Weight Concentration Regularization for Improving Pruning Robustness Under High Sparsity

Anonymous Authors[1]

Abstract

Deep neural networks achieve outstanding performance across vision and language tasks, yet their large parameter counts limit deployment in resource-constrained settings. One-shot pruning reduces model size without retraining, but models trained with standard objectives often suffer substantial accuracy drops under aggressive sparsity. Prior work mitigates this drop along two directions: regularizers such as ℓ_1 and DeepHoyer that shape the weight distribution during training, and pruning-robust optimizers such as SAM, CrAM, and S^2SAM that flatten the loss landscape. Existing regularizers, however, either shrink all weights uniformly (ℓ_1) or induce scale-invariant sparsity (DeepHoyer), without concentrating weight energy onto a small set of informative parameters. We propose a Weight Concentration Regularizer (WCR), a training-time regularizer that amplifies the magnitude of a small subset of parameters while driving the remainder toward zero, so that magnitude pruning predominantly removes parameters with negligible functional contribution. We provide a convergence analysis and evaluate WCR on LLM fine-tuning, image classification, and medical segmentation, demonstrating consistent improvements in pruning robustness across architectures and compatibility with existing pruning-robust optimizers.

1. Introduction

Deep neural networks have achieved strong performance across computer vision, natural language processing, and speech recognition. However, increasing model size also increases computational and memory costs, limiting deployment in resource-constrained environments. Model pruning reduces model size by removing less important parameters while preserving performance (Hoefler et al., 2021). Although iterative pruning with retraining is effective (Frankle & Carbin, 2019), it is computationally expensive. One-shot pruning provides a simpler alternative by removing parameters in a single step without retraining or fine-tuning (Liu et al., 2019; He et al., 2017), but models trained with standard objectives often suffer substantial performance degradation under high sparsity.

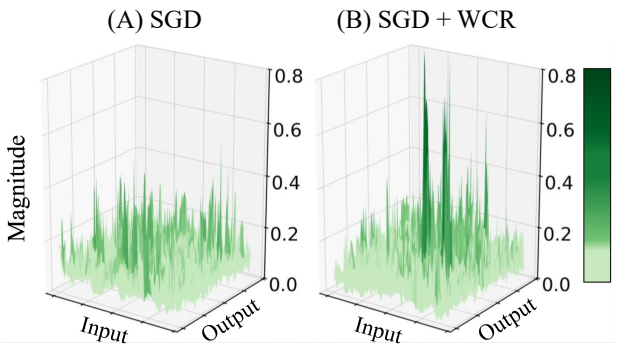

Figure 1. Visualization of parameter magnitudes of ResNet-18 (He et al., 2016) on CIFAR-10 (Krizhevsky, 2009) trained with and without the proposed Weight Concentration Regularizer (WCR). WCR concentrates weight magnitude onto a small subset of parameters.

Prior work has approached this problem through sparsity-inducing regularization and pruning-robust optimization. Methods such as ℓ_1 regularization and DeepHoyer (Yang et al., 2020) encourage sparse weight distributions, while optimizers including SAM (Foret et al., 2021), CrAM (Chen et al., 2023), and S^2SAM improve generalization through flatter minima. However, these approaches do not explicitly encourage weight energy to concentrate on a small subset of important parameters. As a result, when useful information remains diffusely distributed across many weights, magnitude pruning can still remove functionally important parameters.

To address this limitation, we propose *Weight Concentration Regularizer* (WCR), a training-time regularizer that concentrates weight energy onto a small subset of parameters while driving the remaining weights toward zero. Unlike conven-

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

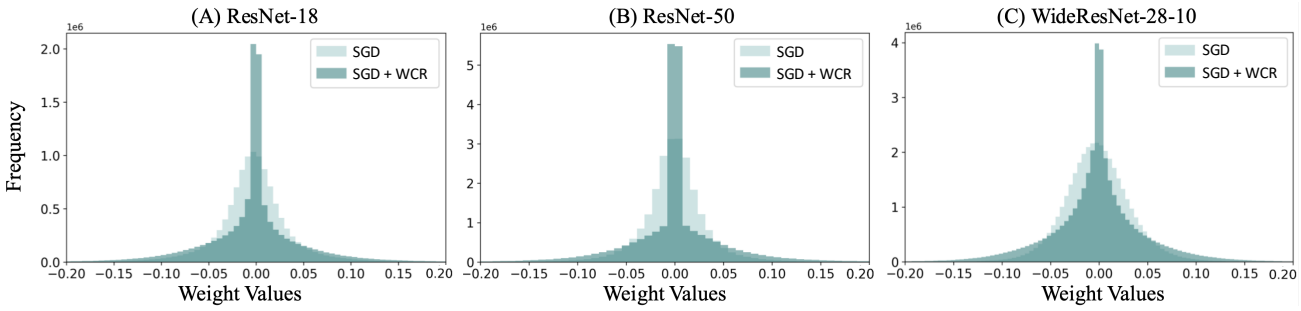

Figure 2. Weight-parameter distributions for models trained with standard SGD and SGD with WCR ($\lambda = 10^{-5}$). (A) ResNet-18/CIFAR-10, (B) ResNet-50/SVHN, (C) WideResNet-28-10/CIFAR-100. WCR yields a heavy-tailed distribution: a sharper peak near zero alongside a wider tail of large-magnitude weights, indicating simultaneous concentration and sparsification.

tional sparsity regularizers that uniformly shrink weights, WCR reshapes the weight magnitude distribution into a heavy-tailed form by penalizing the reciprocal variance of absolute weights. This encourages a small number of dominant weights while suppressing the rest, allowing magnitude pruning to remove low-magnitude parameters with reduced performance degradation.

WCR is compatible with standard SGD and existing pruning-robust optimizers. We evaluate WCR across image classification, medical image segmentation, and LLM fine-tuning, demonstrating consistent improvements in pruning robustness across architectures and tasks. We further provide a theoretical analysis showing that WCR preserves the standard convergence behavior of SGD under mild smoothness assumptions.

Our contributions are summarized as follows.

- We propose Weight Concentration Regularizer (WCR), a training-time regularizer that improves one-shot pruning robustness by concentrating weight energy onto a small subset of parameters.
- We demonstrate that WCR consistently improves pruning robustness across image classification, medical image segmentation, and LLM fine-tuning, while composing effectively with existing pruning-robust optimizers.

2. Related works

Neural Network Pruning and One-Shot Pruning. Neural network pruning removes less important parameters to reduce computational cost while preserving performance (Hoefler et al., 2021). Early methods used second-order sensitivity information (LeCun et al., 1990; Hassibi & Stork, 1993), while later approaches commonly relied on magnitude-based pruning (Han et al., 2015; Li et al., 2017; He et al., 2017). Recent one-shot pruning methods focus on improving robustness without retraining (Liu et al., 2019). Sparsity-inducing regularizers such as L_1 and DeepHoyer (Yang et al., 2020) shape the weight distribution during training, while pruning-robust optimizers including SAM (Foret et al., 2021), CrAM (Chen et al., 2023), and S^2SAM (Kwon et al., 2022) improve robustness through flatter optimization landscapes (Na et al., 2022). In contrast, our work explicitly concentrates weight energy onto a small subset of parameters, producing a clearer separation between important and removable weights under magnitude pruning.

Weight Diversity and Structural Robustness. Prior studies have explored maintaining weight diversity and effective rank to improve model stability and generalization. Methods such as Soft Orthogonality (SO), Double Soft Orthogonality (DSO), and SRIP (Xie et al., 2017; Bansal et al., 2018; Candes & Tao, 2005) regularize the weight structure through orthogonality constraints. Unlike these approaches, our method directly reshapes the weight magnitude distribution for improved pruning robustness.

Extended related works on pruning methods, initialization-based pruning, and diversity-oriented regularization are provided in Appendix A.

3. Method

In this study, we introduce Weight Concentration Regularizer (WCR), a training-time regularizer that encourages weight magnitude to concentrate on a small subset of parameters while driving the rest toward zero. As a result, most of the weight energy becomes concentrated on a few high-magnitude parameters, while the remaining parameters become negligible. Under this distribution, magnitude pruning predominantly removes low-magnitude parameters while preserving the dominant weights.

Preliminaries. We denote the training dataset as $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{M}$, where each input $x_i \in \mathbb{R}^k$ and label y_i belongs to the label space $\mathcal{Y}$. A neural network is parameterized by weights $w \in \mathbb{R}^d$, and its output for input x is written as $f(x; w)$. The empirical loss is $L(w) =$

$\frac{1}{M}\sum_{i=1}^{M}\ell(f(x_i;w),y_i)$, where $\ell(\cdot,\cdot)$ denotes a loss function. We let $\nabla L(w)$ denote the full gradient at w, and $\nabla L_t(w)$ the stochastic gradient computed from a minibatch at iteration t.

Weight Concentration Regularizer (WCR). WCR concentrates weight energy by enlarging the variance of absolute weights during training. To target the main trainable weight tensors, WCR is applied to tensors of dimension greater than one (convolutional kernels and fully-connected weights) and excludes one-dimensional vectors such as biases and batch-norm parameters. Let $\mathcal{W} = \{w^{(1)}, \dots, w^{(L)}\}$ be the selected layer-wise weights. For each layer $w^{(\ell)} = (w_1^{(\ell)}, \dots, w_{n_\ell}^{(\ell)})$, we apply a smooth absolute-value transform for differentiability at the origin:

$$\tilde{w}_i^{(\ell)} = \sqrt{(w_i^{(\ell)})^2 + r} \approx |w_i^{(\ell)}|, \qquad r = 10^{-8}. \quad (1)$$

The variance is computed per-layer to handle scale differences across layers, $\mathrm{Var}(\tilde{w}^{(\ell)}) = \frac{1}{n_\ell}\sum_{i=1}^{n_\ell}(\tilde{w}_i^{(\ell)} - \frac{1}{n_\ell}\sum_{j=1}^{n_\ell}\tilde{w}_j^{(\ell)})^2$, and the penalty is the sum of layer-wise reciprocal variances:

$$\psi(w) = \sum_{w^{(\ell)}\in\mathcal{W}} \frac{1}{\mathrm{Var}(\tilde{w}^{(\ell)}) + \epsilon}, \quad (2)$$

with $\epsilon = 10^{-8}$ for numerical stability. Since $\psi(w)$ depends only on weights, $\nabla\psi(w)$ is deterministic and does not contribute to stochastic gradient variance. The total training objective is

$$L_{total}(w) = L(w) + \lambda\psi(w), \quad (3)$$

where $\lambda > 0$ is a balancing coefficient. Optimization follows the standard SGD update $w_{t+1} = w_t - \eta(\nabla L_t(w_t) + \lambda\nabla\psi(w_t))$, with $\eta > 0$ the learning rate. Full theoretical analysis is provided in Appendix B.

How WCR concentrates weight energy. The penalty $\psi(w)$ admits a closed-form expression that provides insight into how WCR promotes both sparsity and weight concentration. For each layer, the variance of absolute weights can be written as

$$\mathrm{Var}(\tilde{w}^{(\ell)}) = \frac{\|\tilde{w}^{(\ell)}\|_2^2}{n_\ell} - \frac{\|\tilde{w}^{(\ell)}\|_1^2}{n_\ell^2} \quad (4)$$

$$= \frac{\|\tilde{w}^{(\ell)}\|_2^2 \cdot \big(n_\ell - H(\tilde{w}^{(\ell)})\big)}{n_\ell^2} \quad (5)$$

where $H(u) := \|u\|_1^2/\|u\|_2^2 \in [1, n_\ell]$ is the Hoyer sparsity measure.

The corresponding penalty becomes

$$\frac{1}{\mathrm{Var}(\tilde{w}^{(\ell)}) + \epsilon} = \frac{n_\ell^2}{\|\tilde{w}^{(\ell)}\|_2^2 \cdot \big(n_\ell - H(\tilde{w}^{(\ell)})\big) + n_\ell^2\epsilon} \quad (6)$$

This expression highlights that the penalty depends jointly on the total L_2 energy $\|\tilde{w}^{(\ell)}\|_2^2$ and the sparsity gap $n_\ell - H(\tilde{w}^{(\ell)})$. Increasing either term enlarges the denominator, thereby reducing the penalty. In practice, this encourages a distribution in which a small subset of parameters carries most of the weight energy while the remaining parameters become small.

