# Dynamic Region-Aware Gradient Suppression (DRAGS): Enhancing Vision Model Robustness by Suppressing Noisy Feature Regions During Training

Md Muntaqim Meherab[1]*, Nuruzzaman Faruqui[2], Faria Nishat Khan[3], Tanvirul Islam[1], Syed Asif Johan[4], Md. Maruf Billah[5], Kazi Shakkhar Rahman[6], Z N M Zarif Mahmud[1], and Tauhidul As Sami[7]

[1] Dept. of Computer Science and Engineering, Daffodil International University, Dhaka, Bangladesh
{meherab2305101354, tanvirulislam.cse, zarif.cse}@diu.edu.bd
[2] Dept. of Software Engineering, Daffodil International University, Dhaka, Bangladesh
faruqui.swe@diu.edu.bd
[3] PhD in Data Science and Engineering, South Dakota School of Mines and Technology, Rapid City, SD, USA
farianishat.khan@mines.sdsmt.edu
[4] Dept. of Computer Science and Engineering, Rajshahi University of Engineering and Technology, Rajshahi, Bangladesh
Connect.syedasifjohan@gmail.com
[5] Dept. of Electrical and Computer Engineering, North South University, Dhaka, Bangladesh
maruf.billah.232@northsouth.edu
[6] Dept. of Computer Science and Engineering, Islamic University of Technology, Gazipur, Bangladesh
shakkharrahman@iut-dhaka.edu
[7] Dept. of Biomedical Engineering, Bangladesh University of Engineering and Technology, Dhaka, Bangladesh
2118030@bme.buet.ac.bd

**Abstract.** Deep vision models can perform very well on clean test sets but still break down when inputs are corrupted by noise, blur, or occlusion. We introduce *Dynamic Region-Aware Gradient Suppression* (DRAGS), a lightweight training-time mechanism that suppresses gradients from spatial regions detected as noisy or spurious for the current mini-batch. DRAGS computes a saliency score at each location in an intermediate feature map, keeps only the top $\tau$ fraction, and detaches the gradients from the remaining locations. This dynamic, region-level gating differs from classical regularizers that act uniformly across space or channels.
On CIFAR-10 with a ResNet-18 backbone, DRAGS yields a consistent improvement in clean accuracy over a strong baseline (**+1.39** points; 64.15% ± 2.30 vs. 62.76% ± 1.05 across three seeds) and maintains robustness under common corruptions, with slight gains on Gaussian noise and

* Corresponding author: meherab2305101354@diu.edu.bd


M. S. Arefin et al. (eds.), *Proceedings of the International Conference on Intelligent Data Analysis and Applications (IDAA 2025)*, Advances in Intelligent Systems Research 206,
https://doi.org/10.2991/978-94-6239-664-7_47

occlusion and a measured trade-off under motion blur. A short ablation over $\tau$ and layer placement shows that moderate suppression on early layers is the most reliable setting. Heatmaps confirm that DRAGS down-weights visually noisy background zones while preserving informative object regions. Overall, DRAGS is simple to implement and compute-efficient relative to adversarial training, making it a practical option for robustness-minded training at scale.

**Keywords:** Robustness · regularization · gradient gating · dynamic masking · corruptions · CIFAR-10 · ResNet-18 · computer vision

## 1 Introduction

Deep vision models can achieve very strong performance on clean test sets, but they often remain brittle once the input distribution shifts because of noise, blur, or occlusion [6]. Regularization and augmentation methods such as Cutout [2], DropBlock [3], Mixup [11], and AutoAugment [1] help generalization, but they usually act uniformly across space or channels. In other words, they do not distinguish between regions that carry useful signal and regions that are mostly background clutter or artifacts. Adversarial training [4,8] can improve robustness more directly, but it comes with a considerable computational cost and is rarely used as a default training strategy.

In this paper, we explore a simple alternative: instead of regularizing all locations equally, we try to steer the optimizer to listen more to the parts of the feature map that look informative and less to those that look noisy. We propose **DRAGS**—*Dynamic Region-Aware Gradient Suppression*—a small training-time module that sits on top of intermediate feature maps. For each mini-batch, DRAGS builds a saliency proxy at every spatial location, keeps the top $\tau$ fraction of locations, and detaches the gradients flowing through the rest. The forward activations are left unchanged, so the model behaves exactly like the baseline at inference time, but the parameter updates during training are guided toward regions that consistently appear useful.

