DoG is SGD's Best Friend: A Parameter-Free Dynamic Step Size Schedule

Maor Ivgi maor.ivgi@cs.tau.ac.il

Oliver Hinder ohinder@pitt.edu

Yair Carmon ycarmon@tauex.tau.ac.il

Abstract

We propose a tuning-free dynamic SGD step size formula, which we call Distance over Gradients (DoG). The DoG step sizes depend on simple empirical quantities (distance from the initial point and norms of gradients) and have no "learning rate" parameter. Theoretically, we show that a slight variation of the DoG formula enjoys strong parameter-free convergence guarantees for stochastic convex optimization assuming only *locally bounded* stochastic gradients. Empirically, we consider a broad range of vision and language transfer learning tasks, and show that DoG's performance is close to that of SGD with tuned learning rate. We also propose a per-layer variant of DoG that generally outperforms tuned SGD, approaching the performance of tuned Adam.

1 Introduction

While stochastic optimization methods drive continual improvements in machine learning, choosing the optimization parameters—and particularly the learning rate—remains a difficulty. Standard methodologies include searching over a set of learning rates, or simply picking the learning rate from prior work. The former incurs a substantial computational overhead, while the latter risks training a suboptimal model.

The rich literature on adaptive gradient methods (AdaGrad, Adam, and their many variants) offers optimization algorithms that better exploit problem structure [e.g., 27, 45, 32, 79, 55]. However, these methods still have a learning rate parameter that requires tuning. The theoretically-optimal value of this parameter depends on unknown problem properties. For example, on convex problems the optimal learning rate of AdaGrad is related to the distance between the initial point and the optimal solution, while in non-convex settings it is related to the function's smoothness and initial optimality gap [31, 89, 28].

Parameter-free optimization aims to remove the need for such tuning by designing algorithms that achieve a near-optimal rate of convergence with almost no knowledge of the problem properties [82]. Most works in this field [e.g., 64, 19, 40, 58, 9] use advanced online learning techniques to construct algorithms that, for the fundamental setting of stochastic convex optimization (SCO) with bounded stochastic gradients, achieve optimal rates of convergence up to logarithmic factors. However, parameter-free algorithms have yet to make it into common use, perhaps due to the fact that they are quite different from the classical stochastic gradient descent (SGD). Recently, Carmon and Hinder [10] have shown that performing a careful bisection over the SGD step size yields a parameter-free optimization method that is optimal for SCO up to a double-logarithmic factor. While theoretically novel, on a practical level the result leaves much to be desired, as it essentially prescribes the standard recipe of running SGD multiple times with different learning rates.

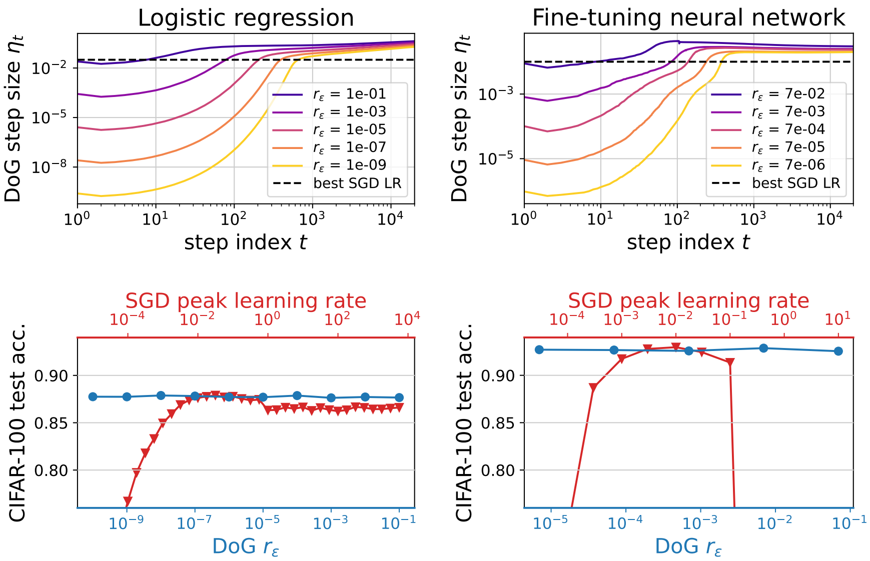


Figure 1: Illustration of DoG for CIFAR-100 classification using logistic regression on last-layer features of a pre-trained ViT-B/32 (left) or end-to-end fine-tuning of the model (right). The top row shows the DoG step size sequence η_t for different values of the initial movement r_ϵ, and the bottom row shows that DoG attains test error on par with carefully tuned SGD (with cosine annealing), even when varying r_ϵ by several orders of magnitude. See details in Appendix D.6.