Unlike scale-invariant sparsity regularizers such as DeepHoyer (Yang et al., 2020), which primarily act on the sparsity structure, WCR also depends on the overall energy scale. Empirically, this leads to weight distributions where a small number of parameters dominate the total energy.

4. Experimental results

4.1. Experimental setup

We design our experiments to evaluate two roles of WCR. First, we test whether WCR improves pruning robustness and composes with pruning-robust optimizers in vision tasks. This setting evaluates whether weight-distribution shaping provides gains beyond loss-landscape smoothing methods such as SAM (Foret et al., 2021), CrAM (Chen et al., 2023), and S^{2}SAM (Kwon et al., 2022).

Second, we evaluate WCR as a standalone weight-shaping regularizer in LLM fine-tuning. Since AdamW + LoRA is the standard optimization pipeline for parameter-efficient fine-tuning, we compare WCR against ℓ_1 and DeepHoyer (Yang et al., 2020) under the same optimizer and pruning protocol, isolating the effect of the regularization mechanism itself.

Datasets and models. For image classification, we train ResNet-18 (He et al., 2016) on CIFAR-10 (Krizhevsky, 2009), ResNet-50 on SVHN (Netzer et al., 2011), WideResNet-28-10 (Zagoruyko & Komodakis, 2016) on CIFAR-100, and ImageNet-pretrained ViT-B/32 (Dosovitskiy et al., 2021) on CIFAR-100 and Tiny-ImageNet (CS231N, 2015). For segmentation, we train ResNet-50–UNet (Ronneberger et al., 2015) on the LGG Brain MRI dataset (Buda et al., 2019).

For LLM fine-tuning, we fine-tune Qwen-2.5-1.5B (Yang et al., 2024) and LLaMA-3.2-1B (Grattafiori et al., 2024) on Commonsense-170K following the train/test protocol of DoRA (Liu et al., 2024). The training set is the union of eight commonsense reasoning datasets: BoolQ, PIQA, SIQA, HellaSwag, WinoGrande, ARC-Easy, ARC-Challenge, and OBQA. We evaluate on the corresponding test splits.

Training setup. All vision models are trained for 200 epochs with batch size 128 for classification and 64 for segmentation. We set λ=10^{-5} for CNNs and λ=10^{-2} for ViT-B/32. Classification follows the original

Table 1. Accuracy (%) for CNNs (CIFAR-10/100, SVHN) and ViT-B/32 (CIFAR-100, Tiny-ImageNet), without ('w/o') and with ('w') WCR. Results are averaged over 3 random seeds. SFW is not applicable to ViT-B/32.

		Dense		92% Pruned		94% Pruned		96% Pruned	
Setting	**Method**	**w/o**	**w**	**w/o**	**w**	**w/o**	**w**	**w/o**	**w**
ResNet-18 CIFAR-10	SGD	92.75	92.63	31.60	80.24	26.10	59.91	19.69	25.70
	SFW	92.54	92.34	92.54	92.32	92.22	92.29	84.15	89.93
	CrAM	94.19	93.86	86.54	92.14	80.40	89.47	66.66	81.72
	SAM	95.19	94.08	87.20	93.91	76.29	93.77	48.36	92.08
	S^2SAM	95.23	93.71	91.02	93.67	87.21	93.59	72.51	93.44
ResNet-50 SVHN	SGD	94.44	94.51	9.16	12.32	9.15	10.84	9.15	10.05
	SFW	95.79	95.80	95.82	95.79	95.73	95.77	18.20	95.49
	CrAM	95.97	95.79	78.51	90.51	53.30	82.65	27.78	31.26
	SAM	96.96	95.98	73.37	95.98	24.98	95.98	9.74	95.99
	S^2SAM	96.88	95.78	73.92	95.78	39.55	95.78	17.28	95.78
WideRes-28-10 CIFAR-100	SGD	75.93	75.82	2.86	10.88	1.00	4.00	1.00	1.00
	SFW	63.01	64.91	63.0	64.90	59.59	62.18	1.00	18.77
	CrAM	76.29	75.86	25.87	55.47	7.43	31.56	3.06	7.73
	SAM	78.43	79.51	4.26	77.95	1.94	73.97	1.00	56.56
	S^2SAM	79.30	79.30	6.98	77.33	3.96	73.95	1.13	58.19

		Dense		60% Pruned		70% Pruned		80% Pruned	
Dataset	**Method**	**w/o**	**w**	**w/o**	**w**	**w/o**	**w**	**w/o**	**w**
ViT-B/32 CIFAR-100	SGD	89.84	89.75	86.30	88.58	78.33	85.66	51.13	73.69
	CrAM	88.71	87.82	86.33	87.08	82.80	85.47	70.31	79.29
	SAM	87.36	87.82	84.83	87.82	80.29	87.97	64.11	84.95
	S^2SAM	88.06	87.90	84.82	86.70	81.95	86.80	69.58	85.25
ViT-B/32 Tiny-ImageNet	SGD	86.89	86.72	82.90	85.43	73.79	79.23	39.59	60.15
	CrAM	87.08	87.12	83.29	84.46	75.36	77.14	47.25	57.12
	SAM	87.29	87.11	82.79	85.77	73.22	79.82	41.57	59.40
	S^2SAM	87.20	87.31	82.62	84.59	72.86	79.90	41.12	59.37

SFW/CrAM/SAM/S^2SAM setups. For segmentation and ViT-B/32, we use an initial learning rate of 0.001 with $0.5\times$ step decay every 50 epochs, with early stopping for segmentation.

For LLMs, we fine-tune with LoRA using rank r=32, α=64, and dropout 0.05, applied to all attention and MLP projections (`q_proj`, `k_proj`, `v_proj`, `o_proj`, `gate_proj`, `up_proj`, `down_proj`). Training uses AdamW with β_1=0.9, β_2=0.999, weight decay 0, batch size 16, gradient accumulation 4, and learning rate 2×10^{-4} with a linear schedule and 100 warmup steps. We train for 2 epochs with cutoff length 256. For both Qwen and LLaMA models, λ=10^{-7} is used for ℓ_1, while λ=10^{-6} is used for DeepHoyer and WCR. Full λ tuning results are provided in Appendix E.4.

One-shot pruning. All pruning is performed after training, with no retraining or fine-tuning. CNNs use global ℓ_1 magnitude pruning over convolution weights. ViT-B/32 is pruned uniformly across Q, K, V, projection, and FFN layers. For LLMs, we apply ℓ_1 magnitude pruning at $\{0\%, 20\%, 30\%\}$ over the LoRA-adapted attention and MLP projections.

Evaluation metrics. For image classification, we report validation accuracy. For segmentation, we report F1, Tversky index, and Hausdorff distance. For LLM fine-tuning, we report test accuracy on each commonsense reasoning dataset and their average.

4.2. Enhancing Pruning-Robust Optimizers in Vision Tasks

4.2.1. Image classification

CNN full training. Table 1 reports CNN classification results on CIFAR-10, SVHN, and CIFAR-100 under one-shot pruning. Adding WCR to SGD, SFW, CrAM, SAM, and S^2SAM consistently improves post-pruning accuracy, with the largest gains appearing at high sparsity. At 96% pruning on CIFAR-10, SAM improves from 48.36 to 92.08, while S^2SAM improves from 72.51 to 93.44. On SVHN, SAM improves from 9.74 to 95.99 at 96% pruning. On CIFAR-100, S^2SAM improves from 1.13 to 58.19. We additionally provide an ablation study on SAM combined with DeepHoyer, ℓ_1, and WCR under different regularization coefficients λ in Appendix E.3. These results suggest that WCR complements existing pruning-robust optimizers by improving the underlying weight distribution during training.

ViT fine-tuning. Table 1 also reports ViT-B/32 results on CIFAR-100 and Tiny-ImageNet. WCR consistently improves transformer-based vision models under pruning, showing that the effect generalizes beyond CNNs. At 80% pruning on CIFAR-100, SAM improves from 64.11 to 84.95, while on Tiny-ImageNet it improves from 41.57 to 59.40. Similar improvements are observed for CrAM and S^2SAM across pruning ratios.

Table 2. Segmentation on LGG MRI (ResNet-50–UNet) at dense, 85%, and 88% pruning. Results are averaged over 3 random seeds. "–" indicates SGD failed to produce segmentation.

	Pruning Ratio	Dense			85% Pruned			88% Pruned		
Model	**Method**	**F1**	**Tversky**	**H-Dist.**	**F1**	**Tversky**	**H-Dist.**	**F1**	**Tversky**	**H-Dist.**
Res50-Unet	SGD	0.9284	0.9439	12.55	0.3083	0.3848	108.78	-	-	-
	SGD + WCR	0.9258	0.9377	13.14	0.5136	0.5294	97.41	0.5280	0.5253	87.55
	SFW	0.9295	0.9452	9.18	0.5017	0.4480	89.97	0.1435	0.1100	84.42
	SFW + WCR	0.9191	0.9346	13.84	0.7938	0.7928	32.61	0.7210	0.6909	31.78
	CrAM	0.9163	0.9366	19.11	0.1877	0.2758	110.31	0.2811	0.2888	105.89
	CrAM + WCR	0.8994	0.9245	21.12	0.4066	0.5005	106.17	0.3264	0.3345	103.99
	SAM	0.9264	0.9378	9.26	0.3903	0.4399	107.11	0.2786	0.2587	100.13
	SAM + WCR	0.9267	0.9442	9.02	0.7244	0.8024	54.07	0.7018	0.7791	58.12
	S^2SAM	0.9228	0.9399	10.62	0.3588	0.4058	106.96	0.3350	0.3689	106.76
	S^2SAM + WCR	0.9294	0.9459	9.00	0.5699	0.6818	72.46	0.5447	0.6567	75.83

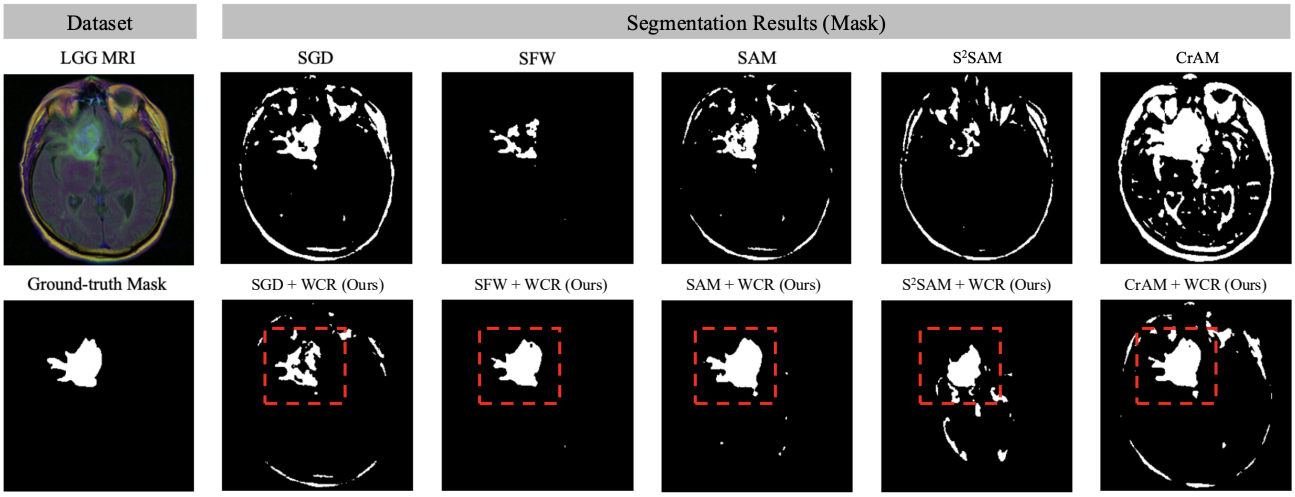

Figure 3. Qualitative segmentation on LGG MRI at 85% pruning, with and without WCR. Additional examples are provided in Appendix E.5.

4.2.2. Segmentation Results

We further evaluate WCR in medical segmentation, where high sparsity can severely degrade dense predictions. Table 2 reports LGG Brain MRI segmentation with ResNet-50–UNet at Dense, 85%, and 88% pruning. WCR consistently improves F1 and Tversky scores and reduces Hausdorff distance after pruning. At 85% pruning, SAM improves from 0.3903 to 0.7244 F1 with WCR. At 88% pruning, SGD recovers from failure to 0.5280 F1. Figure 3 qualitatively shows that WCR better preserves lesion regions, while baselines often produce fragmented or empty masks.

