

[45] Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. Adapterfusion: Non-destructive task composition for transfer learning. *arXiv preprint arXiv:2005.00247*, 2020.

[46] Vaishnav Potlapalli, Syed Waqas Zamir, Salman Khan, and Fahad Khan. PromptIR: Prompting for all-in-one image restoration. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL `https://openreview.net/forum?id=KAlSIL4tXU`.

[47] Christos Sakaridis, Dengxin Dai, and Luc Van Gool. ACDC: The adverse conditions dataset with correspondences for semantic driving scene understanding. In *Proc. IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. URL `https://acdc.vision.ee.ethz.ch/`.

[48] Madeline Chantry Schiappa, Shehreen Azad, Sachidanand Vs, Yunhao Ge, Ondrej Miksik, Yogesh S Rawat, and Vibhav Vineet. Robustness analysis on foundational segmentation models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1786–1796, 2024.

[49] Noam Shazeer, *Azalia Mirhoseini, *Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations*, 2017. URL `https://openreview.net/forum?id=B1ckMDqlg`.

[50] Sheng Shen, Shijia Yang, Tianjun Zhang, Bohan Zhai, Joseph E Gonzalez, Kurt Keutzer, and Trevor Darrell. Multitask vision-language prompt tuning. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 5656–5667, 2024.

[51] Corey Snyder and Minh Do. Streets: A novel camera network dataset for traffic flow. *Advances in Neural Information Processing Systems*, 2019. URL `https://databank.illinois.edu/datasets/IDB-3671567`.

[52] Taeyoung Son, Juwon Kang, Namyup Kim, Sunghyun Cho, and Suha Kwak. Urie: Universal image enhancement for visual recognition in the wild. In *Proc. European Conference on Computer Vision (ECCV)*, 2020.

[53] Carole H Sudre, Wenqi Li, Tom Vercauteren, Sebastien Ourselin, and M Jorge Cardoso. Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations. In *Proc. Medical Image Computing and Computer-Assisted Intervention (MICCAI)*. Springer, 2017.

[54] Mingxing Tan and Quoc Le. EfficientNet: Rethinking model scaling for convolutional neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 6105–6114. PMLR, 09–15 Jun 2019. URL `https://proceedings.mlr.press/v97/tan19a.html`.

[55] Cameron Trotter, Georgia Atkinson, Matt Sharpe, Kirsten Richardson, A Stephen McGough, Nick Wright, Ben Burville, and Per Berggren. Ndd20: A large-scale few-shot dolphin dataset for coarse and fine-grained categorisation. *arXiv preprint arXiv:2005.13359*, 2020. URL `https://data.ncl.ac.uk/collections/The_Northumberland_Dolphin_Dataset_2020/4982342/1`.

[56] Mojtaba Valipour, Mehdi Rezagholizadeh, Ivan Kobyzev, and Ali Ghodsi. DyLoRA: Parameter-efficient tuning of pre-trained models using dynamic search-free low-rank adaptation. In Andreas Vlachos and Isabelle Augenstein, editors, *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 3274–3287, Dubrovnik, Croatia, May 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.eacl-main.239. URL `https://aclanthology.org/2023.eacl-main.239/`.

[57] Rosaura G Vidal, Sreya Banerjee, Klemen Grm, Vitomir Struc, and Walter J Scheirer. Ugˆ 2: A video benchmark for assessing the impact of image restoration and enhancement on automatic visual recognition. In *Proc. IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2018.

[58] Yaqing Wang, Sahaj Agarwal, Subhabrata Mukherjee, Xiaodong Liu, Jing Gao, Ahmed Hassan Awadallah, and Jianfeng Gao. AdaMix: Mixture-of-adaptations for parameter-efficient model tuning. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang, editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5744–5760, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.emnlp-main.388. URL `https://aclanthology.org/2022.emnlp-main.388/`.

[59] Wenhan Xia, Chengwei Qin, and Elad Hazan. Chain of lora: Efficient fine-tuning of language models via residual learning. *arXiv preprint arXiv:2401.04151*, 2024.