Proposed algorithm. In this work, we use key insights from Carmon and Hinder [10] to go a step further and develop a parameter-free step size schedule. For SGD iterations of the form $x_{t+1} = x_t - \eta_t g_t$, where x_t denotes the model parameters at the t'th iteration and g_t denotes the stochastic gradient of the loss function, our proposed dynamic steps size is (for all $t \geq 1$)

$$\eta_t = \frac{\max_{i \le t}\|x_i - x_0\|}{\sqrt{\sum_{i \le t}\|g_i\|^2}}. \qquad \text{(DoG)}$$

In words, the step size at iteration t is the maximum distance to between the initial point and observed iterates, divided by the sum of squared stochastic gradient norms, i.e., Distance over Gradients (DoG). At the first step, we set η_0 to be $r_\epsilon/\|g_0\|$, i.e., we take a normalized gradient step of size r_ϵ; we show that, as long as r_ϵ is small, its precise setting has only mild effect.

Crucially, DoG has no multiplicative "learning rate" parameter: if one considers step sizes of the form $\eta_t = c \cdot \frac{\max_{i \le t}\|x_i - x_0\|}{\sqrt{\sum_{i \le t}\|g_i\|^2}}$ then $c = 1$ is a universally good setting (see Section 2 for a heuristic justification and Section 4.3 for empirical evidence for this claim).

Figure 1 highlights key aspects of DoG. The top row shows the DoG step size sequence for different values of r_ϵ in convex (left) and non-convex (right) stochastic optimization problems. The DoG step size increases rapidly (note the logarithmic x scale) and stabilizes around values close to the optimal SGD step size with little dependence on r_ϵ. The bottom row of the figure compares test errors obtained by DoG and SGD with various step sizes, showing that (for all choices of r_ϵ) DoG performs on par with well-tuned SGD.

1.1 Summary of results

Theoretical guarantees. In Section 3 we analyze DoG for stochastic convex optimization, assuming that the stochastic gradients are bounded for all points in $\mathcal{B}$, a ball around the initial point x_0 with radius $3d_0$, where d_0 is the distance between x_0 and an optimum.

First, we show that if the iterates of DoG remain in $\mathcal{B}$, then with high probability DoG achieves a convergence rate that is optimal up to a factor of $O(\log(1 + \frac{d_0}{r_\epsilon}))$. In practice, DoG appears to indeed be stable as long as r_ϵ is sufficiently small. However, DoG is not always stable: on pathological functions its iterates can move far from the optimum.

To address this, we consider a theoretical, tamed variant of DoG, which we call T-DoG, whose step sizes are smaller by a logarithmic factor. We prove that, with high probability, the T-DoG iterates never leave $\mathcal{B}$. Thus, we obtain a high probability parameter-free convergence guarantee is optimal up logarithmic factors.

To our knowledge, this is the first dynamic SGD step size schedule to attain such theoretical guarantee, and only the third high probability parameter-free guarantee in the literature (following [10, 101]). Moreover, it is the first parameter-free result assuming only locally bounded stochastic gradients. This is significant since the usually-assumed global stochastic gradient bound does not exist in many problems (including least squares).

Empirical study. Our experiments in Section 4 focus on fine-tuning neural networks, because this is a practically important setting that still allows for thorough experiments at a reasonable computational budget. We also perform a small-scale experiment with training a neural network from scratch. Our experiments span 23 natural language understanding and image classification tasks and 8 popular model architectures.[1]

Our results indicate that, compared to DoG, SGD with a cosine step size schedule and tuned base learning rarely attains a relative error improvement of more than 5% (e.g., the difference between accuracy 95% and 95.25%). For convex problems (linear probes), the relative difference in errors is below 1%. In our testbed, well-tuned Adam tends to outperform both SGD and DoG, but a layer-wise version of DoG (which we call L-DoG) closes some of this performance gap.

We also test the sensitivity of DoG to the value of r_ϵ. We find that for most model/task combinations, DoG performs consistently well across a wide range of r_ϵ values as our theory predicts. However, in certain cases, choosing r_ϵ to be too low results in poor performance. We provide some preliminary findings showing that this is due in part to batch normalization.

