Minimal-State Optimization: How Little Optimizer Memory Does AdamW's Accuracy Need?

A Sharpness-Guided, Learning-Rate-Free Study

Anonymous authors
Paper under double-blind review

Abstract

Adaptive optimizers such as AdamW store two extra tensors per parameter (first and second moments), doubling the model's memory footprint. How much of this optimizer state is actually necessary to match AdamW's accuracy? We study this through A*Grad, a learning-rate-free optimizer whose step size is set by a Polyak rule with a trust region, and whose only tunable hyperparameter is a dimensionless ratio ρ. By varying only the geometry of the update direction, A*Grad spans a spectrum of optimizer-state sizes. We find the required state is task-dependent, and a sharpness diagnosis predicts it: the sign-based step (zero state) discards gradient magnitude and converges to minima 5–10× sharper than SGD's (filter-normalized perturbation), which hurts generalization. Restoring magnitude resolves this issue, though the optimal restoration mechanism is strictly task-dependent. For from-scratch vision (CIFAR-100/WRN), a global L_2-normalized step with weight decay suffices at zero optimizer state, outperforming AdamW by $+2.4\%p$. For Transformer fine-tuning (BERT/MNLI), per-coordinate RMS is needed; at half AdamW's state, it outperforms it by $+0.5\%p$ with the smallest seed variance among all methods. We place A*Grad against two recent low-memory optimizers, MicroAdam and NanoAdam, on an accuracy–memory Pareto front, and delineate the operational regimes where A*Grad demonstrates clear advantages (stability, zero-state vision, hyperparameter-free operation) versus scenarios where its benefits are attenuated (certain fine-tuning tasks, peak accuracy at scale). The code of the proposed optimizer is publicly available at Github (https://anonymous.4open.science/r/astar_grad-D001).

1 Introduction

Modern training pipelines spend a large fraction of device memory not on the model or its activations but on optimizer state. AdamW keeps a first moment m and second moment v for every parameter; at fp32 that is 8 bytes per parameter, versus 4 for the weights. For a 110M-parameter BERT-base this is 835 MB of optimizer state alone. A natural question—motivating a recent wave of memory-efficient optimizers (Dettmers et al., 2022; Zhao et al., 2024; Modoranu et al.; zhou et al.)—is: how little of this state do we need to retain AdamW-level accuracy?

We approach this not by compressing AdamW's state but by asking what the minimal state a well-designed step rule requires fundamentally. We investigate this via A*Grad, a learning-rate-free optimizer. Instead of a tuned learning-rate schedule, the step size follows a Polyak rule $\gamma = (L - L^*)/(\bar{G} + \epsilon)$, and a trust region caps each step to a fraction ρ of the current weight norm. There is no learning rate, no schedule, no warmup, and no first moment; the single dimensionless hyperparameter ρ replaces the 3–4 order-of-magnitude learning-rate search with a ∼1.5-decade one.

Within this frame, the only design choice that changes optimizer state is the geometry of the update direction: $\mathrm{sign}(g)$ and $g/\|g\|_2$ add nothing (state 0); $g/\sqrt{\hat{v}}$ adds one tensor v (per-coordinate, like Adam's denominator but without m). This lets us ask the minimal-state question cleanly, and answer it with a diagnosis → prescription → verification chain:

1. Diagnosis. The sign step discards gradient magnitude and converges to sharp minima—5–10× sharper than SGD's under filter-normalized perturbation (Li et al.). Sharpness, not optimization failure, fundamentally limits its generalization.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

2. Prescription (task-dependent). Restoring gradient magnitude resolves this limitation, yet the minimal sufficient state varies across domains: vision needs only global L_2 + weight decay (state 0); Transformer fine-tuning needs per-coordinate RMS (state $\frac{1}{2}$).
3. Verification. Each prescription outperforms AdamW at its task using strictly less state, across 3 seeds; conversely, cross-applying prescriptions yields substantially degraded performance or divergence.