[60] Dongshuo Yin, Leiyi Hu, Bin Li, Youqun Zhang, and Xue Yang. 5%>100%: Breaking performance shackles of full fine-tuning on visual recognition tasks, 2024. URL `https://arxiv.org/abs/2408.08345`.

[61] Fisher Yu, Wenqi Xian, Yingying Chen, Fangchen Liu, Mike Liao, Vashisht Madhavan, Trevor Darrell, et al. Bdd100k: A diverse driving video database with scalable annotation tooling. *arXiv preprint arXiv:1805.04687*, 2018. URL `https://bdd-data.berkeley.edu/`.

[62] Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. Scaling vision transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12104–12113, 2022.

[63] Qingru Zhang, Minshuo Chen, Alexander Bukharin, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. Adaptive budget allocation for parameter-efficient fine-tuning. In *The Eleventh International Conference on Learning Representations*, 2023. URL `https://openreview.net/forum?id=lq62uWRJjiY`.

[64] Yanqi Zhou, Tao Lei, Hanxiao Liu, Nan Du, Yanping Huang, Vincent Zhao, Andrew M Dai, Quoc V Le, James Laudon, et al. Mixture-of-experts with expert choice routing. *Advances in Neural Information Processing Systems*, 35:7103–7114, 2022.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The abstract and introduction of our paper sufficiently reflect the paper's contributions and scope.

   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: We discuss the limitations of the work in Section 5.

   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: [NA]

Justification: The work introduces no theoretical assumptions and results.

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: We disclose all the training and evaluation recipes in Section 4.1.

   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: We do not disclose the code and data at the moment. But we explain the model architecture and experimental settings in detail in Section 4.1.

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, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We explain the detailed experimental settings in Section 4.1. For the composition of the benchmarks used in the paper, please refer to the appendix.

   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: Due to the computational tractability, we conduct each experiment once.

   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: The number of trainable parameters and GPU memory footprint for inference are reported in Table 8. We also explain the computational environments in the appendix.

   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: We follow 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: [NA]

    Justification: The work poses no societal 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: This paper releases no findings with 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: All the datasets in this paper are free from license issues and are cited properly. We also cite the models used in the paper. Please refer to Section 4.1.

    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 introduce 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: This paper contains no experiments involving 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: [NA]

    Justification: This work contains no 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: We used LLMs for editing purposes only.

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.

Appendix

This appendix presents additional experimental details and results that are omitted from the main paper due to space limit.

A Experimental details

Implementation details. Each GaRA module consists of a branch selector, implemented as a two-layer MLP with hidden dimension 64 and ReLU activation, which outputs a binary gating decision (low vs. high rank) using a Gumbel-Sigmoid; and two rank-wise gating networks (one for low-rank, one for high-rank), each composed of a three-layer MLP with hidden dimension 16 and ReLU activation, and produces binary masks over rank-1 components via Gumbel-Sigmoid sampling. The selected components modulate separate LoRA paths applied to the QKV projections. Following RobustSAM [8], we use a combined segmentation loss, $\mathcal{L}_{\text{seg}} = \mathcal{L}_{\text{dice}}$ [53] $+ \mathcal{L}_{\text{focal}}$ [41].

Resource. We use 8 A6000 GPUs for training each method and 1 A6000 GPU for evaluation.

Dataset configuration. Following the dataset configuration introduced in RobustSAM [8], we utilize a diverse collection of segmentation datasets for training and evaluation. For training, we adopt three synthetic datasets: MSRA10K [9] (10,000 images) and LVIS [15] (20,252 images) provide large-scale pixel-wise annotations for salient and instance segmentation, while ThinObject-5K [37] (4,748 images) contains fine-grained masks for thin structures to enhance structural sensitivity. For evaluation, we consider both synthetic and real-world scenarios. The synthetic benchmark includes STREETS [51] (1,000 annotated traffic scenes) and NDD20 [55] (1,000 dolphin segmentation images captured in underwater and above-water environments). We exclude FSS-1000 [34] due to licensing restrictions. For real-world evaluation, we adopt BDD-100K [61], a driving video dataset with 100,000 videos (frame-level annotations at the 10th second) covering diverse conditions, and LIS [5], a low-light instance segmentation dataset with 2,230 paired short/long exposure images (8,920 total images) containing over 10,000 labeled instances.