Our contributions are:

- **Method.** We introduce a simple, drop-in, region-aware gradient gate that runs only at training time and can be plugged into standard backbones without changing their architecture or inference cost.
- **Analysis.** We provide multi-seed evaluation with mean±std reporting and study how the keep ratio and layer placement affect accuracy, robustness, and training time on CIFAR-10 with ResNet-18.
- **Visualization.** We present qualitative heatmaps that illustrate how DRAGS tends to suppress visually noisy background regions while preserving salient object areas.

Table 1: Conceptual comparison of DRAGS with representative robustness and regularization approaches.

| Method | Level | Region selection | Primary effect | Inference cost |
|---|---|---|---|---|
| Cutout / DropBlock | Input / features | Random patches | Masked activations | Unchanged |
| Mixup / AutoAugment | Input | Global transforms | Data / labels | Unchanged |
| Grad-CAM (saliency) | Features / logits | Gradient-based (post-hoc) | Visualization only | Unchanged |
| Adversarial training | Input | Worst-case perturbations | Loss and gradients | Increased |
| **DRAGS (ours)** | Intermediate features | Top-$\tau$ by saliency $S$ | **Gradients gated by $M$** | **Unchanged** |

## 2 Related Work

**Robustness to Common Corruptions.** CIFAR-10-C/ImageNet-C benchmark robustness to noise, blur, and other corruptions [6]. Antialiasing and blur-aware layers [12] can help, though effects can be architecture-dependent.
**Regularization & Augmentation.** Spatial dropout and masking techniques (Cutout [2], DropBlock [3]) remove random patches or contiguous regions; Mixup/AutoAugment [1, 11] reshape the data distribution. These are input-level; DRAGS intervenes at *feature-level* during backprop.
**Saliency & Gradient Manipulation.** Gradient-based explanations (Grad-CAM [9]) relate spatial saliency to decision making. DRAGS shares the intuition that not all locations should contribute equally, but instead modifies gradients during training to suppress unhelpful regions.
**Adversarial Training.** While effective [4, 8], adversarial training is compute-intensive. DRAGS offers a compute-light alternative that improves clean accuracy and preserves robustness on common corruptions without adversarial inner loops.

### 2.1 Novelty and Positioning of DRAGS

Most robustness methods fall into three groups: (i) make inputs harder through augmentation or masking, (ii) harden the model with adversarial objectives, or (iii) inspect saliency maps after training without changing the learning rule. DRAGS sits a bit differently. It takes a simple, instance-wise saliency signal $S$ from intermediate features and uses it *inside* the training loop to gate gradients via a mask $M$, while leaving the forward activations and architecture untouched.

This makes DRAGS more targeted than uniform regularizers (which treat all locations equally) and much lighter than adversarial training (no inner-loop optimization, no extra inference cost). Table 1 highlights this contrast.

## 3 Method: Dynamic Region-Aware Gradient Suppression (DRAGS)

Let $\mathbf{X} \in \mathbb{R}^{B\times C\times H\times W}$ be feature maps at an intermediate layer. DRAGS computes a region saliency proxy $S \in \mathbb{R}^{B\times 1\times H\times W}$, e.g., average across channels (optionally after local pooling), and forms a *top-k* binary mask $M = \mathbf{1}\{S \text{ in top } \tau\}$ where