Contributions. (i) A*Grad, a learning-rate-free optimizer (Polyak step + dimensionless trust region) with a non-divergence guarantee (§6): the trust region bounds each step to $\rho\|w\|$, cancelling the $\sqrt{d}$ blow-up of sign steps regardless of dimension. (ii) A sharpness diagnosis that both explains sign's failure (measured 5–10×) and predicts the task-dependent minimal state. (iii) Minimal-state results: vision at state 0 (AdamW $+2.4\%p$), Transformer at state $\frac{1}{2}$ (AdamW $+0.5\%p$), each 3-seed, holding from BERT-base to BERT-large. (iv) An accuracy–memory Pareto comparison with MicroAdam and NanoAdam, including a reproduction check that corrected our own NanoAdam baseline by 10× density. (v) Two reusable methodological lessons: seed expansion identifying latent numerical instabilities, and rigorous baseline verification preventing flawed comparative assessments.

We are explicit about scope: all results are at the BERT-base/large and CIFAR scale on Quadro RTX 8000 (Turing, no native bf16); we cannot run 7B+ models or native bf16, which limits the MicroAdam comparison (§8). Our claim is not "A*Grad dominates"—at scale and on some tasks the accuracy gap narrows or reverses—but that a sharpness-guided, learning-rate-free method reaches AdamW's accuracy at a fraction of its state, with the best stability among the methods compared.

2 Method: A*Grad

With parameters $w \in \mathbb{R}^d$, loss L, gradient $g_t = \nabla L(w_t)$, and a direction d_t (one of $\mathrm{sign}(g)$, $g/\|g\|_2$, $g/\sqrt{\hat{v}+\epsilon}$):

$$\bar{G}_t = \alpha\,\bar{G}_{t-1} + (1-\alpha)\,\langle g_t, d_t\rangle \qquad \text{(Polyak denominator; scalar)} \tag{1}$$

$$\gamma_t^P = (L_t - L^*)/(\bar{G}_t + \epsilon) \qquad \text{(Polyak step; } L^*\text{=0 default)} \tag{2}$$

$$\gamma_t = \min\big(\gamma_t^P,\ \rho\,\|w_t\|_2/\|d_t\|_2\big) \qquad \text{(trust region)} \tag{3}$$

$$w_{t+1} = (1-\gamma_t\lambda)\,w_t - \gamma_t\,d_t \qquad \text{(decoupled weight decay } \lambda) \tag{4}$$

Theoretical Foundation of the Polyak Step. For $d = \mathrm{sign}(g)$, $\langle g, d\rangle = \|g\|_1$, and $\gamma = (L - L^*)/\|g\|_1$ is the exact steepest-descent step in the L_∞ geometry whose dual norm is L_1. The same derivation gives $\|g\|_2$ for L_2 and $\sum g^2/\sqrt{\hat{v}}$ for RMS. No learning rate appears.

Dimension-Free Stability via Trust Region. A sign step moves all d coordinates by γ at once, so its L_2 displacement is $\gamma\sqrt{d}$. This $\sqrt{d}$ amplification breaks the linear approximation of the Polyak step and diverges on large models: the unconstrained formulation ($\rho = \infty$) reaches loss $\approx 4 \times 10^{29}$ within 3 epochs on CIFAR. The cap $\rho\|w\|_2/\|d\|_2$ makes the displacement $\le \rho\|w\|_2$ regardless of direction or dimension (Lemma 1). ρ is dimensionless, so the cap tracks model size automatically; the same $\rho = 0.0015$ works from BERT-base to BERT-large.

Adaptive warmup, not a constant cap. Empirically, for $\rho \ge 0.5\%$ the Polyak step falls below the cap after the first few steps; the cap is active only early, and deactivates as L_t (hence γ^P) shrinks. Only $\rho = 0.1\%$ remained perpetually bounded by the trust region, degenerating into fixed-step signSGD and failing to converge. This absorbs the warmup schedule into the algorithm.