B Empirical analysis

B.1 Rank distribution analysis

We analyzed the gated rank distribution across all blocks and corruption types on the MSRA10K dataset. As shown in Figure a1, the activated rank values vary significantly not only across corruption types but also across blocks, indicating that GaRA dynamically adapts its rank selection.

B.2 Corruption type vs. gated rank

To further investigate whether our gating mechanism responds differently depending on the corruption type, we conduct an F-statistics analysis on the number of activated ranks across transformer blocks. In this setting, each corruption type is treated as a group, and each sample's gated rank serves as an observation. We compute the F-statistic, which measures the ratio of between-group to within-group variance, and the p-value, which indicates whether the observed variation could arise by chance.

As shown in Figure a2, most blocks exhibit a statistically significant dependency between corruption types and the number of activated ranks (i.e., high F-statistic and low p-value). Interestingly, blocks 2, 5, 8, and 11 do not show such significance. These blocks use standard attention, whereas the others adopt window attention, according to the original structure of SAM [28]. In the window-based setting, different windows within the same image may capture varied content under the same corruption type, leading the shared gating network to attend more to the global corruption style rather than content. This behavior highlights the corruption-aware design of our gating module.

B.3 Justification for rank space separation

The rank space separation is motivated by our empirical observations that, in a unified rank space, the model exhibits limited diversity in (rank-1) component activation across different degraded inputs. Specifically, as shown in Table a1, the model tends to activate components within a narrow

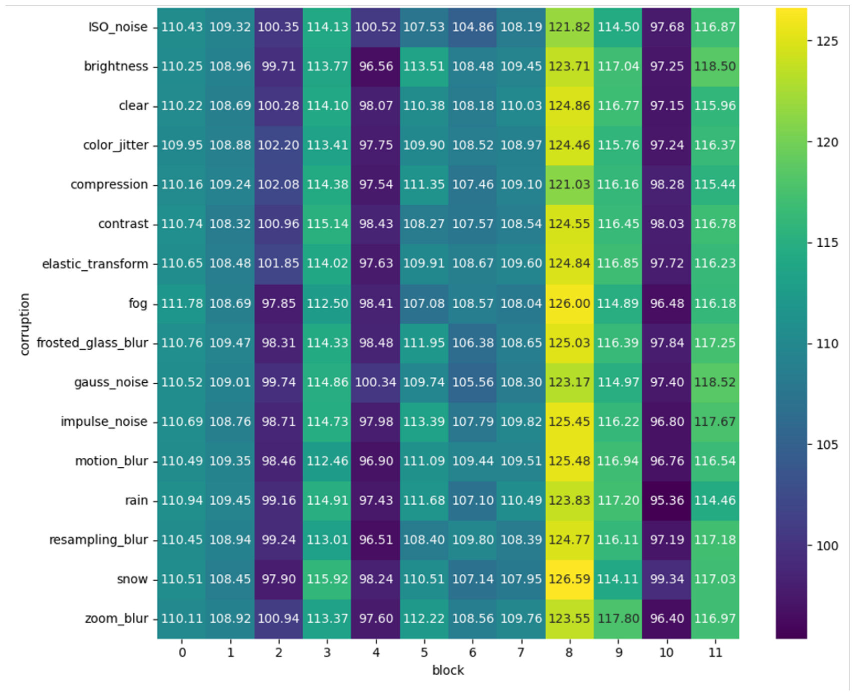

Figure a1: Block-wise and corruption-wise average activated ranks on MSRA10K. Each cell represents the mean number of activated rank components for a given corruption type and transformer block. The clear variance across both axes demonstrates that GaRA performs input- and corruption-aware rank selection, rather than using a fixed configuration.

Table a1: Effect of rank space separation.

Variant	Rank Space Separation	(rank-1) Selection	IoU	# of Active (rank-1) Components	Range
Unified Rank Space		✓	73.3	161.8 ± 7.7	133–190
GaRA-SAM	✓	✓	**77.0**	85.7 ± 75.8	5–181

range (133-190; std: 7.7), limiting its ability to adjust representational capacity according to each degraded input. This is potentially suboptimal because different degraded inputs require different representational capacities. By separating the rank space into lower-rank and higher-rank regions, we allow the model to activate a more diverse range of components (5-181; std: 75.8), yielding significant performance improvement. We guess this is because the rank space separation into two exclusive sets, each specialized for different degraded inputs demanding different representation capacities, helps mitigate potential conflicts during (rank-1) component selection and promotes more diverse usage of components.