4.3. LLM Fine-Tuning: Comparison with Weight-Shaping Regularizers

We evaluate WCR in LoRA-based LLM fine-tuning under a fixed AdamW + LoRA pipeline on Qwen-2.5-1.5B and LLaMA-3.2-1B fine-tuned on Commonsense-170K, and compare against ℓ_1 and DeepHoyer regularization. The regularization coefficients are selected through a λ sweep on Qwen-2.5-1.5B and then applied to LLaMA-3.2-1B. Full tuning results are provided in Appendix E.4.

Table 3 reports per-task and average accuracy under one-shot magnitude pruning at Dense, 20%, and 30% pruning ratios. Across both models, WCR remains competitive after pruning and achieves the highest average accuracy at 30% pruning among the compared regularizers. These results suggest that encouraging concentrated weight distributions during LoRA fine-tuning can improve robustness under one-shot magnitude pruning.

4.4. Generalization analysis

We measure the principal Hessian eigenvalue, a standard proxy for loss-surface curvature (Lee et al., 2023; Keskar et al., 2016). Figure 4 shows that WCR lowers this curvature for both SGD and SAM on CIFAR-10/ResNet-18 while improving validation accuracy, suggesting that WCR is associated with flatter solutions in this setting. We use a longer training schedule (300 epochs) for this analysis to

Table 3. Accuracy (%) on eight commonsense reasoning benchmarks for Qwen-2.5-1.5B and LLaMA-3.2-1B fine-tuned with LoRA on Commonsense-170K, under one-shot magnitude pruning at dense, 20%, and 30% pruning ratios.

Model	Pruning	Method	BoolQ	PIQA	SIQA	ARC-c	ARC-e	OBQA	HellaS	WinoG	AVG
Qwen-2.5-1.5B	Dense	AdamW	43.91	81.99	72.47	69.45	83.33	81.00	80.04	68.82	72.63
		AdamW + ℓ_1	65.05	80.90	74.05	65.44	79.88	75.60	75.80	73.40	73.77
		AdamW + DeepHoyer	39.08	83.73	75.54	74.49	89.02	83.60	87.30	74.66	75.93
		AdamW + WCR	65.96	80.85	74.41	69.37	85.35	79.60	84.05	76.01	76.95
	20%	AdamW	62.23	56.96	64.33	69.45	84.09	79.80	82.87	73.24	71.62
		AdamW + ℓ_1	64.43	78.78	73.23	63.48	78.16	76.40	73.03	71.19	72.34
		AdamW + DeepHoyer	37.83	81.34	75.33	72.87	87.04	83.20	85.44	72.06	74.39
		AdamW + WCR	65.78	80.25	73.90	68.09	83.75	77.00	81.22	72.45	75.31
	30%	AdamW	62.17	46.68	62.23	63.91	80.39	74.20	78.18	64.33	66.51
		AdamW + ℓ_1	63.09	75.19	67.81	57.68	71.42	70.00	67.31	67.32	67.48
		AdamW + DeepHoyer	60.76	59.19	67.09	62.46	81.19	75.40	61.06	64.88	66.50
		AdamW + WCR	65.66	71.06	71.44	60.24	76.89	70.40	70.85	69.77	69.54
LLaMA-3.2-1B	Dense	AdamW	64.01	76.50	72.98	52.05	71.46	68.00	66.19	68.11	67.41
		AdamW + ℓ_1	60.95	66.00	67.35	49.23	65.45	62.20	52.44	64.88	61.06
		AdamW + DeepHoyer	63.33	75.90	72.88	52.82	70.96	67.20	68.46	70.17	67.71
		AdamW + WCR	62.35	73.72	70.57	50.09	67.34	64.40	60.75	67.40	64.58
	20%	AdamW	61.44	67.74	66.17	42.06	58.16	54.60	48.70	61.56	57.55
		AdamW + ℓ_1	50.64	29.76	51.64	43.86	56.52	44.00	39.15	53.59	46.14
		AdamW + DeepHoyer	61.35	71.76	70.37	48.38	64.14	62.60	53.25	64.01	61.98
		AdamW + WCR	62.20	73.72	69.19	46.08	63.01	60.00	59.97	64.56	62.34
	30%	AdamW	60.55	55.60	52.30	32.59	41.12	23.20	26.32	52.96	43.08
		AdamW + ℓ_1	49.79	29.54	11.16	10.67	12.08	7.40	24.95	51.22	24.60
		AdamW + DeepHoyer	57.52	55.39	49.64	30.97	47.22	45.60	11.01	56.12	44.18
		AdamW + WCR	61.47	63.06	59.98	32.08	47.73	46.60	10.40	59.59	47.61

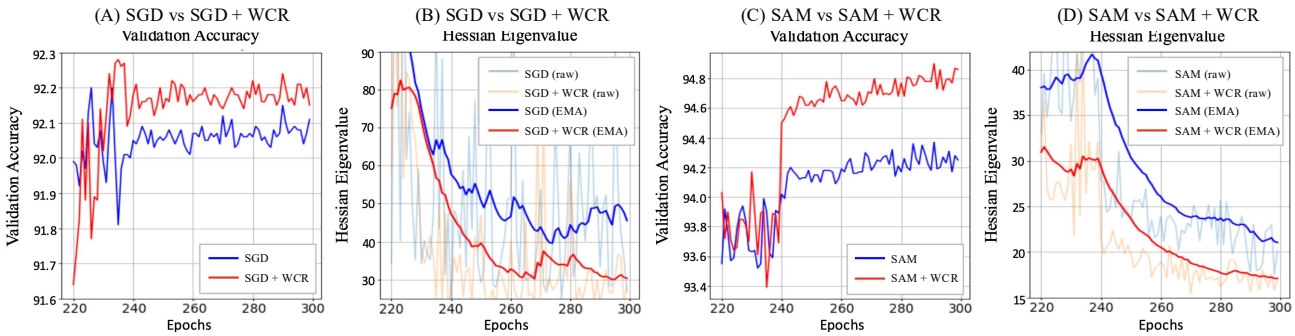

Figure 4. SGD/SAM with and without WCR on CIFAR-10/ResNet-18 (batch 1024, 300 epochs). (A,C) accuracy; (B,D) principal Hessian eigenvalue. WCR lowers curvature and raises accuracy.

better assess generalization behavior and stability.

5. Conclusion

This work introduces the Weight Concentration Regularizer (WCR), a training-time regularizer that improves pruning robustness without modifying standard optimization pipelines. WCR encourages a weight distribution in which a small subset of parameters carries most of the weight energy while the remaining weights are driven toward near-zero values, producing a clearer separation between large- and small-magnitude parameters. Under this distribution, magnitude-based pruning can remove many low-magnitude parameters with reduced performance degradation. Empirically, WCR consistently improves pruning robustness across image classification, medical segmentation, and LoRA-based LLM fine-tuning, while composing effectively with pruning-robust optimizers. Overall, our results suggest that shaping the weight magnitude distribution during training is an effective approach for improving robustness under high sparsity.

Limitations and future work. Our theoretical analysis relies on a smoothness assumption on the concentration objective, and extending the analysis beyond this setting remains future work. Although we observe stable training behavior across models and datasets, our LLM experiments focus only on the standard AdamW + LoRA pipeline and do not include SAM-family optimizers due to their additional overhead in parameter-efficient fine-tuning. Extending WCR to LLM-scale pruning-robust optimization and structured pruning remains an important direction.

References

Bansal, N., Chen, X., and Wang, Z. Can we gain more from orthogonality regularizations in training deep cnns? In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, 2018.

Buda, M., Saha, A., and Mazurowski, M. A. Association of genomic subtypes of lower-grade gliomas with shape features automatically extracted by a deep learning algorithm. *Computers in Biology and Medicine*, 2019.

Candes, E. J. and Tao, T. Decoding by linear programming. *IEEE transactions on information theory*, 51(12):4203–4215, 2005.

Chen, L., Li, X., and Hu, X. Cram: Sharpness-aware minimization for efficient model compression. In *International Conference on Learning Representations*, 2023.

CS231N, S. Tiny imagenet visual recognition challenge, 2015.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

Foret, P., Kleiner, A., Mobahi, H., and Neyshabur, B. Sharpness-aware minimization for efficiently improving generalization. In *International Conference on Learning Representations*, 2021.

Frankle, J. and Carbin, M. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations*, 2019.

Gao, Z., Zhang, Y., Lu, W., Ma, M., Hu, X., and Cheng, M.-M. Bilevelpruning: Unified dynamic and static channel pruning for convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18873–18883, 2024.

Grattafiori, A., Dubey, A., Jauhri, A., Pandey, A., Kadian, A., Al-Dahle, A., Letman, A., Mathur, A., Schelten, A., Vaughan, A., et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Han, S., Pool, J., Tran, J., and Dally, W. Learning both weights and connections for efficient neural network. In *Advances in Neural Information Processing Systems*, 2015.

Hassibi, B. and Stork, D. G. Second order derivatives for network pruning: Optimal brain surgeon. *Advances in Neural Information Processing Systems (NeurIPS)*, 5:164–171, 1993.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

He, Y., Zhang, X., and Sun, J. Channel pruning for accelerating very deep neural networks. In *International Conference on Computer Vision*, 2017.

Hoefler, T., Alistarh, D., Ben-Nun, T., Dryden, N., and Peste, A. Sparsity in deep learning: pruning and growth for efficient inference and training in neural networks. In *The Journal of Machine Learning Research*, 2021.

Keskar, N. S., Mudigere, D., Nocedal, J., Smelyanskiy, M., and Tang, P. T. P. On large-batch training for deep learning: Generalization gap and sharp minima. *arXiv preprint arXiv:1609.04836*, 2016.

Krizhevsky, A. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Kwon, J., Park, J., Kim, D., and Choi, J. S2sam: Stable and sharpness-aware model for pruning and quantization. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2022.

LeCun, Y., Denker, J. S., and Solla, S. A. Optimal brain damage. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 2, pp. 598–605. Morgan Kaufmann, 1990.

Lee, N., Ajanthan, T., and Torr, P. H. Snip: Single-shot network pruning based on connection sensitivity. In *International Conference on Learning Representations*, 2019.

Lee, S., He, C., and Avestimehr, S. Achieving small-batch accuracy with large-batch scalability via hessian-aware learning rate adjustment. *Neural Networks*, 158:1–14, 2023.

Li, H., Kadav, A., Durdanovic, I., Samet, H., and Graf, H. P. Pruning filters for efficient convnets. In *International Conference on Learning Representations*, 2017.

Liu, S.-Y., Wang, C.-Y., Yin, H., Molchanov, P., Wang, Y.-C. F., Cheng, K.-T., and Chen, M.-H. Dora: Weight-decomposed low-rank adaptation. *arXiv preprint arXiv:2402.09353*, 2024.

Liu, Z., Sun, M., Zhou, T., Huang, G., and Darrell, T. Rethinking the value of network pruning. In *International Conference on Learning Representations*, 2019.

Lu, M., Luo, X., Chen, T., Chen, W., Liu, D., and Wang, Z. Learning pruning-friendly networks via frank-wolfe: One-shot, any-sparsity, and no retraining. In *International Conference on Learning Representations*, 2022.

Na, C., Mehta, S. V., and Strubell, E. Train flat, then compress: Sharpness-aware minimization learns more compressible models. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pp. 4909–4936, December 2022.

Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., and Ng, A. Y. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

Ronneberger, O., Fischer, P., and Brox, T. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, 2015.

Tanaka, H., Kunin, D., Yamins, D. L. K., and Ganguli, S. Pruning neural networks without any data by iteratively conserving synaptic flow. In *Advances in Neural Information Processing Systems*, 2020.

Wang, C., Zhang, G., and Grosse, R. Picking winning tickets before training by preserving gradient flow. In *International Conference on Learning Representations*, 2020.

Xie, D., Xiong, J., and Pu, S. All you need is beyond a good init: Exploring better solution for training extremely deep convolutional neural networks with orthonormality and modulation. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

Yang, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Li, C., Liu, D., Huang, F., Wei, H., Lin, H., Yang, J., Tu, J., Zhang, J., Yang, J., Yang, J., Zhou, J., Lin, J., Dang, K., Lu, K., Bao, K., Yang, K., Yu, L., Li, M., Xue, M., Zhang, P., Zhu, Q., Men, R., Lin, R., Li, T., Tang, T., Xia, T., Ren, X., Ren, X., Fan, Y., Su, Y., Zhang, Y., Wan, Y., Liu, Y., Cui, Z., Zhang, Z., and Qiu, Z. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

Yang, H., Wen, W., and Li, H. DeepHoyer: Learning sparser neural network with differentiable scale-invariant sparsity measures. In *International Conference on Learning Representations (ICLR)*, 2020.