State accounting. $\bar{G}_t$ is a single scalar (0 MB). Sign/L_2 add nothing; RMS adds one tensor v (state $1{\cdot}d$ = half of AdamW's $2{\cdot}d$).

RMS needs bias correction. With $v = \mathrm{EMA}(g^2)$ at $\beta = 0.999$, early v is underestimated and $d = g/\sqrt{v}$ explodes. Proper $\hat{v} = v/(1-\beta^t)$ is necessary: without it, one of three BERT seeds diverged (32.7 = chance); with it, all three are stable (74.6/74.9/74.8, ±0.18; §6.2).

3 Diagnosis: Sign Steps Discard Magnitude and Converge to Sharp Minima

On CIFAR-100 (WRN-16-8 (Zagoruyko & Komodakis, 2016), 200 epochs) the sign step trains well but generalizes poorly—training accuracy is lower than SGD's (98.2 vs 99.5) yet its minimum is far sharper. We measure this with filter-normalized random perturbations (Li et al.): perturb the trained weights by ϵ in a filter-normalized random direction and record the loss increase (60-epoch checkpoints).

ϵ	A*Grad(sign) Δloss	SGD Δloss	ratio
0.02	0.028	0.006	4.9×
0.10	2.38	0.29	8.3×

The sign-trained minimum is 5–10× sharper. The mechanism: sign(g) throws away gradient magnitude, so it cannot exploit flat-direction information to settle into wide basins. This diagnosis is prescriptive: restoring gradient magnitude directly addresses sharpness. The central question then becomes identifying the minimal sufficient state, which proves to be task-dependent (§4).

4 Task-adaptive prescription: the minimal state that fixes sharpness

Table 1: Main results. Each prescription outperforms AdamW at its task using strictly less state; the other task's prescription is worse (vision RMS) or diverges (Transformer L_2). 3 seeds, mean±std.

Task	Optimizer	Accuracy	Opt. state	vs AdamW
CIFAR-100 (WRN, 200ep)	A*Grad (L_2+wd)	**74.04 ± 0.02**	0 MB	**+2.39**
	SGD	74.55 ± 0.13	0 MB	+2.90
	AdamW	71.65 ± 0.05	84 MB	—
	A*Grad (rms)	71.29 ± 0.56	44 MB ($\frac{1}{2}$)	−0.36 (worse)
BERT-MNLI (393k, 3ep)	A*Grad (rms)	**83.63 ± 0.17**	417 MB ($\frac{1}{2}$)	**+0.52**
	AdamW	83.11	835 MB	—
	NanoAdam	83.05 ± 0.13	167 MB (20%)	−0.06
	A*Grad (L_2)	diverges	0 MB	— (diverges)

Vision (from-scratch): state 0. Restoring magnitude globally ($g/\|g\|_2$ + decoupled weight decay) suffices for vision at zero optimizer state (Table 1). A*Grad outperforms AdamW by +2.39%p at 0 MB and is within 0.5%p of a tuned SGD—without a learning rate. Ablation (seed 42): sign 68.4 < L_2 71.8 < L_2+wd 74.1, exactly as the sharpness diagnosis predicts.

Transformer (fine-tuning): state $\frac{1}{2}$. Global normalization is not enough: per-layer gradient scales vary too much, and a per-tensor (blockwise) variant fails to match the performance of AdamW. Per-coordinate RMS is needed—one tensor v, half of AdamW's state—surpassing AdamW by +0.52%p with the smallest seed variance (Table 1).

The prescription is genuinely task-dependent (both directions). The vision prescription (L_2, state 0) diverges on BERT (chance, at $\rho \in \{0.01, 0.001\}$). Conversely, per-coordinate RMS is worse on vision: 71.29 at half the state, no better than AdamW—allocating additional optimizer state paradoxically yields lower accuracy. The minimal sufficient state is set by the task's gradient geometry, and the sharpness diagnosis systematically identifies the requisite memory configuration. ρ itself is task-dependent (CIFAR optimum 0.02 vs BERT 0.0015).