C Additional experimental results

C.1 Generalization ability beyond driving scenarios

To further evaluate the generalization capability of GaRA-SAM beyond driving scenarios, we additionally tested it on the ISIC-2016 medical segmentation dataset [10]. As shown in Table a2, using box prompts, GaRA-SAM showed notable improvements over SAM. These results indicate that GaRA-SAM maintains strong performance even in domains with very different visual characteristics.

C.2 Isolating the effect of dynamic (rank-1) component selection

In Table a3, we tested variants using only dynamic (rank-1) component selection within a unified rank space. Enabling the dynamic selection improves performance when using a unified rank space with the small maximum rank (i.e., 16), highlighting the benefit of adaptively choosing components

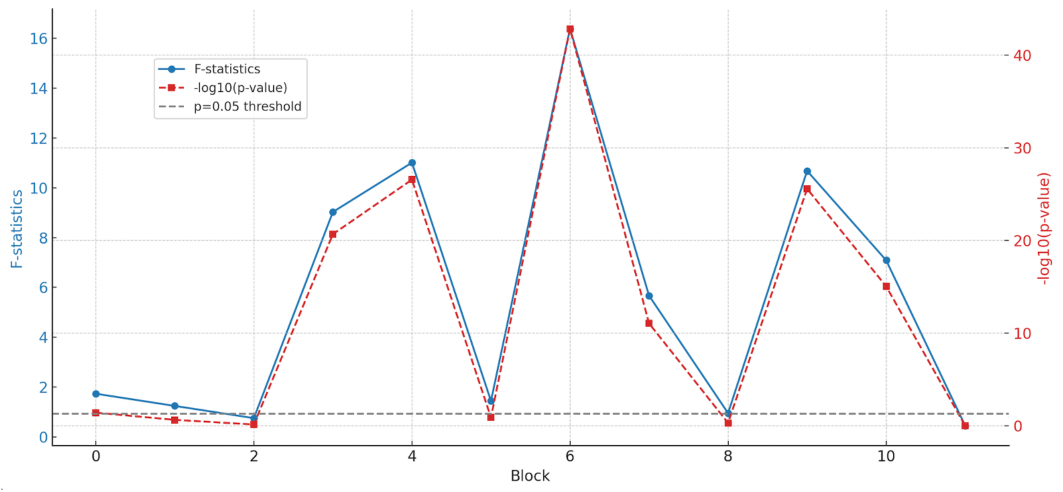

Figure a2: Statistical analysis of gated rank selection under different corruption types on MSRA10K. For each block, we compute the F-statistic and p-value (shown as $-\log_{10}(p)$). A higher F-statistic and p-value below 0.05 (dashed line) indicate that the variation in activated ranks across corruption types is statistically significant.

Table a2: Generalization performance on the ISIC-2016 medical segmentation dataset.

Model	**IoU**	**Dice**	**AP**
SAM	70.6	80.7	76.8
GaRA-SAM	**78.7**	**86.2**	**81.7**

per input. However, using a unified rank space with the large maximum rank (i.e., 256) did not lead to better results. We attribute this to the limited diversity in active components when using a unified space, which can restrict the ability to adjust representational capacities according to each degraded input, thus hindering input-adaptive modulation. It motivated our design to split the rank space, which led to improved performance.

C.3 Further finetuning GaRA-SAM on a real-world dataset

GaRA-SAM supports training on real-world corrupted datasets without requiring clean counterparts. To validate this capability, we further fine-tune GaRA-SAM on real-world degradation datasets, namely BDD-100K and LIS. In contrast, RobustSAM is not designed for unpaired training and cannot be directly applied in this setting. Nevertheless, for a fair comparison, we fine-tune RobustSAM on the same datasets using only segmentation loss, despite the lack of paired supervision. We evaluate both models on BDD+LIS (seen during fine-tuning) and ACDC, which serves as a real-world dataset not seen during fine-tuning. As shown in Table a4, this additional finetuning leads to significant performance improvements across all benchmarks. This result highlights that GaRA-SAM not only generalizes well in a zero-shot setting, but also scales effectively with real-world corrupted inputs, whereas RobustSAM lacks inherent support for this setting and is only included for comparative purposes.