Yao, Z., Gholami, A., Keutzer, K., and Mahoney, M. W. Pyhessian: Neural networks through the lens of the hessian. *arXiv preprint arXiv:1912.07145*, 2019.

Zagoruyko, S. and Komodakis, N. Wide residual networks. In *British Machine Vision Conference (BMVC)*, 2016.

Appendix

This appendix provides additional materials that complement the main paper. It includes the full theoretical analysis supporting the convergence behavior of our Weight Concentration Regularizer (WCR), extended implementation details for all experiments, and additional qualitative and quantitative results that further illustrate the effect of WCR. The appendix is organized as follows:

- **Appendix** A: Extended Related Works
- **Appendix** B: Theoretical analysis and proofs for the proposed WCR.
- **Appendix** C: PyTorch implementation of the Weight Concentration Regularizer.
- **Appendix** D: Experimental setup, implementation details, and evaluation metrics.
- **Appendix** E: Extended experimental results.

A. Broader impact

Neural Network Pruning. Model pruning methods commonly drop a subset of model parameters to enable neural network training on resource-constrained systems (Hoefler et al., 2021). Early pruning approaches, such as Optimal Brain Damage (LeCun et al., 1990) and Optimal Brain Surgeon (Hassibi & Stork, 1993) pruned parameters based on the second-order sensitivity information to minimize the increase in loss. Recently, several studies have proposed parameter magnitude-based pruning methods, typically using an L_1-norm criterion to remove parameters with small magnitudes. Han et al. (Han et al., 2015) demonstrated that such unstructured, magnitude-based methods can achieve high sparsity but often require fine-tuning. In contrast, structured pruning methods remove entire channels or filters to reduce computational cost while preserving the key structural characteristics of model parameters (Li et al., 2017; He et al., 2017). More recent studies have explored dynamic and adaptive pruning approaches (Gao et al., 2024), which adjust sparsity during training to balance compactness and performance.

One-Shot Pruning. One-shot pruning methods have been highlighted recently, which remove less important parameters in a single step after training (Liu et al., 2019). A common direction is to shape the weight distribution during training through sparsity-inducing regularization. L_1 regularization shrinks all weights uniformly toward zero, while DeepHoyer (Yang et al., 2020) uses the scale-invariant ratio $\|w\|_1/\|w\|_2$ to protect large weights from shrinkage. Both target sparsity without concentrating weight energy on important parameters; in contrast, our regularizer concentrates weight energy on a small subset of weights while pushing the rest toward zero. SFW (Lu et al., 2022) formulates pruning as a constrained optimization problem with a K-sparse constraint. Recently, Sharpness-Aware Minimization (SAM) (Foret et al., 2021) is known to not only improve generalization but also make models more robust to model pruning (Na et al., 2022). CrAM (Chen et al., 2023) further enhances compression efficiency through perturbation-based optimization, and S^2SAM (Kwon et al., 2022) achieves similar effects with a single-step approximation. These methods improve pruning robustness mainly through loss-landscape smoothing and perturbation-based optimization. In contrast, our work focuses on reshaping the weight distribution during training, and can be combined with these approaches. There have been several initialization-based pruning methods. SNIP (Lee et al., 2019) identifies important connections by measuring each weight's influence on the loss using a small batch of data, whereas GraSP (Wang et al., 2020) preserves gradient flow to maintain signal propagation during the early training stage. SynFlow (Tanaka et al., 2020) selects critical weights by maximizing synthetic information flow based on absolute weight values. Although these methods are effective for early-stage sparsification, they do not explicitly consider improving pruning robustness during training. Therefore, we exclude these initialization-based methods from our comparison.

Weight Diversity and Structural Robustness. Maintaining weight diversity which can be represented as rank has been shown to improve model stability and generalization. Soft Orthogonality (SO) (Xie et al., 2017) and Double Soft Orthogonality (DSO) (Bansal et al., 2018) preserve the effective rank of weights by constraining their Gram matrix toward the identity, but they rely on costly matrix-level computations that limit scalability. SRIP (Bansal et al., 2018; Candes & Tao, 2005) minimizes the spectral norm of the difference between the Gram matrix of a weight matrix and the identity matrix. While these methods improve model generalization, they do not explicitly encourage concentrated weight distributions for pruning. In contrast, our work concentrates weight magnitude onto a small subset of parameters during training, reshaping the weight distribution so that magnitude pruning preferentially removes low-magnitude weights.

B. Theoretical Analysis

In this section, we provide the detailed proofs of the lemmas and corollaries presented in the main paper. Specifically, we establish that adding the Weight Concentration Regularizer (WCR) preserves the convergence properties of stochastic gradient descent under standard smoothness and bounded-variance assumptions.

Assumption B.1 (β_1-smoothness)**.** The loss function $L : \mathbb{R}^d \to \mathbb{R}$ is β_1-smooth, i.e., its gradient is β_1-Lipschitz continuous $\|\nabla L(u) - \nabla L(v)\| \leq \beta_1 \|u - v\|, \forall u, v \in \mathbb{R}^d$. Equivalently, for any $u, v \in \mathbb{R}^d$, the following descent lemma holds:

$$L(u) \leq L(v) + \langle \nabla L(v), u - v \rangle + \tfrac{\beta_1}{2} \|u - v\|^2. \tag{7}$$

Assumption B.2 (Unbiased stochastic gradient)**.** At each iteration t, the stochastic gradient $\nabla L_t(w_t)$ is an unbiased estimator of the true gradient:

$$\mathbb{E}[\nabla L_t(w_t)] = \nabla L(w_t). \tag{8}$$

Assumption B.3 (Local smoothness of $\psi(w)$)**.** The regularizer $\psi : \mathbb{R}^d \to \mathbb{R}$ is continuously differentiable. Moreover, we assume that $\psi(w)$ has a β_2-Lipschitz continuous gradient over a bounded domain $\mathcal{W}$ containing the iterates $\{w_t\}$, i.e.,

$$\|\nabla \psi(w_1) - \nabla \psi(w_2)\| \leq \beta_2 \|w_1 - w_2\|, \quad \forall w_1, w_2 \in \mathcal{W}. \tag{9}$$

We now provide the proofs for all the proposed lemmas and the theorem as follows.

Lemma B.4. *Under the β_1-smoothness of L and the β_2-smoothness of ψ, the combined objective $L_{\text{total}}(w) = L(w) + \lambda\psi(w)$ is β-smooth over $\mathcal{W}$ for $\beta \leq \beta_1 + \lambda\beta_2$. Together with the bounded-variance conditions $\mathbb{E}[\nabla L_t(w_t)] = \nabla L(w_t)$ and $\mathbb{E}\big[\|\nabla L_t(w_t) - \nabla L(w_t)\|^2\big] \leq \frac{\sigma^2}{b}$ (where the regularizer ψ is deterministic, implying $\sigma_\psi = 0$), if the step size satisfies $\eta \leq 1/\beta$, then the classic SGD update $w_{t+1} = w_t - \eta\big(\nabla L_t(w_t) + \lambda \nabla \psi(w_t)\big)$ guarantees:*

$$\mathbb{E}\big[L(w_{t+1}) + \lambda\psi(w_{t+1})\big] \leq \mathbb{E}\big[L(w_t) + \lambda\psi(w_t)\big] - \frac{\eta}{2}\mathbb{E}[\|\nabla L(w_t) + \lambda\nabla\psi(w_t)\|^2] + \frac{\eta^2\beta}{2b}\sigma^2 \tag{10}$$

Proof. From the β-smoothness of L_{total}, we have:

$$\begin{aligned} &L(w_{t+1}) + \lambda\psi(w_{t+1}) \\ &\qquad \leq L(w_t) + \lambda\psi(w_t) + \mathbb{E}[\langle \nabla L(w_t) + \lambda\nabla\psi(w_t),\, w_{t+1} - w_t \rangle] + \frac{\beta}{2}\|w_{t+1} - w_t\|^2 \end{aligned}$$

Substituting the SGD update $w_{t+1} = w_t - \eta(\nabla L(w) + \lambda\nabla\psi(w))$, we obtain:

$$\begin{aligned} &L(w_{t+1}) + \lambda\psi(w_{t+1}) \\ &\qquad \leq L(w_t) + \lambda\psi(w_t) - \eta\,\langle \nabla L(w_t) + \lambda\nabla\psi(w_t),\, \nabla L(w_t) + \lambda\nabla\psi(w_t) \rangle \\ &\qquad\quad + \frac{\eta^2\beta}{2}\|L_t(w_t) + \lambda\psi(w_t)\|^2 \end{aligned}$$

Taking the expectation, and noting that $\mathbb{E}[\nabla L_t(w_t)] = \nabla L(w_t)$ and then $\mathbb{E}[\nabla L_t(w_t) + \lambda\nabla\psi(w)] = \nabla L(w_t) + \lambda\nabla\psi(w_t)$, we get:

$$\begin{aligned} &\mathbb{E}\big[L(w_{t+1}) + \lambda\psi(w_{t+1})\big] \\ &\qquad \leq \mathbb{E}\big[L(w_t) + \lambda\psi(w_t)\big] - \eta\,\mathbb{E}[\langle \nabla L(w_t) + \lambda\nabla\psi(w_t),\, \nabla L(w_t) + \lambda\nabla\psi(w_t) \rangle] \\ &\qquad\quad + \frac{\eta^2\beta}{2}\mathbb{E}[\|\nabla L_t(w_t) + \lambda\nabla\psi(w_t)\|^2] \\ &\qquad = \mathbb{E}\big[L(w_t) + \lambda\psi(w_t)\big] - \eta\,\mathbb{E}[\|\nabla L(w_t) + \lambda\nabla\psi(w_t)\|^2] \\ &\qquad\quad + \frac{\eta^2\beta}{2}\mathbb{E}[\|\nabla L_t(w_t) + \lambda\nabla\psi(w_t)\|^2] \end{aligned}$$

The stochastic gradient variance is bounded as:

$$\begin{aligned}
&\mathbb{E}[L(w_{t+1}) + \lambda\psi(w_{t+1})] \\
&\quad\leq \mathbb{E}[L(w_t) + \lambda\psi(w_t)] - \eta\mathbb{E}[\langle(\nabla L(w_t) + \lambda\nabla\psi(w_t)), (\nabla L_t(w_t) + \lambda\nabla\psi(w_t))\rangle] \\
&\qquad + \frac{\eta^2\beta}{2}\mathbb{E}[\|(\nabla L_t(w_t) + \lambda\nabla\psi(w_t))\|^2] \\
&\quad= \mathbb{E}[(L(w_t) + \lambda\psi(w_t))] - \eta\mathbb{E}[\|(\nabla L(w_t) + \lambda\nabla\psi(w_t))\|^2] \\
&\qquad + \frac{\eta^2\beta}{2}\mathbb{E}[\|(\nabla L_t(w_t) + \lambda\nabla\psi(w_t))\|^2]
\end{aligned}$$

Expanding the variance term, we obtain

$$\begin{aligned}
&\mathbb{E}\big[\|\nabla L_t(w_t) + \lambda\nabla\psi(w_t)\|^2\big] \qquad (11)\\
&\quad= \mathbb{E}\big[\|(\nabla L_t(w_t) + \lambda\nabla\psi(w_t)) - (\nabla L(w_t) + \lambda\nabla\psi(w_t)) + (\nabla L(w_t) + \lambda\nabla\psi(w_t))\|^2\big] \\
&\quad= \mathbb{E}\big[\|\nabla L_t(w_t) - \nabla L(w_t)\|^2\big] + 2\,\mathbb{E}\big[\langle\nabla L_t(w_t) - \nabla L(w_t),\, \nabla L(w_t) + \lambda\nabla\psi(w_t)\rangle\big] \\
&\qquad + \mathbb{E}\big[\|\nabla L(w_t) + \lambda\nabla\psi(w_t)\|^2\big] \qquad (12)
\end{aligned}$$