Scale generalization: BERT-large (335M). Scaling from BERT-base to BERT-large demonstrates consistent empirical trends with one notable distinction (Table 2): A*Grad-rms remains best and by far the most stable (std 0.19 vs ∼ 0.6). The accuracy gap narrows at scale (+0.21%p), but the 3× reduction in seed variance represents the primary robustness benefit—a consequence of being learning-rate-free.

Table 2: BERT-large (335M), MNLI, 3 epochs, 3 seeds. A*Grad-rms keeps the best mean and by far the smallest variance. MicroAdam diverges (bf16 emulation on sm_75).

Optimizer	Accuracy	±std	State
A*Grad-rms	**85.83**	**0.19**	50%
AdamW	85.62	0.64	100%
NanoAdam	85.50	0.59	20%
MicroAdam	32.74	—	(diverged)

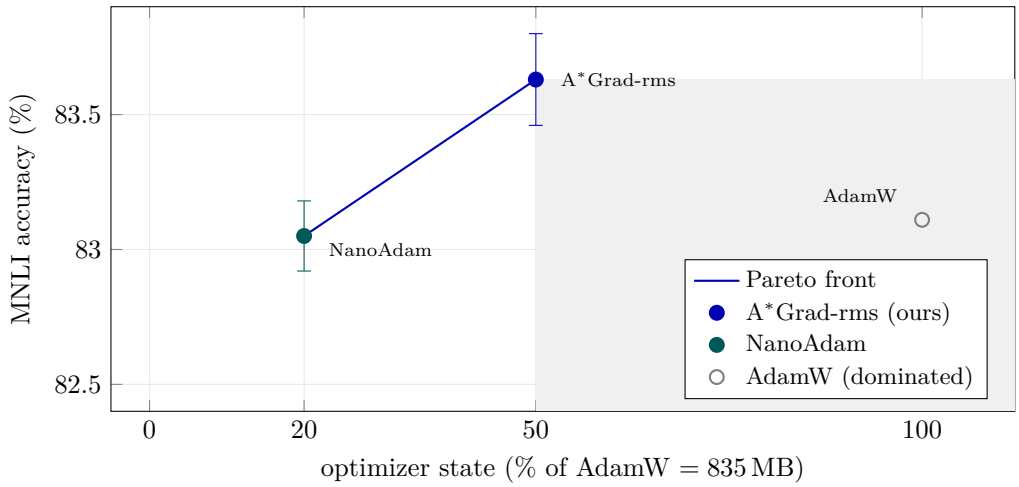


Figure 1: Accuracy–memory Pareto front, BERT-MNLI (3 epochs, 3 seeds). A*Grad-rms and NanoAdam do not dominate each other (rms +0.58%p accuracy, NanoAdam 2.5× less state); both are on the front. AdamW is dominated by A*Grad-rms (higher accuracy at half the state). The zero-state point (A*Grad-L_2) belongs to the front for vision only. MicroAdam is omitted from the plot due to divergence: 68.0 at a 20k subset, and it fails at full scale on our Turing GPU—31.8 at full BERT-base, 32.7 at BERT-large (bf16 emulation).

5 Low-memory optimizer landscape: a Pareto comparison

We compare against two recent methods with the same goal but different mechanisms: MicroAdam (Modoranu et al.) top-k sparsifies the gradient with quantized error feedback (official code; its kernel requires bf16 gradients); NanoAdam (zhou et al.) runs Adam on a weight subset with minimal magnitude (density k, gradient-free mask, classifier/LayerNorm excluded; no official code, we reimplement).