C.4 Quantitative results

We report additional experimental results in Tables a5,a6,a7,a8,a9, including evaluations using the pixel accuracy (PA) metric. PA values are consistently reported across all benchmark settings using both point and box prompts.

Table a3: Ablation on dynamic (rank-1) component selection and rank-space gating.

Variant	**(rank-1) Component Selection**	**Rank space gating**	**IoU**
w/o (rank-1) component selection (max rank 16)			75.0
w/ (rank-1) component selection (max rank 16)	✓		76.2
w/o (rank-1) component selection (max rank 256)			73.4
w/ (rank-1) component selection (max rank 256)	✓		73.3
GaRA-SAM (max rank 256)	✓	✓	**77.0**

Table a4: Fine-tuning results with ViT-L. Models are further fine-tuned on BDD+LIS and evaluated on BDD+LIS and ACDC using a box prompt.

Method	Fine-tuning	BDD+LIS		ACDC	
		IoU	Dice	IoU	Dice
SAM		74.3	82.0	65.9	72.6
RobustSAM		78.1	86.1	67.5	77.0
RobustSAM	✓	<u>85.9</u>	<u>91.5</u>	<u>83.9</u>	<u>89.5</u>
GaRA-SAM		80.0	87.4	71.6	80.4
GaRA-SAM	✓	**89.5**	**93.8**	**88.3**	**92.6**

C.5 Qualitative results

We present additional qualitative comparisons in Fig. a3, showing the segmentation results of SAM, HQ-SAM, RobustSAM, and our GaRA-SAM under a variety of degraded inputs. The top section illustrates results under synthetic corruptions (*e.g.*, motion blur, snow, noise), while the bottom section includes real-world challenges such as low-light and adverse weather conditions. Compared to other baselines, GaRA-SAM consistently demonstrates more robust and accurate segmentation masks, especially in cases where the standard SAM or HQ-SAM fails to localize object boundaries. These results highlight GaRA-SAM's strong generalization capabilities across both synthetic and naturally occurring corruptions.

Table a5: Additional evaluation results on LVIS including pixel accuracy (PA) using point and box prompts.

Backbone	Method	Point Prompts						Box Prompts					
		Degrade			Clear			Degrade			Clear		
		IoU	PA	Dice	IoU	PA	Dice	IoU	PA	Dice	IoU	PA	Dice
ViT-B	SAM	65.0	89.8	76.3	70.1	91.5	80.1	75.6	93.5	84.5	78.4	93.7	86.1
	HQ-SAM	69.5	91.8	80.1	76.0	93.3	84.8	79.0	94.8	87.1	83.1	95.3	89.7
	AirNet+SAM	64.8	89.8	76.1	70.1	91.5	80.1	75.4	93.4	84.4	78.3	93.6	86.0
	URIE+SAM	64.8	89.9	76.2	70.0	91.4	80.0	74.5	93.2	83.7	77.9	93.5	85.8
	RobustSAM	72.7	93.0	82.6	77.2	93.9	85.8	81.5	95.0	89.0	84.3	95.5	90.8
	GaRA-SAM	**77.0**	**94.4**	**85.7**	**81.3**	**95.3**	**88.7**	**83.7**	**96.0**	**90.5**	**86.1**	**96.3**	**91.9**
ViT-L	SAM	66.2	87.1	76.0	75.0	91.5	83.0	79.9	94.8	87.7	82.8	95.1	89.4
	HQ-SAM	72.6	92.5	82.1	79.0	94.1	86.7	81.1	95.4	88.6	84.7	95.8	90.8
	AirNet+SAM	66.0	86.9	75.7	74.8	91.4	82.8	79.7	94.8	87.6	82.7	95.0	89.4
	URIE+SAM	66.4	87.6	76.3	74.8	91.5	83.0	79.4	94.7	87.4	82.7	95.0	89.4
	RobustSAM	75.6	94.1	84.5	80.0	95.1	87.5	83.6	**96.0**	90.3	85.8	96.3	91.6
	GaRA-SAM	**78.5**	**94.8**	**86.6**	**82.6**	**95.6**	**89.4**	**84.4**	**96.0**	**90.8**	**86.7**	**96.5**	**92.2**

Table a6: Evaluation results on MSRA using point and box prompts.