By the unbiasedness assumption (12) simplifies to

$$\mathbb{E}\big[\|\nabla L_t(w_t) + \lambda\nabla\psi(w_t)\|^2\big] = \mathbb{E}\big[\|\nabla L_t(w_t) - \nabla L(w_t)\|^2\big] + \mathbb{E}\big[\|\nabla L(w_t) + \lambda\nabla\psi(w_t)\|^2\big] \qquad (13)$$

By the gradient-variance bound assumption, we have

$$\begin{aligned}
&\mathbb{E}\big[\|(\nabla L_t(w_t) + \lambda\nabla\psi(w_t)) - (\nabla L(w_t) + \lambda\nabla\psi(w_t))\|^2\big] \qquad (14)\\
&\quad= \mathbb{E}\big[\|\nabla L_t(w_t) - \nabla L(w_t)\|^2\big] \leq \frac{\sigma^2}{b} \qquad (15)
\end{aligned}$$

Combining (13) and (15), we obtain

$$\mathbb{E}\big[\|\nabla L_t(w_t) + \lambda\nabla\psi(w_t)\|^2\big] \leq \mathbb{E}\big[\|\nabla L(w_t) + \lambda\nabla\psi(w_t)\|^2\big] + \frac{\sigma^2}{b} \qquad (16)$$

Substituting the variance bound (13)–(15) into the descent inequality yields:

$$\begin{aligned}
&\mathbb{E}[L(w_{t+1}) + \lambda\psi(w_{t+1})] \\
&\quad\leq \mathbb{E}[L(w_t) + \lambda\psi(w_t)] - \eta\mathbb{E}[\|\nabla L(w_t) + \lambda\nabla\psi(w_t)\|^2] \\
&\qquad + \frac{\eta^2\beta}{2}\left(\mathbb{E}[\|\nabla L(w_t) + \lambda\nabla\psi(w_t)\|^2] + \frac{\sigma^2}{b}\right) \\
&\quad= \mathbb{E}[L(w_t) + \lambda\psi(w_t)] - \eta\mathbb{E}[\|\nabla L(w_t) + \lambda\nabla\psi(w_t)\|^2] \\
&\qquad + \frac{\eta^2\beta}{2}\mathbb{E}[\|\nabla L(w_t) + \lambda\nabla\psi(w_t)\|^2] + \frac{\eta^2\beta\sigma^2}{2b} \\
&\quad= \mathbb{E}[L(w_t) + \lambda\psi(w_t)] + \left(-\eta + \frac{\eta^2\beta}{2}\right)\mathbb{E}[\|\nabla L(w_t) + \lambda\nabla\psi(w_t)\|^2] + \frac{\eta^2\beta\sigma^2}{2b} \\
&\quad= \mathbb{E}[L(w_t) + \lambda\psi(w_t)] - \left(\eta - \frac{\eta^2\beta}{2}\right)\mathbb{E}[\|\nabla L(w_t) + \lambda\nabla\psi(w_t)\|^2] + \frac{\eta^2\beta\sigma^2}{2b}
\end{aligned}$$

Finally, given $\eta \leq \frac{1}{\beta}$, we have $\eta - \frac{\eta^2\beta}{2} = \eta\left(1 - \frac{\eta\beta}{2}\right) \geq \frac{\eta}{2}$ and, since $\eta\beta \leq 1$ implies $1 - \frac{\eta\beta}{2} \geq \frac{1}{2}$.

Therefore,

$$\mathbb{E}\big[L(w_{t+1}) + \lambda\psi(w_{t+1})\big] \leq \mathbb{E}\big[L(w_t) + \lambda\psi(w_t)\big] - \frac{\eta}{2}\mathbb{E}[\|\nabla L(w_t) + \lambda\,\nabla\psi(w_t)\|^2] + \frac{\eta^2\beta}{2b}\sigma^2$$

□

Theorem B.5. *Under the same smoothness and bounded-variance conditions stated in Lemma B.4, and for a fixed step size $0 < \eta \leq 1/\beta$, the SGD update satisfies:*

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}[\|\nabla L(w_t) + \lambda\nabla\psi(w_t)\|^2] \leq \frac{2}{T\eta}\big((L(w_0) + \lambda\psi(w_0))\big) - \frac{2}{T\eta}\big(\mathbb{E}[(L(w_T) + \lambda\psi(w_T))]\big) + \frac{\eta\beta\sigma^2}{2b}$$

Proof. Based on the descent lemma for classic SGD, we have:

$$\mathbb{E}[L(w_{t+1}) + \lambda\psi(w_{t+1})] \leq \mathbb{E}[L(w_t) + \lambda\psi(w_t)] - \frac{\eta}{2}\mathbb{E}[\|\nabla L(w_t) + \lambda\nabla\psi(w_t)\|^2] + \frac{\eta^2\beta\sigma^2}{2b} \tag{17}$$

Averaging (17) across T iterations, we obtain:

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}[L(w_{t+1}) + \lambda\psi(w_{t+1})] \leq \frac{1}{T}\sum_{t=0}^{T-1}(\mathbb{E}[L(w_t) + \lambda\psi(w_t)]) - \frac{\eta}{2T}\sum_{t=0}^{T-1}\mathbb{E}[\|\nabla L(w_t) + \lambda\nabla\psi(w_t)\|^2] + \frac{1}{T}\sum_{t=0}^{T-1}\left(\frac{\eta^2\beta\sigma^2}{2b}\right)$$

Rearranging the terms and substituting $\frac{1}{T}\sum_{t=0}^{T-1}\big(\mathbb{E}[L(w_t)+\lambda\psi(w_t)] - \mathbb{E}[L(w_{t+1})+\lambda\psi(w_{t+1})]\big) = \frac{1}{T}\big(L(w_0)+\lambda\psi(w_0) - \mathbb{E}[L(w_t) + \lambda\psi(w_t)]\big)$, we get:

$$\begin{aligned}\frac{\eta}{2T}\sum_{t=0}^{T-1}\mathbb{E}[\|\nabla L(w_t) + \lambda\nabla\psi(w_t)\|^2] &\leq \frac{1}{T}\sum_{t=0}^{T-1}\big(\mathbb{E}[(L(w_t) + \lambda\psi(w_t))] - \mathbb{E}[L(w_{t+1}) + \lambda\psi(w_{t+1})]\big) + \frac{\eta^2\beta\sigma^2}{2b} \\ &= \frac{1}{T}\big((L(w_0) + \lambda\psi(w_0)) - \mathbb{E}[L(w_T) + \lambda\psi(w_T)]\big) + \frac{\eta^2\beta\sigma^2}{2b}\end{aligned}$$

Dividing both sides by $\frac{\eta}{2}$, we obtain:

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}[\|(\nabla L(w_t) + \lambda\nabla\psi(w_t))\|^2] \leq \frac{2}{T\eta}\big((L(w_0) + \lambda\psi(w_0)) - \mathbb{E}[L(w_T) + \lambda\psi(w_T)]\big) + \frac{\eta\beta\sigma^2}{2b}$$

□

Corollary B.6. *Under the same smoothness and bounded-variance assumptions stated in Lemma B.4, and assuming further that L_{total} is bounded below by $L_{\text{total}}^{\inf} > -\infty$, let the step size be constant $\eta = \frac{c}{\sqrt{T}}$ with $0 < c \leq \frac{1}{\beta}$. Then the SGD iterates satisfy:*

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}[\big\|\nabla L(w_t) + \lambda\nabla\psi(w_t)\big\|^2] \leq \frac{2\big((L(w_0) + \lambda\psi(w_0)) - L_{total}^{\inf}\big)}{c}\,T^{-1/2} + \frac{\beta c}{2b}\sigma^2\,T^{-1/2} \tag{18}$$

Proof. Starting from Theorem 5,

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\big[\|\nabla L(w_t)+\lambda\nabla\psi(w_t)\|^2\big] \tag{19}$$

$$\leq \frac{2}{T\eta}((L(w_0)+\lambda\psi(w_0))-\mathbb{E}[L(w_T)+\lambda\psi(w_T)])+\frac{\beta\eta}{2b}\sigma^2 \tag{20}$$

Because $L(w_T)+\lambda\psi(w_T)\geq L_{\text{total}}^{\inf}$, define $\Delta_0=(L(w_0)+\lambda\psi(w_0))-L_{\text{total}}^{\inf}$. Substituting $\eta\leq\frac{c}{\sqrt{T}}$ yields

$$\begin{aligned}\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\big[\|\nabla L(w_t)+\lambda\nabla\psi(w_t)\|^2\big] &\leq \frac{2\Delta_0}{(c/\sqrt{T})T}+\frac{\beta c}{2b}\sigma^2T^{-1/2}\\ &= \frac{2\Delta_0}{c}\,T^{-1/2}+\frac{\beta c}{2b}\sigma^2\,T^{-1/2}\end{aligned} \tag{21}$$

The overall convergence rate is $\mathcal{O}(\frac{1}{\sqrt{T}})$.

□

Corollary B.7 (Diminishing step size)**.** *Under the same smoothness and bounded-variance assumptions stated in Lemma B.4, and assuming further that the total objective $L(w)+\lambda\psi(w)$ is bounded below by $L_{\text{total}}^{\inf}>-\infty$, consider SGD with a diminishing step-size sequence $(\eta_t)_{t\geq0}$ satisfying:*

$$0<\eta_t\leq\frac{1}{\beta},\quad \sum_{t=0}^{\infty}\eta_t=\infty,\quad \sum_{t=0}^{\infty}\eta_t^2<\infty$$

Then the (step-size)–weighted average of squared gradients vanishes when $\nabla L_{\text{total}}(w_t)=\nabla L(w_t)+\lambda\nabla\psi(w_t)$:

$$\lim_{T\to\infty}\frac{1}{\sum_{t=0}^{T-1}\eta_t}\sum_{t=0}^{T-1}\eta_t\,\mathbb{E}\left[\|\nabla L_{\text{total}}(w_t)\|^2\right]=0$$

In particular,

$$\liminf_{T\to\infty}\min_{0\leq t\leq T-1}\mathbb{E}\left[\|\nabla L_{\text{total}}(w_t)\|^2\right]=0$$

Therefore, there exists a subsequence along which the expected squared gradient converges to 0*.*

Proof. From the β-smoothness of L_{total} and the SGD update $w_{t+1}=w_t-\eta_t(\nabla L_t(w_t)+\lambda\nabla\psi(w_t)))$, the same argument as in the lemma yields, for any t with $\eta_t\leq1/\beta$,

$$\mathbb{E}\big[L(w_{t+1})\big]\;\leq\;\mathbb{E}\big[L(w_t)\big]-\frac{\eta_t}{2}\,\mathbb{E}\big[\|\nabla L(w_t)+\lambda\nabla\psi(w_t)\|^2\big]+\frac{\beta\,\eta_t^2}{2b}\,\sigma^2$$

Summing from $t=0$ to $T-1$ and telescoping gives

$$\begin{aligned}\frac{1}{2}\sum_{t=0}^{T-1}\eta_t\,\mathbb{E}[\|\nabla L(w_t)+\lambda\nabla\psi(w_t)\|^2] &\leq \mathbb{E}\big[L(w_0)\big]-\mathbb{E}\big[L(w_T)\big]+\frac{\beta}{2b}\,\sigma^2\sum_{t=0}^{T-1}\eta_t^2\\ &\leq \big(L(w_0)-L^{\inf}\big)+\frac{\beta}{2b}\,\sigma^2\sum_{t=0}^{T-1}\eta_t^2\end{aligned}$$

Divide both sides by $S_T:=\sum_{t=0}^{T-1}\eta_t$ to obtain

$$\frac{1}{2S_T}\sum_{t=0}^{T-1}\eta_t\,\mathbb{E}[\|\nabla L(w_t)+\lambda\nabla\psi(w_t)\|^2]\leq\frac{(L(w_0)+\lambda\psi(w_0))-(L+\lambda\psi)^{\inf}}{S_T}+\frac{\beta}{2b}\,\sigma^2\frac{\sum_{t=0}^{T-1}\eta_t^2}{S_T}$$

By the step-size conditions, $S_T \to \infty$ and $\sum_{t=0}^{T-1} \eta_t^2 / S_T \to 0$ as $T \to \infty$. Hence the right-hand side tends to 0, which proves

$$\lim_{T\to\infty} \frac{1}{\sum_{t=0}^{T-1} \eta_t} \sum_{t=0}^{T-1} \eta_t \, \mathbb{E}\big[\|\nabla L(w_t) + \lambda \nabla \psi(w_t)\|^2\big] \;=\; 0$$