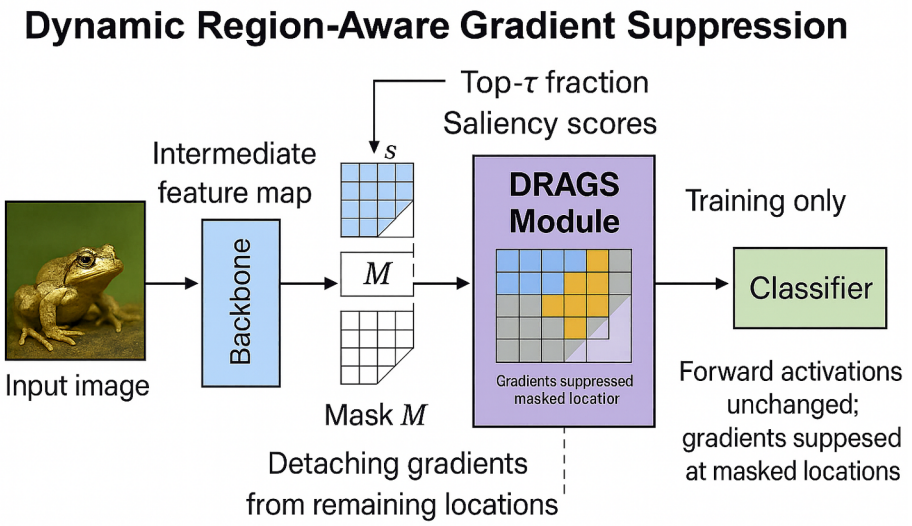


Fig. 1: Overview of the Dynamic Region-Aware Gradient Suppression (DRAGS) module. An input image is processed by the backbone to produce an intermediate feature map. DRAGS computes saliency scores $S$, selects the top-$\tau$ locations to form a binary mask $M$, and uses it to suppress gradients at masked locations while leaving forward activations unchanged. The classifier operates on the backbone features as usual; DRAGS is applied only at training time.

$\tau \in (0, 1]$ is the keep ratio. During backprop, we detach gradients from locations where $M = 0$:

$$\tilde{\mathbf{X}} = M \odot \mathbf{X} + (1 - M) \odot \text{stopgrad}(\mathbf{X}). \tag{1}$$

Forward activations pass through unchanged, preserving inference-time behavior. Only gradients from the bottom $(1 - \tau)$ fraction are suppressed. An overview of how DRAGS is inserted into a standard backbone and how gradients are gated during training is shown in Fig. 1.

### 3.1 Design Choices

**Saliency proxy.** We use channel-average (optionally preceded by $k \times k$ average pooling) to stabilize ranking across channels.

**Gating granularity.** A spatial mask operates per location.

**Placement.** We insert DRAGS in early/mid blocks (e.g., ResNet-18 `layer1`/`layer2`) to steer feature learning without interfering with late layers.

**Hyperparameters.** Top-$k$ ratio $\tau$ and pool kernel $k$ are the only tunables. In our ablation we vary $\tau \in \{0.05, 0.10, 0.15\}$.

**Algorithm 1** DRAGS (training-time, per mini-batch)

**Require:** Feature maps $\mathbf{X} \in \mathbb{R}^{B\times C\times H\times W}$, keep ratio $\tau$, pool kernel $k$.
1: $\mathbf{A} \leftarrow \text{AvgPool}_k(\mathbf{X})$ ▷ stabilize local responses
2: $S \leftarrow \text{Mean}_C(\mathbf{A})$ ▷ saliency proxy, shape $B \times 1 \times H \times W$
3: For each $b$: compute threshold $t_b$ so that $\tau$ fraction of $S_b$ are $\geq t_b$
4: $M_b \leftarrow \mathbf{1}\{S_b \geq t_b\}$ ▷ binary top-$k$ spatial mask
5: $\tilde{\mathbf{X}} \leftarrow M \odot \mathbf{X} + (1 - M) \odot \text{stopgrad}(\mathbf{X})$
6: **return** $\tilde{\mathbf{X}}$

Table 2: Baseline vs DRAGS on CIFAR-10 (mean±std over seeds 42, 43, 44).

| Model | Clean | Gaussian | Motion Blur | Occlusion | Train Time (s) |
|---|---|---|---|---|---|
| Baseline | $62.76 \pm 1.05$ | $57.30 \pm 2.51$ | $27.16 \pm 1.64$ | $55.06 \pm 2.71$ | $113.25 \pm 2.69$ |
| DRAGS | $\mathbf{64.15} \pm 2.30$ | $\mathbf{57.54} \pm 2.08$ | $25.41 \pm 1.12$ | $\mathbf{55.26} \pm 1.38$ | $225.76 \pm 3.21$ |

### 3.2 Complexity

DRAGS adds only light pooling and top-$k$ operations. In our runs, training time roughly doubled compared with the baseline (no gating), still far cheaper than adversarial training.

## 4 Experimental Setup

**Dataset.** CIFAR-10 [7] train/test split; we additionally evaluate under four common test conditions: clean, Gaussian noise, motion blur, and occlusion (as in CIFAR-10-C [6]).

**Backbone.** ResNet-18 [5].

**Optimization.** Stochastic gradient descent with standard settings; label smoothing and heavy augmentation were not used to isolate DRAGS effects [10].

**Training budget.** Each run used a short schedule (2 epochs) to standardize compute across ablations and seeds; we report *mean*±*std* over three seeds $\{42, 43, 44\}$. While longer schedules would increase absolute accuracy, we focus on the *differential* impact of DRAGS, which is stable across seeds.

**DRAGS placement.** Unless otherwise stated, DRAGS is inserted at `layer1` and `layer2` with pool kernel $k$=3 and keep ratio $\tau$=0.10.