Backbone	Method	Point Prompts						Box Prompts					
		Degrade			Clear			Degrade			Clear		
		IoU	PA	Dice	IoU	PA	Dice	IoU	PA	Dice	IoU	PA	Dice
ViT-B	SAM	75.9	93.5	84.5	79.1	94.8	86.6	86.6	97.0	92.2	88.8	97.5	93.4
	HQ-SAM	82.8	95.7	89.6	86.6	96.7	92.0	89.7	97.7	94.3	92.5	98.3	95.9
	AirNet+SAM	75.7	93.5	84.3	79.1	94.8	86.6	86.4	96.9	92.1	88.8	97.5	93.5
	URIE+SAM	74.6	93.2	83.5	77.6	94.5	85.6	86.0	96.8	91.8	88.8	97.5	93.5
	RobustSAM	86.6	96.6	92.3	89.6	97.4	94.2	89.5	97.6	94.2	92.1	98.2	95.7
	GaRA-SAM	**89.4**	**97.5**	**94.0**	**91.4**	**98.0**	**95.2**	**92.3**	**98.3**	**95.8**	**93.9**	**98.7**	**96.8**
ViT-L	SAM	77.3	89.0	84.6	82.0	92.1	88.1	87.6	97.2	92.9	88.9	97.5	93.5
	HQ-SAM	85.0	95.9	91.0	87.6	96.6	92.6	89.4	97.7	94.1	91.4	98.1	95.2
	AirNet+SAM	76.9	88.7	84.3	81.7	91.8	87.8	87.5	97.2	92.8	89.0	97.5	93.6
	URIE+SAM	78.1	90.0	85.4	83.0	93.0	88.9	87.6	97.2	92.8	89.6	97.7	94.0
	RobustSAM	87.6	96.8	92.9	90.1	97.4	94.4	91.6	98.1	95.4	93.6	98.6	96.5
	GaRA-SAM	**89.6**	**97.5**	**94.1**	**91.6**	**98.0**	**95.3**	**92.6**	**98.4**	**96.0**	**94.0**	**98.7**	**96.8**

Table a7: Evaluation results on COCO using point and box prompts.

Enc.	Method	**Point Prompts**						**Box Prompts**					
		Degrade			Clear			Degrade			Clear		
		IoU	PA	Dice	IoU	PA	Dice	IoU	PA	Dice	IoU	PA	Dice
ViT-B	SAM	65.2	90.2	76.3	69.6	91.7	79.6	76.0	93.6	84.8	78.9	93.9	86.4
	HQ-SAM	70.2	92.2	80.6	76.0	93.6	84.8	79.6	95.1	87.5	83.6	95.6	90.1
	AirNet+SAM	65.0	90.1	76.1	69.5	91.7	79.5	75.9	93.6	84.7	78.8	93.9	86.3
	URIE+SAM	65.0	90.1	76.2	69.6	91.6	79.6	75.0	93.4	84.0	78.2	93.7	85.8
	RobustSAM	72.7	93.0	82.6	77.1	93.9	85.7	81.6	95.2	89.1	84.5	95.8	90.9
	GaRA-SAM	**77.2**	**94.5**	**85.8**	**81.0**	**95.3**	**88.4**	**84.2**	**96.3**	**90.8**	**86.4**	**96.5**	**92.1**
ViT-L	SAM	66.9	87.8	76.5	74.6	91.4	82.6	80.4	95.1	88.0	82.9	95.3	89.4
	HQ-SAM	73.2	92.8	82.5	79.0	94.3	86.7	81.7	95.7	89.0	85.1	96.1	91.1
	AirNet+SAM	66.6	87.6	76.3	74.4	91.3	82.5	80.3	95.1	87.9	82.8	95.3	89.3
	URIE+SAM	66.6	87.8	76.4	74.3	91.4	82.4	79.9	95.0	87.7	82.8	95.3	89.4
	RobustSAM	75.3	94.0	84.3	79.9	95.2	87.6	83.8	96.1	90.5	86.2	96.5	91.9
	GaRA-SAM	**78.9**	**94.8**	**86.9**	**82.3**	**95.5**	**89.1**	**84.9**	**96.3**	**91.2**	**86.9**	**96.7**	**92.4**

Table a8: Evaluation results on STREETS+NDD using point and box prompts.