A reproduction check corrected our own baseline. Our first NanoAdam reimplementation used density 1%, giving 81.5 (well behind A*Grad-rms). A paper-value reproduction check (does it reach the reported 84.30?) failed, and the appendix showed the main-experiment density is 10% → 4% linear decay—a 10× error, plus a missed classifier/LayerNorm exclusion. After correction, NanoAdam reaches 83.05 ± 0.13 at 20% state (Table 1), and reproduces the paper's protocol (5 epochs, batch 8) at 83.28 vs the reported 84.30—sufficiently consistent to validate the empirical soundness. This correction altered our comparative assessment: NanoAdam is a substantially more competitive baseline than our suboptimal preliminary implementation suggested.

The Pareto front. Figure 1: A*Grad-rms and NanoAdam do not dominate each other, both on the front; AdamW is dominated by A*Grad-rms. A*Grad occupies the front's extremes—zero-state (vision) and top-accuracy (Transformer)—while NanoAdam holds the low-memory middle.

Task diversity: A*Grad does not dominate. Across four GLUE tasks (Table 3), A*Grad-rms outperforms baselines on two tasks while underperforming on QNLI by 0.7%p even after ρ tuning. We report these findings transparently: the claim is not per-task accuracy dominance but the collective advantages—competitiveness

+ stability + learning-rate-free + task-adaptivity (the zero-state vision point fine-tuning-only baselines cannot reach) + minimal state.

Table 3: Task diversity, BERT-base, 3 epochs, seed 42 (accuracy). A*Grad-rms wins 2/4 and loses QNLI even after hyperparameter optimization of ρ.

Task	A*Grad-rms	AdamW	NanoAdam	winner
MNLI	83.63	83.11	83.05	rms
MRPC	85.05	84.56	84.80	rms
SST-2	91.97	92.09	91.74	AdamW (+0.12)
QNLI	89.49*	90.19	90.24	NanoAdam (rms −0.7)

*best over a ρ retune; rms cannot reach 90 on QNLI even after tuning.

6 Robustness, theory, and methodology

6.1 Non-divergence guarantee

Lemma 1 (step bound). Under the trust region equation 3, for every step and every direction, $\|\gamma_t d_t\|_2 \leq \rho \|w_t\|_2$.

Proof. $\gamma_t \leq \rho\|w_t\|_2/\|d_t\|_2$ by equation 3, so $\gamma_t\|d_t\|_2 \leq \rho\|w_t\|_2$. □

This cancels the $\sqrt{d}$ amplification of sign steps, dimension-independently. Numerically verified: l1/l2/rms all attain $\max \|\Delta w\|/\|w\| = \rho$ exactly (0 violations), the equality holding early when the cap is active.

Corollary 1 (non-divergence). With update equation 4 and $\gamma_t\lambda \leq 1$, $\|w_t\|_2 \leq (1+\rho)^t\|w_0\|_2$ for all t.

Proof. $\|w_{t+1}\|_2 \leq (1-\gamma_t\lambda)\|w_t\|_2 + \gamma_t\|d_t\|_2 \leq (1+\rho)\|w_t\|_2$ by Lemma 1; induct. □

While this bound is conservative (reflecting worst-case accumulation), divergence—the raw $\rho = \infty$ behaviour (loss $\approx 4 \times 10^{29}$)—is strictly precluded for any $\rho < \infty$. Since the Polyak step falls below the cap after the first steps, the trust region acts as an adaptive warmup. Establishing formal non-convex convergence rates remains an important objective for future theoretical work (via signSGD/normalized-SGD theory (Bernstein et al., 2018)).

6.2 Methodological notes

Seed expansion caught a robustness bug. The RMS variant's first success (seed 42) hid a defect: seed 43 diverged (32.7) from missing bias correction. Only seed expansion revealed it; $\hat{v} = v/(1-\beta^t)$ fixed all three seeds. Evaluating on a single seed would have published a fragile algorithm.

Speed is not an advantage (negative result). Per step, A*Grad-rms is comparable to or slower than AdamW (the global reductions $\langle g, d\rangle, \|w\|, \|d\|$ cost time AdamW's per-parameter state avoids), and the memory saving does not raise throughput (activations dominate at large batch). Fusing the reductions changes fp32 rounding which, fed through the Polyak denominator $\bar{G}$, shifts the trajectory and degrades accuracy—a hazard specific to lr-free optimizers whose global reductions enter the step size. Consequently, we report no computational speed advantage.