Finally, since $\min_{0\le t\le T-1} a_t \le \frac{\sum_{t=0}^{T-1} \eta_t a_t}{\sum_{t=0}^{T-1} \eta_t}$ for any nonnegative sequence (a_t), we get

$$\min_{0\le t\le T-1} \mathbb{E}\big[\|\nabla L(w_t) + \lambda \nabla \psi(w_t)\|^2\big] \le \frac{\sum_{t=0}^{T-1} \eta_t \, \mathbb{E}\big[\|\nabla L(w_t) + \lambda \nabla \psi(w_t)\|^2\big]}{\sum_{t=0}^{T-1} \eta_t} \xrightarrow[T\to\infty]{} 0$$

which implies that even with the additional regularizer $\psi(w)$, the expected squared gradient is guaranteed to converge to 0 as $T \to \infty$. □

C. PyTorch Implementation

Figure 5 shows a PyTorch implementation of WCR designed for easy adoption. The regularizer is written as a standalone function that returns a single scalar loss term, so it can be directly added to any training objective as $L_{\text{total}}(w) = L_{\text{task}}(w) + \psi(w)$ without changing the model architecture or the optimizer. In practice, this makes WCR easy to plug into an existing PyTorch training loop with minimal code changes.

```python
def variance_regularization(model, lambda_var=0.01):
    variance_loss = torch.tensor(0.0, device=next(model.parameters()).device)
    lambda_var = float(lambda_var)
    for param in model.parameters():
        if param.requires_grad and len(param.shape) > 1:
            abs_param = param.abs()
            mean_param = abs_param.mean()
            var_param = ((abs_param - mean_param) ** 2).mean()
            variance_loss += lambda_var / (var_param + 1e-8)
    return variance_loss
```

Figure 5. PyTorch implementation of the proposed Weight Concentration Regularizer (WCR). For each layer, the regularizer computes the reciprocal of the variance of the absolute weight values, concentrating weight energy onto a small subset of parameters to improve pruning robustness. The final regularization loss is the sum of these layer-wise terms.

D. Experimental Setups

D.1. Implementation Details

All experiments were conducted on four NVIDIA A40 GPUs (48GB each). Each model was trained on a single GPU, while multiple runs were executed in parallel across the four devices. The full training configurations, including model architectures, datasets, number of epochs, batch sizes, and regularization coefficients, are summarized in Tables 4, 5, and 6.

Table 4. Summary of datasets, network architectures, and training settings used in vision experiments.

Task	Model	Dataset	# Epochs	Batch Size	λ
Classification	ResNet-18 (He et al., 2016)	CIFAR-10 (Krizhevsky, 2009)	200	128	1e-5
	ResNet-50 (He et al., 2016)	SVHN (Netzer et al., 2011)	200	128	1e-5
	WideRes28-10 (Zagoruyko & Komodakis, 2016)	CIFAR-100 (Krizhevsky, 2009)	200	128	1e-5
	ViT-B/32 (Dosovitskiy et al., 2021)	CIFAR-100 (Krizhevsky, 2009)	200	128	1e-2
	ViT-B/32 (Dosovitskiy et al., 2021)	Tiny-ImageNet (CS231N, 2015)	200	128	1e-2
Segmentation	Res50-Unet (Ronneberger et al., 2015)	LGG MRI (Buda et al., 2019)	200	64	1e-5

Table 5. Training configurations for classification and segmentation experiments. "–" denotes settings that are not applicable.

Task	Methods	Model	LR Initial	LR Scheduler	LR Decay Rate	ρ	Momentum
Classification	SGD	Res-18/50, WideRes-28	0.1	Dynamic tuning	0.7	–	0.9
	SFW (Lu et al., 2022)	Res-18/50, WideRes-28	1.0	Dynamic tuning	0.7	–	0.9
	CrAM (Chen et al., 2023)	Res-18/50, WideRes-28	0.1	Dynamic tuning	0.7	0.05	0.9
	SAM (Foret et al., 2021)	Res-18/50, WideRes-28	0.1	Dynamic tuning	0.7	0.5	0.9
	S^2SAM (Kwon et al., 2022)	Res-18/50, WideRes-28	0.1	Dynamic tuning	0.7	0.5	0.9
	SGD	ViT-B/32	0.001	Step decay	0.5	–	0.9
	SFW (Lu et al., 2022)	ViT-B/32	–	–	–	–	–
	CrAM (Chen et al., 2023)	ViT-B/32	0.001	Step decay	0.5	0.05	0.9
	SAM (Foret et al., 2021)	ViT-B/32	0.001	Step decay	0.5	0.25	0.9
	S^2SAM (Kwon et al., 2022)	ViT-B/32	0.001	Step decay	0.5	0.25	0.9
Segmentation	SGD	Res50-Unet	0.01	Step decay	0.5	–	0.9
	SFW (Lu et al., 2022)	Res50-Unet	0.01	Step decay	0.5	–	0.9
	CrAM (Chen et al., 2023)	Res50-Unet	0.01	Step decay	0.5	0.05	0.9
	SAM (Foret et al., 2021)	Res50-Unet	0.01	Step decay	0.5	0.25	0.9
	S^2SAM (Kwon et al., 2022)	Res50-Unet	0.01	Step decay	0.5	0.25	0.9

Table 6. LLM fine-tuning configuration used for Qwen-2.5-1.5B and LLaMA-3.2-1B on Commonsense-170K.

Configuration	Value
Fine-tuning method	LoRA
LoRA rank r	32
LoRA α	64
LoRA dropout	0.05
Target modules	`q_proj`, `k_proj`, `v_proj`, `o_proj`, `gate_proj`, `up_proj`, `down_proj`
Optimizer	AdamW
β_1, β_2	0.9, 0.999
Weight decay	0
Learning rate	2×10^{-4}
LR schedule	Linear schedule with 100 warmup steps
Epochs	2
Batch size	16
Gradient accumulation	4
Cutoff length	256
Regularization coefficient λ	10^{-6}
Sweep values for λ	$\{10^{-5}, 10^{-6}, 10^{-7}\}$
Pruning ratios	Dense, 20%, 30%

For classification tasks, we trained ResNet-18, ResNet-50, WideResNet-28-10, and ViT-B/32 for 200 epochs using a batch size of 128. Segmentation experiments on the LGG MRI dataset used ResNet-50–UNet for 200 epochs with a batch size of 64. For LLM experiments, we fine-tuned Qwen-2.5-1.5B and LLaMA-3.2-1B with LoRA on Commonsense-170K under the configuration in Table 6. The regularization strength λ was selected according to the model and task setting, as summarized in Tables 4 and 6.

Learning Rate Scheduling. The learning rate strategies differ across tasks:

- **Dynamic Tuning (CNN classification).** The learning rate is reduced by a factor of 10 at one-third and two-thirds of the total training epochs. After epoch 20, the learning rate is additionally adjusted every five epochs using a moving-loss comparison: if the recent 5-epoch loss average exceeds the recent 10-epoch average, the learning rate is multiplied by 0.7; otherwise it is slightly increased by multiplying by 1.06.
- **Step Decay (ViT and Segmentation).** For ViT-B/32 and segmentation experiments, we use a step-decay rule, multiplying the learning rate by 0.5 every 50 epochs.
- **Linear Schedule (LLM fine-tuning).** For LLM fine-tuning, we use a linear learning-rate schedule with 100 warmup steps.

All SFW experiments use the official default configuration provided in the public repository, including an initial learning rate

of 1.0, dynamic_change learning rate scheduling, momentum of 0.9, zero weight decay, a k-sparse polytope constraint with $K = 10$ and $K_{\text{frac}} = 0.05$, diameter parameter $\tau = 15$, constraint rescaling mode set to initialization, and the SFW_Init option disabled. For constrained updates, SFW performs projection onto the k-sparse polytope at every optimization step. CrAM, SAM, and S^2SAM follow the default settings specified in their official repositories. However, for ViT-B/32 and ResNet50–UNet, the default perturbation radius $\rho = 0.5$ in SAM and S^2SAM led to unstable training. We therefore selected a smaller ρ by gradually reducing the value and choosing the one that yielded stable training and strong validation performance.

D.2. Benchmark Datasets

This work evaluates WCR across image classification, medical image segmentation, and LLM fine-tuning. All pruning is performed after training or fine-tuning without additional retraining.

Image Classification. We assess pruning robustness on four classification benchmarks:

- **CIFAR-10** (Krizhevsky, 2009): A natural image dataset of 32×32 RGB images containing 10 object categories.
- **CIFAR-100** (Krizhevsky, 2009): An extension of CIFAR-10 with 100 fine-grained categories.
- **SVHN** (Netzer et al., 2011): A real-world digit classification dataset derived from street-view house numbers.
- **Tiny-ImageNet** (CS231N, 2015): A mid-scale ImageNet subset with 64×64 RGB images across 200 categories.

Image Segmentation. We evaluate medical segmentation on:

- **LGG Brain MRI Dataset** (Buda et al., 2019): A collection of brain MRI scans from lower-grade glioma patients, paired with expert-annotated tumor masks for pixel-wise segmentation.

LLM Fine-Tuning. We evaluate LoRA-based LLM fine-tuning on Commonsense-170K, following the protocol of DoRA (Liu et al., 2024). The training set is formed by combining the training splits of eight commonsense reasoning datasets: BoolQ, PIQA, SIQA, HellaSwag, WinoGrande, ARC-Easy, ARC-Challenge, and OBQA. Evaluation is performed on the corresponding test splits.

D.3. Evaluation Metrics

This section summarizes the evaluation metrics used across classification, segmentation, and LLM fine-tuning experiments.

Image Classification. For all classification benchmarks, model performance is measured using validation accuracy.

Image Segmentation. For segmentation on the LGG Brain MRI dataset, we use three complementary metrics:

- **F1-score (F1)**: Measures the harmonic mean of precision and recall, capturing overlap between prediction and ground truth.
- **Tversky Index**: A generalization of the Dice score that asymmetrically penalizes false positives and false negatives.
- **Hausdorff Distance (H-Dist.)**: Quantifies the largest boundary deviation between the predicted mask and ground truth. Lower values indicate more accurate boundary localization.

LLM Fine-Tuning. For LLM fine-tuning, we report test accuracy on each commonsense reasoning benchmark and the average accuracy across all eight tasks.

D.4. One-shot Pruning Method

All pruning is performed in a *one-shot* manner after training or fine-tuning, without any retraining or additional fine-tuning.

For CNNs, convolutional weights are globally pruned using an L_1 magnitude threshold. For ViT-B/32, pruning is applied uniformly across the Query, Key, Value, Projection, and FFN layers. In additional appendix analyses, we also evaluate ViT models under pruning configurations that selectively prune only Q, QK, or QKV components in Section E.2.

For LLMs, we apply L_1 magnitude pruning to the LoRA-adapted attention and MLP projections after fine-tuning. Specifically, pruning is applied to `q_proj`, `k_proj`, `v_proj`, `o_proj`, `gate_proj`, `up_proj`, and `down_proj`. We evaluate dense, 20%, and 30% pruned models.

D.5. Principal Hessian Eigenvalue Estimation

We follow the Hessian eigenvalue computation procedure (Yao et al., 2019). In particular, the top Hessian eigenvalues reported in Section 4.4 are computed using the power-iteration method with Hessian-vector products.

Table 7. Hyperparameters used for estimating the top Hessian eigenvalue $\widehat{\lambda}_{\max}$ via power iteration.

Hyperparameter	Meaning	Value
Max iteration	Number of power-iteration steps T	20
Hessian batch size	Mini-batch size $\lvert\mathcal{B}\rvert$ used for estimation	1024
# of batch average	Number of mini-batches M averaged to reduce noise	12
# Epochs	Number of epochs	300

We use the top Hessian eigenvalue $\widehat{\lambda}_{\max}$ as a curvature-based proxy to characterize the local sharpness of the loss landscape during training. By tracking $\widehat{\lambda}_{\max}$ across epochs, we evaluate whether WCR guides optimization toward flatter regions, supporting the generalization trends observed in Section 4.4.

Let $w \in \mathbb{R}^d$ denote the vector of trainable parameters. For a mini-batch $\mathcal{B}$, let $\mathcal{L}_{\mathcal{B}}(w)$ denote the mini-batch objective, $g_{\mathcal{B}}(w) := \nabla_w \mathcal{L}_{\mathcal{B}}(w)$ the mini-batch gradient, and $H_{\mathcal{B}}(w) := \nabla_w^2 \mathcal{L}_{\mathcal{B}}(w)$ the Hessian.