Enc.	Method	**Point Prompts**						**Box Prompts**					
		Degrade			Clear			Degrade			Clear		
		IoU	PA	Dice	IoU	PA	Dice	IoU	PA	Dice	IoU	PA	Dice
ViT-B	SAM	74.4	98.9	83.5	81.8	99.5	89.1	80.2	99.7	88.3	85.4	99.8	91.8
	HQ-SAM	75.4	99.4	84.5	82.2	99.6	89.5	80.8	99.7	88.7	86.2	99.8	92.3
	AirNet+SAM	74.3	98.9	83.4	81.8	99.5	89.1	80.1	99.7	88.2	85.4	99.8	91.8
	URIE+SAM	74.1	99.1	83.4	81.0	99.5	88.6	79.8	99.7	88.0	84.9	99.7	91.5
	RobustSAM	74.5	99.4	84.1	81.0	99.5	88.8	81.8	99.7	89.4	86.2	99.8	92.4
	GaRA-SAM	**77.6**	**99.5**	**86.4**	**82.9**	**99.7**	**90.1**	**84.1**	**99.8**	**91.0**	**87.7**	99.8	**93.3**
ViT-L	SAM	72.9	96.7	81.3	82.1	98.8	89.0	81.5	99.7	89.2	86.4	99.8	92.5
	HQ-SAM	76.8	99.3	85.4	83.9	99.6	90.6	81.7	99.7	89.3	86.7	99.8	92.6
	AirNet+SAM	72.5	96.6	81.0	81.9	98.7	88.7	81.4	99.7	89.1	86.4	99.8	92.5
	URIE+SAM	72.6	97.3	81.4	81.2	98.9	88.4	81.1	99.7	88.9	86.1	99.8	92.2
	RobustSAM	75.7	99.4	84.8	82.7	99.6	89.9	83.0	99.7	90.2	87.5	99.8	93.1
	GaRA-SAM	**80.0**	**99.6**	**88.0**	**85.3**	**99.7**	**91.7**	**85.3**	**99.8**	**91.7**	**88.7**	99.8	**93.9**

Table a9: Evaluation results on BDD+LIS using point and box prompts.

Enc.	Method	**Point Prompts**			**Box Prompts**		
		IoU	PA	Dice	IoU	PA	Dice
ViT-B	SAM	65.2	87.8	75.4	74.2	92.3	82.7
	HQ-SAM	67.6	90.0	77.9	69.4	91.7	78.0
	AirNet+SAM	64.1	87.0	74.5	73.4	92.1	82.1
	URIE+SAM	65.7	89.0	76.1	74.1	**92.5**	82.8
	RobustSAM	<u>69.5</u>	**91.1**	<u>79.5</u>	<u>75.7</u>	<u>92.4</u>	<u>84.1</u>
	GaRA-SAM	**71.3**	<u>91.0</u>	**80.9**	**76.8**	91.8	**85.3**
ViT-L	SAM	68.9	86.4	77.8	74.3	91.9	82.0
	HQ-SAM	<u>72.7</u>	<u>91.6</u>	<u>81.7</u>	76.3	<u>93.4</u>	84.3
	AirNet+SAM	67.2	84.8	76.4	73.1	91.6	81.3
	URIE+SAM	69.7	87.8	78.8	74.4	92.0	82.1
	RobustSAM	71.4	91.1	80.9	<u>78.1</u>	92.5	<u>86.1</u>
	GaRA-SAM	**74.8**	**92.2**	**83.4**	**80.0**	**93.5**	**87.4**