7 Related work

Polyak step size. Classic Polyak subgradient steps (Polyak) set $\gamma = (f - f^*)/\|g\|^2$; stochastic variants revisit them (Loizou et al., 2021). A*Grad changes the direction geometry, smooths the denominator with an EMA, and adds the trust region for large-model stability. Sign/normalized methods. signSGD (Bernstein et al., 2018) and Lion (Chen et al., 2023) (sign + decoupled wd) are zero-first-moment sign methods but need a tuned learning rate; A*Grad is learning-rate-free and task-adaptive in direction. Low-memory optimizers. 8-bit Adam (Dettmers et al., 2022), GaLore (Zhao et al., 2024), MicroAdam (Modoranu et al.), NanoAdam

(zhou et al.) compress Adam's state; A*Grad instead asks what state a step rule needs, reaching zero (vision) or half (Transformer). Sharpness. Filter-normalized sharpness (Li et al.), SAM (Foret et al., 2021). We use sharpness diagnostically to predict minimal state, not as a training objective.

8 Limitations

- Scale. Results are at BERT-base/large ($\leq$ 335M) and CIFAR. We cannot run 7B+ models or native bf16 (Quadro RTX 8000), so the modern-LLM-scale regime is untested. The MicroAdam comparison is compromised by this: its bf16 kernel emulates on Turing GPU and fails at full scale (31.8 base, 32.7 large)—this reflects our hardware, not necessarily the method.
- Contextual Performance Trade-offs. A*Grad-rms slightly underperforms on SST-2 (-0.12) and QNLI (-0.7) among the four evaluated GLUE tasks, and the base$\rightarrow$large gap over AdamW narrows to $+0.2\%p$. The durable edge is stability ($3\times$ smaller std) and learning-rate-free operation, not raw peak accuracy.
- Parity with Tuned SGD in Vision. A*Grad matches rather than exceeds the accuracy of a thoroughly tuned SGD baseline; the central takeaway is surpassing AdamW with zero optimizer state.
- Theoretical Scope. We provide a rigorous non-divergence guarantee rather than non-convex convergence rates.
- ρ is task-dependent. Learning-rate-free, but one ρ per task family is still needed; QNLI additionally shows ρ sensitivity (divergence at $\rho \geq 0.003$).

9 Conclusion

This work investigated the minimal optimizer state required to preserve AdamW's accuracy through a sharpness-guided, learning-rate-free framework. The sign step's zero-state failure is a sharpness failure (5–10$\times$), and restoring gradient magnitude fixes it at a task-dependent minimal state: zero for from-scratch vision (AdamW $+2.4\%p$), half for Transformer fine-tuning (AdamW $+0.5\%p$), holding to BERT-large with the smallest seed variance among all methods. On the empirical accuracy–memory Pareto frontier, A*Grad occupies both boundary regimes (zero-state and high-accuracy) while NanoAdam spans the intermediate memory regime, and AdamW is dominated. We do not claim per-task accuracy dominance; we claim that a diagnosis-driven, learning-rate-free optimizer reaches AdamW's accuracy at a fraction of its state, stably, and that the diagnosis prescribes how much state each task needs.

Acknowledgments

(Anonymous for review.)