## 5 Results

### 5.1 Main Comparison (3 Seeds)

Table 2 reports clean and corrupted accuracy (%) and training time in seconds. DRAGS improves clean accuracy by **+1.39** points on average and maintains robustness under Gaussian noise and occlusion with a small cost on motion blur.
**Per-seed raw results.** For completeness, Table 3 lists the raw per-seed scores you obtained.

Table 3: Per-seed accuracy (%) and train time (s).

| Seed | Model | Clean | Gauss | Motion | Occl. | Time |
|---|---|---|---|---|---|---|
| 42 | Base | 62.31 | 54.42 | 29.01 | 52.31 | 112.42 |
| 42 | DRAGS | 66.10 | 59.93 | 26.63 | 56.84 | 222.92 |
| 43 | Base | 63.96 | 59.03 | 25.90 | 57.72 | 111.07 |
| 43 | DRAGS | 61.61 | 56.13 | 25.18 | 54.64 | 225.13 |
| 44 | Base | 62.00 | 58.46 | 26.57 | 55.14 | 116.26 |
| 44 | DRAGS | 64.75 | 56.56 | 24.42 | 54.31 | 229.24 |

### 5.2 Ablation Study

We sweep the keep ratio $\tau \in \{0.05, 0.10, 0.15\}$ and placement (`layer1` vs `layer1+layer2`). Table 4 shows *mean*±*std* over seeds. Trends:

- Moderate keep ratios ($\tau$=0.10 or 0.15) on `layer1` yield the most reliable clean accuracy (50.08% ± 1.69 to 50.50% ± 2.21).
- Adding `layer2` generally increases compute ($\sim$ 92.0 s vs 66.0 s) and can reduce stability at high suppression (e.g., $\tau$=0.15).
- Robustness under Gaussian/occlusion tracks clean performance; motion blur is sensitive to spatial gating and benefits from careful tuning.

### 5.3 Visualization

Fig. 2 shows DRAGS heatmaps overlaid on test images. Suppressed regions (cool colors) typically align with background clutter and occluders, while salient object regions are preserved.

### 5.4 Bar Chart Summary

As shown in Fig. 3, DRAGS consistently outperforms the Baseline across clean and corrupted conditions.

Table 4: Ablation on top-$k$ ratio $\tau$ and layer placement (mean±std over seeds).

| $\tau$ | Layers | Clean | Gaussian | Motion Blur | Occlusion | Train Time (s) |
|---|---|---|---|---|---|---|
| 0.05 | `layer1` | 48.66 ± 2.74 | 47.72 ± 1.05 | 25.97 ± 1.92 | 41.46 ± 0.71 | 68.15 ± 1.18 |
| 0.05 | `layer1+layer2` | 48.66 ± 2.74 | 47.72 ± 1.05 | 25.97 ± 1.92 | 41.46 ± 0.71 | 90.78 ± 1.40 |
| 0.10 | `layer1` | **50.08** ± 1.69 | **48.90** ± 1.74 | 25.20 ± 2.84 | **44.12** ± 2.35 | 66.54 ± 0.56 |
| 0.10 | `layer1+layer2` | 49.55 ± 2.23 | 48.29 ± 1.88 | **26.80** ± 3.52 | 43.57 ± 2.20 | 92.67 ± 0.36 |
| 0.15 | `layer1` | 50.50 ± 2.21 | 48.01 ± 1.31 | 26.30 ± 1.49 | 42.75 ± 1.33 | **65.84** ± 0.52 |
| 0.15 | `layer1+layer2` | 45.96 ± 5.11 | 44.65 ± 3.87 | 26.19 ± 3.30 | 39.22 ± 4.95 | 93.45 ± 0.39 |

## 6 Discussion

**Where DRAGS helps.** Clean accuracy and noise/occlusion robustness benefit from region-aware suppression, likely because DRAGS discourages reliance on background textures and spurious edges.

**Motion blur sensitivity.** Spatially uniform blur reduces localized contrast; a strict top-$k$ mask can discard weak-but-useful evidence. Tuning $\tau$ and using earlier layers mitigates this.

**Compute trade-off.** DRAGS doubles training time in our setup but remains far cheaper than adversarial inner loops. At inference, there is *no* overhead.

## 7 Limitations and Broader Impact

We trained with a short schedule to enable extensive ablations. Absolute accuracy would increase with longer schedules; we expect conclusions to persist. DRAGS' suppression relies on a saliency proxy; poorly chosen proxies could attenuate minority cues. Evaluating fairness and class-conditional effects is an important future direction.