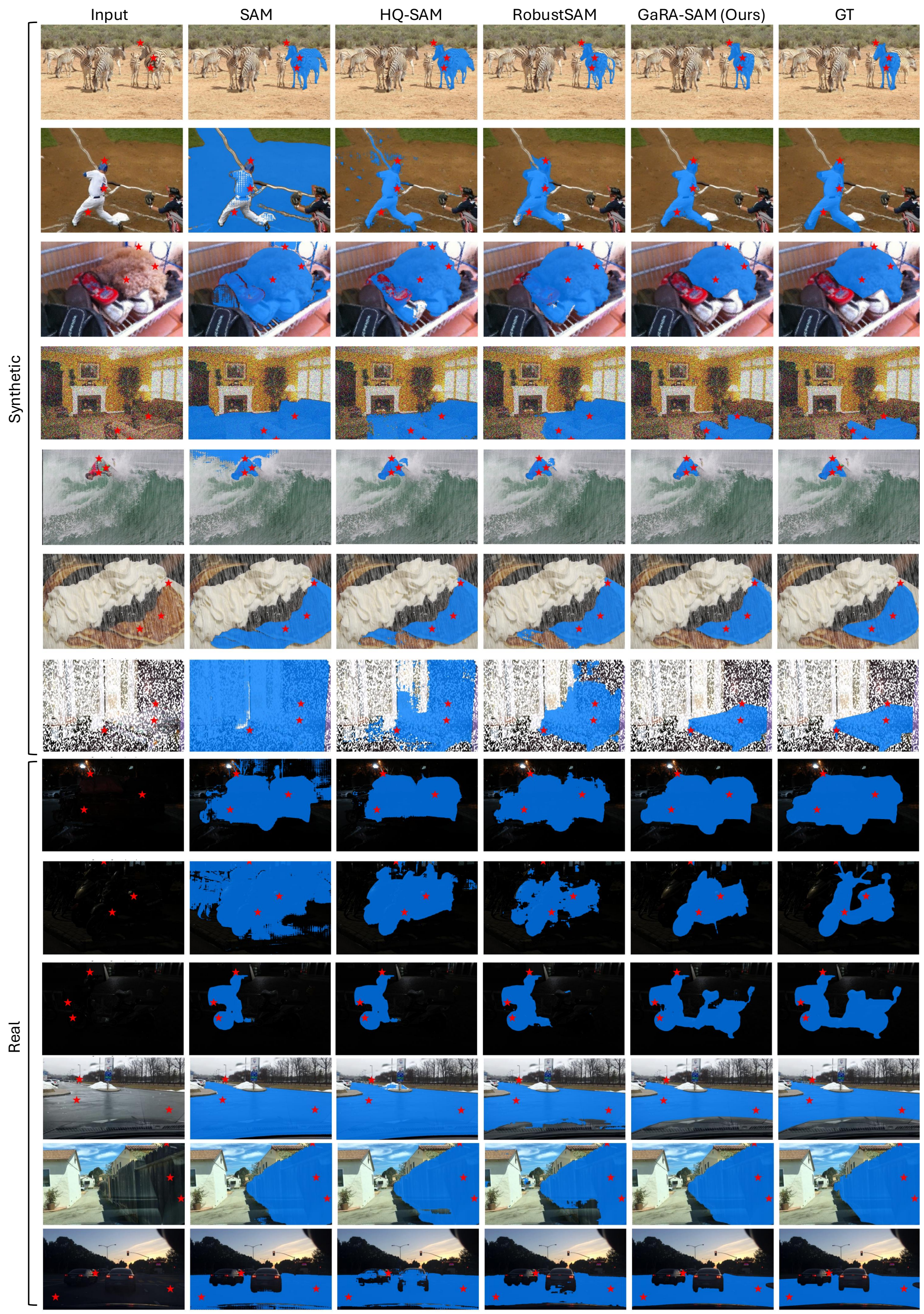

Figure a3: Qualitative results under both synthetic (top) and real-world (bottom) corruptions. From left to right: input image, SAM, HQ-SAM, RobustSAM, our method GaRA-SAM, and the ground truth (GT). GaRA-SAM consistently produces more accurate and complete segmentations, particularly under challenging conditions such as noise, blur, and low-light scenarios.