Reproducibility Statement

Reproducibility is a first-class concern of this work, and two of our findings are themselves about it (§6.2). Concretely: (i) the optimizer, all baselines including our NanoAdam reimplementation, and every experiment runner are released as code; (ii) Appendix A gives the complete hyperparameters—batch size, sequence length, epochs, per-optimizer learning rates and settings, augmentation, and weight decay—for both benchmarks, with Tables 4–5 listing each optimizer's exact configuration; (iii) all headline numbers are reported over seeds $\{42, 43, 44\}$ as mean$\pm$std, so variance (not just a lucky seed) is visible—indeed seed expansion is what revealed the RMS bias-correction bug; (iv) the exact hardware is stated (Quadro RTX 8000, Turing/sm_75) because it materially affects the MicroAdam comparison (bf16 emulation), and we say so rather than hide it; (v) our NanoAdam reimplementation is validated against the paper's reported MNLI number under the paper's own protocol (83.28 vs. 84.30; §5), so the comparison does not rest on a weakened baseline; (vi) the non-divergence lemma (Lemma 1) is checked numerically in addition to being proved; (vii) datasets are the standard public CIFAR-100 and GLUE/MNLI loaders with no bespoke preprocessing. Reproduction commands are included with the released code.

Ethics Statement

This work studies a general-purpose optimization algorithm and does not involve human subjects, personal or sensitive data, or generative models with obvious misuse potential; the datasets used (CIFAR-100 and GLUE/MNLI) are standard public benchmarks. The most direct societal dimension is positive: by reaching AdamW-level accuracy at a fraction of the optimizer state, the method lowers the memory barrier to fine-tuning, which can broaden access to training on modest hardware and reduce the memory (and hence energy) overhead of optimization. As a general optimizer, it carries the same dual-use profile as any training method—it could in principle be applied to harmful models—but introduces no capability specific to such use. We have taken care to state limitations and negative results explicitly (§8, §6.2: no speed advantage, per-task losses, hardware-bound MicroAdam failure, a corrected baseline) so the claims are neither over- nor under-stated. We declare no competing interests.

Use of Generative AI

AI tools (Claude) were used solely to proofread and polish the English text to improve readability. The authors conceived and executed the research and take full responsibility for all content in this manuscript.

References

Jeremy Bernstein, Yu-Xiang Wang, Kamyar Azizzadenesheli, and Animashree Anandkumar. signSGD: Compressed optimisation for non-convex problems. In Proceedings of the 35th International Conference on Machine Learning, volume 80 of Proceedings of Machine Learning Research, pp. 560–569. PMLR, 10–15 Jul 2018.

Xiangning Chen, Chen Liang, Da Huang, Esteban Real, Kaiyuan Wang, Hieu Pham, Xuanyi Dong, Thang Luong, Cho-Jui Hsieh, Yifeng Lu, and Quoc V Le. Symbolic discovery of optimization algorithms. In Advances in Neural Information Processing Systems, volume 36, pp. 49205–49233. Curran Associates, Inc., 2023. doi: 10.52202/075280-2140.

Tim Dettmers, Mike Lewis, Sam Shleifer, and Luke Zettlemoyer. 8-bit optimizers via block-wise quantization. In International Conference on Learning Representations, 2022.

Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. In International Conference on Learning Representations, 2021.

Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. In Advances in Neural Information Processing Systems.

Nicolas Loizou, Sharan Vaswani, Issam Hadj Laradji, and Simon Lacoste-Julien. Stochastic polyak step-size for sgd: An adaptive learning rate for fast convergence. In Proceedings of The 24th International Conference on Artificial Intelligence and Statistics, volume 130 of Proceedings of Machine Learning Research, pp. 1306–1314. PMLR, 13–15 Apr 2021.

Ionut-Vlad Modoranu, Mher Safaryan, Grigory Malinovsky, Eldar Kurtic, Thomas Robert, Peter Richtárik, and Dan Alistarh. Microadam: Accurate adaptive optimization with low space overhead and provable convergence. pp. 1–43. doi: 10.52202/079017-0001.

B.T. Polyak. Introduction to Optimization. ISBN 9780911575149.

Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. pp. 87.1–87.12, 2016. doi: 10.5244/C.30.87.

Jiawei Zhao, Zhenyu Zhang, Beidi Chen, Zhangyang Wang, Anima Anandkumar, and Yuandong Tian. GaLore: Memory-efficient LLM training by gradient low-rank projection. In Proceedings of the 41st International Conference on Machine Learning, volume 235 of Proceedings of Machine Learning Research, pp. 61121–61143. PMLR, 21–27 Jul 2024.