We estimate $\widehat{\lambda}_{\max}$ by power iteration using Hessian-vector products. The mini-batch objective is

$$\mathcal{L}_{\mathcal{B}}(w) = \frac{1}{|\mathcal{B}|} \sum_{(x_i, y_i) \in \mathcal{B}} \ell(f_w(x_i), y_i). \tag{22}$$

Starting from $v_0 \sim \mathcal{N}(0, I)$ normalized as $v_0 \leftarrow v_0 / \|v_0\|_2$, we iterate for $t = 0, 1, \ldots, T-1$:

$$s_t(w) = g_{\mathcal{B}}(w)^\top v_t, \tag{23}$$

$$u_t(w) = \nabla_w s_t(w) = H_{\mathcal{B}}(w) v_t, \tag{24}$$

$$\lambda_t = \frac{v_t^\top u_t(w)}{v_t^\top v_t} = \frac{v_t^\top H_{\mathcal{B}}(w) v_t}{\|v_t\|_2^2}, \tag{25}$$

$$v_{t+1} = \frac{u_t(w)}{\|u_t(w)\|_2}. \tag{26}$$

We report $\widehat{\lambda}_{\max} := \lambda_{T-1}$. To reduce mini-batch noise, we optionally average over M batches:

$$\widehat{\lambda}_{\max} = \frac{1}{M} \sum_{m=1}^{M} \widehat{\lambda}_{\max}(\mathcal{B}_m). \tag{27}$$

For the Hessian analysis in Section 4.4 and Figure 4, we use the setting in Table 7. All remaining experiments use the training configurations described in Section D.1.

E. Experimental results

This section presents additional experimental results that complement the findings reported in the main paper. We first provide extended ablation studies examining the behavior of the proposed Weight Concentration Regularizer (WCR) under different regularization strengths, architectures, and pruning configurations.

E.1. Additional results with aggressive pruning

Table 8 reports additional classification results conducted at pruning rates higher than those presented in the main paper. These experiments evaluate pruning robustness at extreme sparsity levels (96–98%) on CIFAR-10 and SVHN using ResNet-18 and ResNet-50.

Table 8. Accuracy comparison (%) across high pruning rates and datasets (CIFAR-10, SVHN) using ResNet-18/50, with and without WCR, denoted as "w/o" and "w."

		Dense		**96% Pruned**		**97% Pruned**		**98% Pruned**	
Setting	**Method**	**w/o**	**w**	**w/o**	**w**	**w/o**	**w**	**w/o**	**w**
ResNet-18 CIFAR-10	SGD	92.75	92.63	19.69	25.70	19.22	20.30	15.32	16.31
	SFW	92.54	92.34	84.15	89.93	59.12	66.52	12.61	15.02
	CrAM	94.19	93.86	66.66	81.72	35.00	69.08	17.27	41.77
	SAM	95.19	94.08	48.36	92.08	36.87	87.33	21.54	54.56
	S^2SAM	95.23	93.71	72.51	93.44	37.06	89.72	17.37	81.40
ResNet-50 SVHN	SGD	94.44	94.51	9.15	10.05	9.15	10.05	9.15	10.05
	SFW	95.79	95.80	18.20	95.49	6.69	9.69	6.69	9.69
	CrAM	95.97	95.79	27.78	31.26	11.34	12.86	6.87	8.47
	SAM	96.96	95.98	9.74	95.99	9.69	95.90	9.75	95.59
	S^2SAM	96.88	95.78	17.28	95.78	10.93	95.74	9.56	95.62

Across high-sparsity pruning settings, the proposed WCR consistently improves accuracy under very high pruning, even when baseline performance collapses. The gains are most prominent when coupled with sharpness-aware optimizers such as SAM and S^2SAM, where WCR preserves most of the dense-model performance despite removing up to 98% of the weights.

E.2. Impact of Regularization Coefficient λ for Vision tasks

Tables 9 and 10 study the effect of the regularization coefficient λ on post-pruning accuracy and the resulting weight statistics. Consistent with our analysis, increasing λ enlarges the overall weight variance $\mathrm{Var}(w)$, reflecting that WCR concentrates weight energy onto a smaller subset of parameters and thereby increases the spread between large and small weights.

For CNNs (ResNet-18 on CIFAR-10, Table 9), the optimal λ depends on the base optimizer. Under SGD, $\lambda = 1 \times 10^{-5}$ gives the best accuracy at moderate pruning rates (90–94%), while smaller $\lambda = 5 \times 10^{-6}$ becomes preferable at the most aggressive 96% rate. Under SAM, the implicit flatness bias already suppresses $\mathrm{Var}(w)$ by roughly an order of magnitude (cf. the NA columns), and WCR remains effective across a wide range of $\lambda \in [10^{-6}, 10^{-5}]$, recovering accuracy from 48.36% to over 92% at 96% pruning.

For ViTs (ViT-B/32 on CIFAR-100, Table 10), a substantially larger $\lambda = 1 \times 10^{-2}$ is required to obtain meaningful pruning resilience across Q-, QK-, and QKV-pruning configurations. The gap between optimal λ values for CNNs and ViTs reflects the different scales of attention weights relative to convolutional kernels; overly large λ (e.g., 1×10^{-1}) eventually hurts accuracy, particularly under aggressive QK- and QKV-pruning where it interferes with the attention structure itself.

Table 9. Performance comparison across different regularization coefficient (λ) values on CIFAR-10 with ResNet-18. The column labeled *NA* denotes the baseline without the variance regularizer.

SGD									
Pruning	λ	NA	1e-6	5e-6	1e-5	5e-5	1e-4	5e-4	1e-3
Rate	$Var(w)$	0.0019	0.0021	0.0030	0.0035	0.0060	0.0081	0.0165	0.0226
Dense		92.75	93.14	91.87	92.63	92.79	91.62	90.54	90.66
90%		43.57	68.55	77.00	90.88	76.73	64.87	66.22	58.22
92%		31.60	57.53	65.47	80.24	68.05	53.47	52.11	47.39
94%		26.10	41.02	51.0	59.91	49.56	41.62	31.5	35.25
96%		19.69	24.53	36.79	25.70	20.60	21.72	12.75	20.18
SAM									
Pruning	λ	NA	1e-6	5e-6	1e-5	5e-5	1e-4	5e-4	1e-3
Rate	$Var(w)$	0.00005	0.00010	0.00018	0.00024	0.00050	0.00069	0.00151	0.00212
Dense		95.19	94.65	94.11	94.08	92.81	91.93	90.75	90.29
90%		91.26	94.47	94.22	94.04	92.77	91.82	90.75	90.13
92%		87.20	94.15	94.04	93.91	92.77	91.82	90.71	90.10
94%		76.29	92.61	93.69	93.77	92.69	91.74	90.69	90.08
96%		48.36	84.54	91.91	92.08	92.50	91.61	89.75	87.72

Table 10. Performance comparison of ImageNet-Pretrained Vision Transformer (ViT-B/32) models on CIFAR-100 under different pruning configurations and regularization strengths λ. Each block shows results for Q, QK, and QKV-pruning, where pruning is applied respectively to the Query, Query–Key, and Query–Key–Value within the self-attention mechanism.

Pruning Rate	λ $Var(w)$	**N/A** 0.0070	**1e-5** 0.0070	**1e-4** 0.0073	**1e-3** 0.0101	**1e-2** 0.0209	**1e-1** 0.0574
Q-Pruning							
Dense		89.84	90.00	90.15	89.62	88.75	85.23
60%		89.52	89.75	89.70	89.43	88.65	85.23
70%		88.88	89.15	89.19	89.04	88.58	85.16
80%		87.57	87.63	87.95	88.15	87.90	84.69
90%		80.29	81.27	81.51	83.84	85.13	82.75
92%		76.25	77.44	77.68	81.22	83.04	81.22
94%		68.97	70.52	70.66	76.48	78.88	78.54
96%		56.31	57.86	58.25	66.83	70.33	71.73
QK-Pruning							
Dense		89.84	90.00	90.15	89.62	88.75	85.23
60%		88.75	89.25	89.11	89.29	88.79	85.23
70%		87.09	87.23	87.47	88.29	88.11	85.15
80%		79.52	80.72	81.65	84.90	86.06	84.16
90%		51.54	52.61	54.57	65.31	73.41	75.01
92%		41.56	43.22	43.84	54.82	64.43	50.61
94%		31.67	33.52	33.45	41.31	50.65	33.18
96%		23.32	24.75	24.06	27.64	33.25	19.52
QKV-Pruning							
Dense		89.84	90.00	90.15	89.62	88.75	85.23
60%		86.30	86.68	87.29	88.23	88.58	85.23
70%		78.33	78.76	80.45	84.83	85.66	83.81
80%		51.13	52.93	55.48	69.63	73.69	68.11
90%		9.55	12.18	11.14	15.00	14.63	5.07
92%		7.23	7.83	7.75	9.21	8.82	2.84
94%		5.07	6.02	5.27	6.29	3.32	1.78
96%		3.04	4.42	3.62	4.61	2.31	1.51

E.3. Ablation study for SAM with regularizers

To further analyze the interaction between pruning-robust optimization and weight-shaping regularization, we evaluate SAM combined with DeepHoyer, ℓ_1, and WCR under different regularization coefficients λ. Table 11 reports the validation accuracy of ResNet-18 on CIFAR-10 under one-shot magnitude pruning. The results show that the effectiveness of each regularizer depends on the choice of λ and pruning ratio. In particular, WCR consistently maintains strong post-pruning accuracy across pruning levels and remains effective even at high sparsity.

Table 11. Validation accuracy (%) of ResNet-18 on CIFAR-10 under one-shot magnitude pruning for SAM combined with DeepHoyer, ℓ_1, and WCR, evaluated with different regularization coefficients λ.

Pruning Rate	SAM	**SAM + DeepHoyer**			**SAM + ℓ_1**			**SAM + WCR (Ours)**		
λ	NA	1e-4	1e-5	1e-6	1e-4	1e-5	1e-6	1e-4	1e-5	1e-6
Dense	92.75	65.13	90.31	92.04	89.64	90.18	94.12	91.93	94.08	94.65
90%	43.57	63.90	89.14	92.02	89.63	90.16	94.09	91.82	94.04	94.47
92%	31.60	63.54	89.03	91.43	88.16	90.08	93.75	91.82	93.91	94.15
94%	26.10	62.15	88.91	90.12	88.03	89.64	91.68	91.74	93.77	92.61
96%	19.69	62.07	88.89	90.05	73.59	80.56	82.79	91.61	92.08	84.54

E.4. Impact of Regularization Coefficient λ for LLM Fine-Tuning

Table 12 reports the effect of the regularization coefficient λ on Qwen2.5-1.5B under LoRA fine-tuning followed by one-shot magnitude pruning. We compare ℓ_1, DeepHoyer, and WCR across $\lambda \in \{10^{-7}, 10^{-6}, 10^{-5}\}$ under dense, 20%, and 30% pruning.

WCR with $\lambda = 10^{-6}$ achieves the highest average accuracy at both 20% and 30% pruning, reaching 75.31% and 69.54%, respectively. At dense, WCR with $\lambda = 10^{-7}$ attains the highest average accuracy in the table (77.60%). Across the three λ values, WCR's 30%-pruned average ranges from 64.76% to 69.54%, ℓ_1 from 24.49% to 67.48%, and DeepHoyer from 57.61% to 66.50%. The unregularized baseline reaches 72.63%, 71.62%, and 66.51% at dense, 20%, and 30% pruning, respectively.

For ℓ_1, the 30%-pruned average decreases from 67.48% at $\lambda = 10^{-7}$ to 49.81% at $\lambda = 10^{-6}$ and 24.49% at $\lambda = 10^{-5}$. For DeepHoyer, the dense average is highest at $\lambda = 10^{-6}$ (75.93%); at $\lambda = 10^{-5}$, the 20%-pruned average is 64.45% and the 30%-pruned average is 65.45%.

Table 12. Per-task test accuracy (%) of Qwen2.5-1.5B with LoRA (r=32) under one-shot magnitude pruning at dense, 20%, and 30% pruning ratios, across different regularization methods and coefficients λ.