## 8 Conclusion

We presented DRAGS, a simple training-time region-aware gradient gate. Across three seeds on CIFAR-10 with ResNet-18, DRAGS improves clean accuracy and preserves robustness to common corruptions, with interpretable heatmaps that show the mechanism's effect. DRAGS is easy to add to standard backbones and offers a practical path to robustness with minimal engineering.

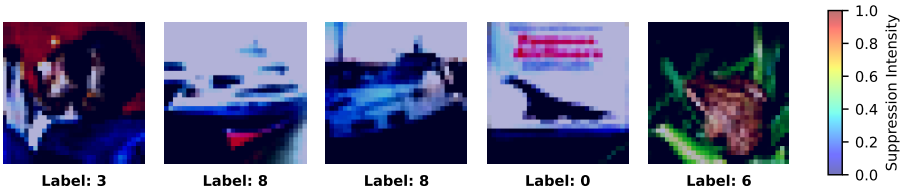


Fig. 2: DRAGS heatmap overlays on CIFAR-10 examples. The colorbar indicates suppression intensity (higher = more suppressed). Best viewed in color and zoomed in.

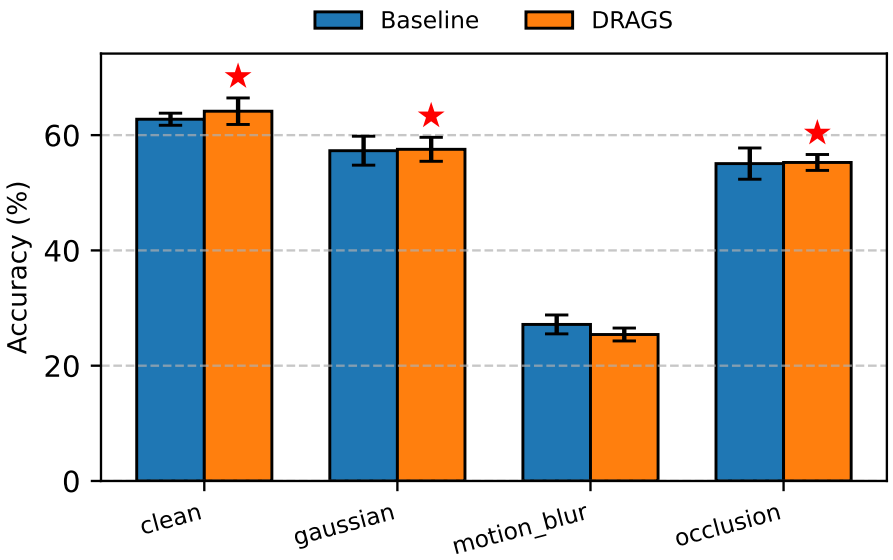


Fig. 3: Baseline vs DRAGS across clean and corrupted conditions (mean over three seeds).

## References

1. Cubuk, E.D., Zoph, B., Mane, D., Vasudevan, V., Le, Q.V.: Autoaugment: Learning augmentation policies from data. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2019)
2. DeVries, T., Taylor, G.W.: Improved regularization of convolutional neural networks with cutout (2017)
3. Ghiasi, G., Lin, T.Y., Le, Q.V.: Dropblock: A regularization method for convolutional networks. In: Advances in Neural Information Processing Systems (NeurIPS) (2018)
4. Goodfellow, I.J., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples. In: International Conference on Learning Representations (ICLR) (2015)
5. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016)
6. Hendrycks, D., Dietterich, T.: Benchmarking neural network robustness to common corruptions and perturbations. In: International Conference on Learning Representations (ICLR) (2019)
7. Krizhevsky, A.: Learning multiple layers of features from tiny images. Tech. rep., University of Toronto (2009)
8. Madry, A., Makelov, A., Schmidt, L., Tsipras, D., Vladu, A.: Towards deep learning models resistant to adversarial attacks. In: International Conference on Learning Representations (ICLR) (2018)
9. Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D.: Grad-cam: Visual explanations from deep networks via gradient-based localization. In:

Proceedings of the IEEE International Conference on Computer Vision (ICCV) (2017)
10. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016)
11. Zhang, H., Cisse, M., Dauphin, Y.N., Lopez-Paz, D.: mixup: Beyond empirical risk minimization. In: International Conference on Learning Representations (ICLR) (2018)
12. Zhang, R.: Making convolutional networks shift-invariant again. In: Proceedings of the International Conference on Machine Learning (ICML) (2019)