Chao zhou, Tom Jacobs, Advait Gadhikar, and Rebekka Burkholz. Pay attention to small weights. In Advances in Neural Information Processing Systems, pp. 146132–146167. Curran Associates, Inc. doi: 10.52202/085713-4890.

A Experimental details

Common setup. All runs use hardware of 4× Quadro RTX 8000 (48 GB, Turing, compute capability 7.5; no native bf16). Each configuration is repeated over seeds $\{42, 43, 44\}$ unless noted (single seed for the task-diversity scan, Table 3). Reported numbers are the best test accuracy over training (CIFAR) or the final validation_matched accuracy on the full 9,815-example split (MNLI). Datasets: CIFAR-100 and GLUE (nyu-mll/glue) are downloaded via their standard loaders; no extra preprocessing beyond what is listed below. Precision is fp32 throughout, except MicroAdam whose kernel forces bf16 (emulated on sm_75). The A*Grad EMA decay is $\alpha = 0.9$ and $\epsilon = 10^{-8}$ in all runs; A*Grad has no learning rate. Decoupled weight decay (λ) follows the Lion form and is applied only where stated.

Table 4: BERT-MNLI fine-tuning. Batch size 32, max sequence length 128, weight decay 0, no warmup, no LR schedule. Epochs: 3 for the full 393k-example results (Tables 1, 2, 3), 1 for the 20k-subset LR sweeps, 5 only for the NanoAdam paper-protocol reproduction (batch 8). AdamW/NanoAdam/MicroAdam use $\beta = (0.9, 0.999)$.

Optimizer	Learning rate	Key hyperparameters
A*Grad-rms	— (lr-free)	$\rho = 0.0015$
A*Grad-L_2	— (lr-free)	$\rho \in \{0.01, 0.001\}$ (both diverge on BERT)
AdamW (base)	2×10^{-5}	—
AdamW (large)	1×10^{-5}	—
NanoAdam	7×10^{-5}	density $0.1 \to 0.04$ (linear), mask interval 383
MicroAdam	5×10^{-5}	$m = 10$, quant. block 10^5, $k_{\text{init}} = 0.01$, bf16
SGD	1×10^{-2}	momentum 0

Table 5: CIFAR-100 from-scratch. Model WRN-16-8 (no dropout), batch size 128, 200 epochs. Augmentation: RandomCrop(32, padding 4) + RandomHorizontalFlip + per-channel normalization. Weight decay is 0 except the A*Grad-L_2+wd and A*Grad-rms rows ($\lambda = 5 \times 10^{-4}$).

Optimizer	Learning rate	Key hyperparameters
A*Grad-L_2+wd	— (lr-free)	$\rho = 0.01$, $\lambda = 5 \times 10^{-4}$
A*Grad-rms	— (lr-free)	$\rho = 0.01$, $\lambda = 5 \times 10^{-4}$
SGD	0.1	momentum 0
AdamW	1×10^{-3}	$\beta = (0.9, 0.999)$, wd 0

Per-optimizer tuning. A*Grad is learning-rate-free; its only hyperparameter ρ was tuned once per task family (CIFAR $\rho = 0.02$ optimum for sign, 0.01 for L_2/rms; BERT $\rho = 0.0015$), consistent with the task-dependence noted in §4. Baseline learning rates were selected by sweeps: AdamW on BERT over $\{1, 2\} \times 10^{-5}$ (base/large), NanoAdam over $\{9 \times 10^{-5}, \ldots, 3 \times 10^{-3}\}$ (full-training optimum 3×10^{-4}; the 20k-subset optimum is a larger 3×10^{-3}, since short runs favour larger steps), MicroAdam over $\{1 \times 10^{-5}, \ldots, 5 \times 10^{-4}\}$. Numbers use each method's best setting.