Reg	λ	Pruning Rate	BoolQ	PIQA	SIQA	ARC-C	ARC-E	OBQA	HellaS	WinoG	Avg
none	N/A	dense	43.91	81.99	72.47	69.45	83.33	81.00	80.04	68.82	72.63
		20%	62.23	56.96	64.33	69.45	84.09	79.80	82.87	73.24	71.62
		30%	62.17	46.68	62.23	63.91	80.39	74.20	78.18	64.33	66.51
l1	1×10^{-7}	dense	65.05	80.90	74.05	65.44	79.88	75.60	75.80	73.40	73.77
		20%	64.43	78.78	73.23	63.48	78.16	76.40	73.03	71.19	72.34
		30%	63.09	75.19	67.81	57.68	71.42	70.00	67.31	67.32	67.48
	1×10^{-6}	dense	61.99	67.36	63.92	42.24	58.04	55.80	40.14	62.67	56.52
		20%	62.08	65.45	62.28	39.68	56.40	56.40	37.39	62.12	55.22
		30%	61.22	60.88	55.89	35.41	47.35	48.20	32.22	57.30	49.81
	1×10^{-5}	dense	62.26	59.47	57.16	33.53	42.80	45.20	27.88	57.62	48.24
		20%	62.17	53.16	56.96	32.08	40.66	45.00	25.83	57.38	46.65
		30%	57.43	35.80	13.00	9.04	10.40	14.20	9.38	46.65	24.49
DeepHoyer	1×10^{-7}	dense	46.42	81.50	73.90	70.22	86.87	82.00	57.70	72.77	71.42
		20%	62.20	71.93	68.27	63.31	81.73	74.00	52.06	57.62	66.39
		30%	63.91	75.52	30.45	69.20	82.74	71.60	0.85	66.61	57.61
	1×10^{-6}	dense	39.08	83.73	75.54	74.49	89.02	83.60	87.30	74.66	75.93
		20%	37.83	81.34	75.33	72.87	87.04	83.20	85.44	72.06	74.39
		30%	60.76	59.19	67.09	62.46	81.19	75.40	61.06	64.88	66.50
	1×10^{-5}	dense	66.82	82.97	75.08	74.40	87.96	82.80	67.18	61.33	74.82
		20%	66.76	83.19	74.56	71.25	87.12	80.40	28.18	24.15	64.45
		30%	65.90	76.50	69.75	64.16	81.65	76.60	44.20	44.83	65.45
WCR	1×10^{-7}	dense	64.53	81.66	74.62	72.27	86.91	81.60	85.01	74.19	77.60
		20%	63.43	51.74	70.47	67.66	85.77	77.40	73.74	75.93	70.77
		30%	63.82	77.26	45.60	55.63	74.16	70.00	75.46	56.12	64.76
	1×10^{-6}	dense	65.96	80.85	74.41	69.37	85.35	79.60	84.05	76.01	76.95
		20%	65.78	80.25	73.90	68.09	83.75	77.00	81.22	72.45	75.31
		30%	65.66	71.06	71.44	60.24	76.89	70.40	70.85	69.77	69.54
	1×10^{-5}	dense	65.12	79.84	73.62	67.81	83.92	78.20	82.01	74.38	75.61
		20%	64.89	78.73	72.68	66.35	81.94	75.60	78.94	71.02	73.77
		30%	64.71	69.32	69.87	58.47	74.83	68.80	68.24	68.15	67.80

E.5. Additional segmentation results

To further examine the effect of the proposed Weight Concentration Regularizer (WCR) under high pruning rates, we provide additional qualitative segmentation results in Figure 6 and Figure 7. Each figure compares the segmentation masks produced by several pruning-robust training methods, both with and without WCR, alongside the corresponding ground-truth tumor annotations.

Across all examples, models trained without WCR often show degraded predictions after 85% one-shot pruning. Typical failure patterns include incomplete tumor regions, loss of boundary continuity, and irregular or noisy shapes. In contrast, models trained with WCR consistently preserve more complete lesion structures and maintain clearer and more accurate tumor boundaries. Methods such as SFW, SAM, and S^2SAM particularly benefit from the addition of WCR, which stabilizes the model's internal representations even under severe sparsity.

These qualitative findings are consistent with the quantitative results presented in the main paper. By promoting broader and more structured weight distributions during training, WCR reduces the distortion caused by aggressive pruning and leads to better segmentation masks, especially for tumors with complex shapes or low-contrast regions.

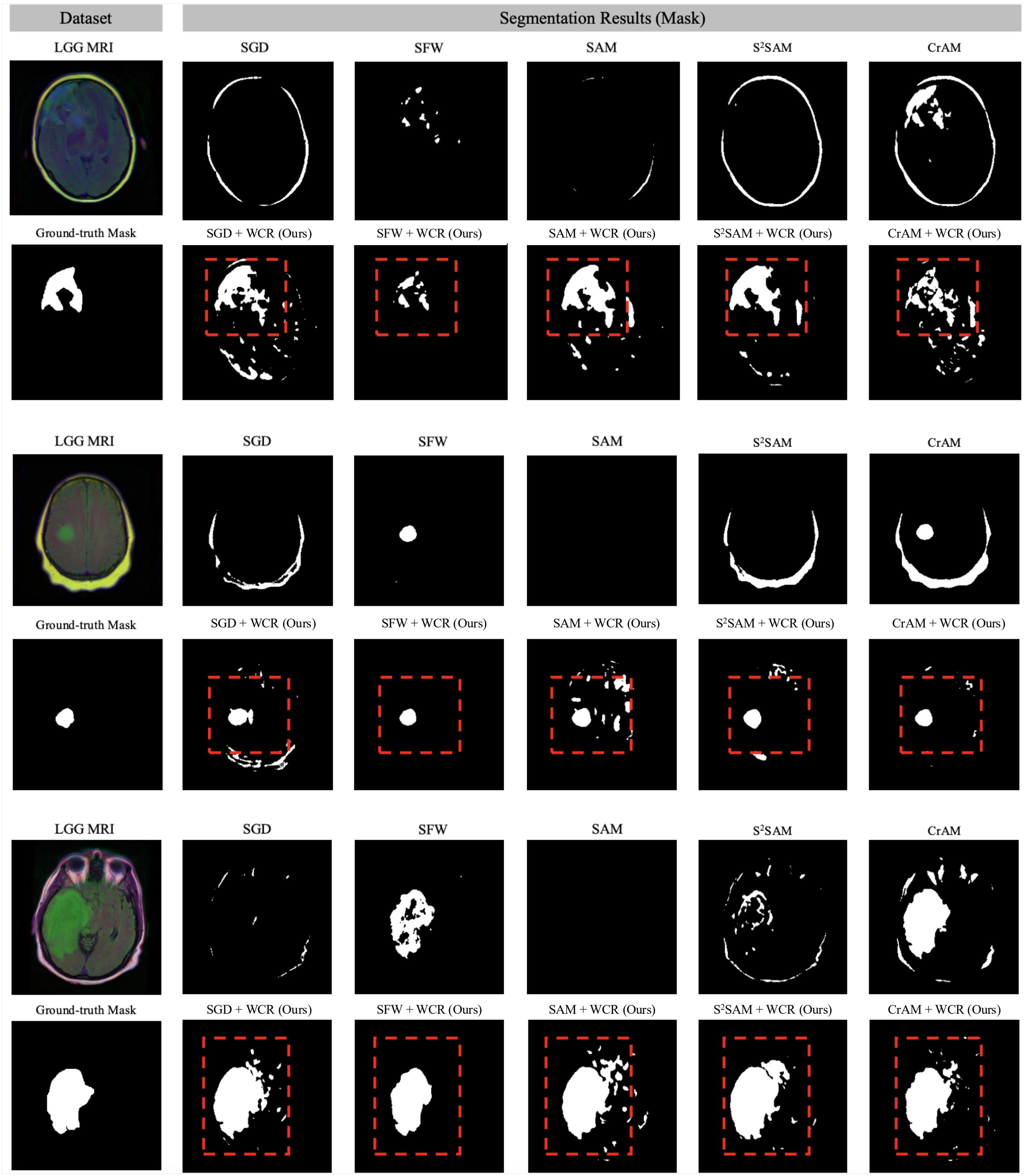

Figure 6. Qualitative segmentation results on the LGG MRI dataset using the ResNet-50–UNet architecture under 85% pruning. Each column displays the outputs of different pruning-robust training methods, shown both with and without our proposed Weight Concentration Regularizer (WCR), along with the corresponding ground-truth mask.

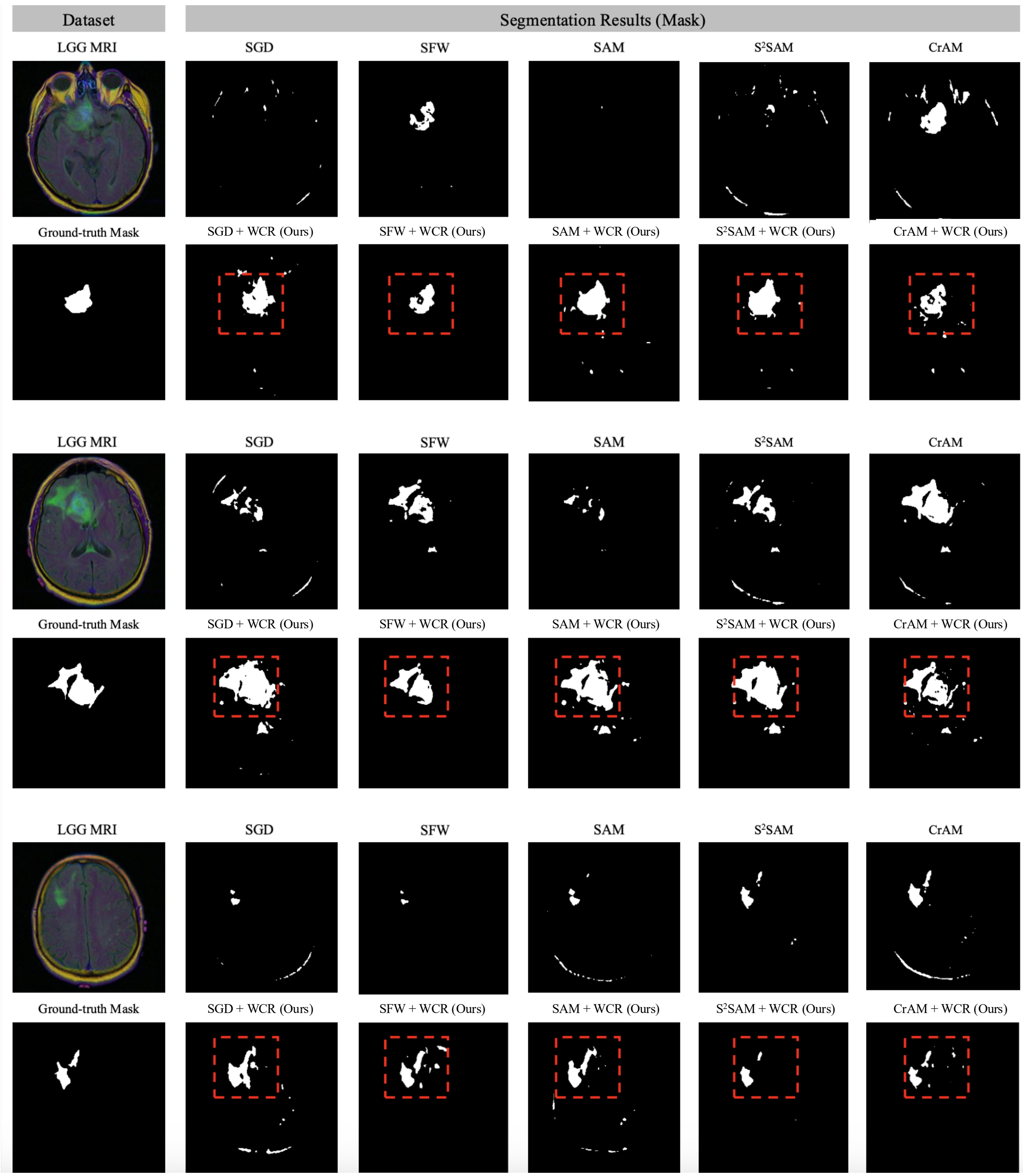

Figure 7. Qualitative segmentation results on the LGG MRI dataset using the ResNet-50–UNet architecture under 85% pruning. Each column displays the outputs of different pruning-robust training methods, shown both with and without our proposed Weight Concentration Regularizer (WCR), along with the corresponding ground-truth mask.