Put together, our theory and experiments suggest DoG has the potential to save significant computation currently spent on learning rate tuning at little or no cost in performance—especially if we reinvest some of the saved computation in training a larger model on more data.

2 Algorithm derivation

Before providing rigorous theoretical guarantees for DoG, in this section we explain the origin of the algorithm. Our starting point is the following result by Carmon and Hinder [10]. Suppose we run T iterations of SGD with fixed step size η, i.e., the recursion $x_{t+1} = x_t - \eta g_t$, where x_t is the SGD iterate and g_t is the stochastic gradient at step t. If, for some $c \in (0, 1)$, it happens to hold

[1] A reference PyTorch implementation of our proposed algorithms is available at github.com/formll/dog.

that

$$\eta = c \cdot \frac{\max_{k \le T} \|x_k - x_0\|}{\sqrt{\sum_{k \le T} \|g_k\|^2}}, \tag{1}$$

then the averaged iterates satisfies an excess loss bound that is at most a factor $\frac{1}{c(1-c^2)}$ larger than the worst-case optimal bound achieved by perfectly tuned SGD.[2]

The condition (1) is an implicit equation: it allows us to check whether the choice of step size η is good only after running T steps of SGD using that η. Solving this implicit equation therefore requires multiple calls to SGD. We derive the DoG step size sequence by making the equation explicit: we choose η_t so that equation (1) holds at each step. For $c = 1$, this yields the step size formula (DoG). Our reason for choosing $c = 1$ is that it is the threshold under which a solution to the implicit equation yields an optimal rate of convergence. Therefore, in practice we expect 1 to be close to the highest stable value of c, and thus obtain the best performance; we verify this empirically in Section 4.3.

3 Theoretical analysis

3.1 Preliminaries

Problem setting. Our goal is to minimize a loss function $f : \mathcal{X} \to \mathbb{R}$ where $\mathcal{X} \subseteq \mathbb{R}^m$ (including the unconstrained setting $\mathcal{X} = \mathbb{R}^m$ as an important special case). We perform our analysis under the following standard convexity assumption.

Assumption 1 (Convexity)**.** *The function f is convex, its domain $\mathcal{X}$ is closed and convex, and its minimum is attained at some $x_\star \in \mathcal{X}$, i.e., $f_\star := \inf_{x \in \mathcal{X}} f(x) = f(x_\star)$.*

In Appendix A we discuss a possible relaxation of convexity under which our results continue to hold.

To minimize f we assume that access to a *stochastic gradient oracle* $\mathcal{G}$. When queried at point $x \in \mathcal{X}$ the oracle returns a stochastic (sub)gradient estimator $\mathcal{G}(x)$ satisfying $\mathbb{E}[\mathcal{G}(x) \mid x] \in \partial f(x)$. With slight abuse of notation, we write $\nabla f(x) := \mathbb{E}[\mathcal{G}(x) \mid x]$. We make the following assumption, where $\|\cdot\|$ denotes the Euclidean norm.

Assumption 2 (Pointwise bounded stochastic gradients)**.** *There exists some continuous function $\ell : \mathcal{X} \to \mathbb{R}$ such that $\|\mathcal{G}(x)\| \le \ell(x)$ almost surely.*

Given Assumption 2 we also define

$$L_\star = \max_{x \in \mathcal{X} : \|x - x_0\| \le 3\|x_0 - x_\star\|} \ell(x). \tag{2}$$

Assumption 2 is weaker than conventional assumptions in parameter-free stochastic optimization, which either uniformly bound the stochastic gradients, i.e., $\|\mathcal{G}(x)\| \le L$ for all $x \in \mathcal{X}$ [see, e.g., 65, 19], or uniformly bound the gradient variance [42]. However, even least squares problems (with $\mathcal{G}(x) = (\langle a, x \rangle - b)a$ for random $a \in \mathbb{R}^m$ and $b \in \mathbb{R}$) violate both uniform bounds. In contrast, $L_\star$ is finite under the mild assumption that a, b are bounded random variables.

[2] This results holds in the non-stochastic case [10, Proposition 1], but a qualitatively similar results holds with high probability in the stochastic case as well [10, Proposition 3].

Algorithm statement. We study (projected) SGD with dynamic learning rate schedule $\{\eta_t\}$, i.e.,

$$x_{t+1} = \mathrm{Proj}_{\mathcal{X}}(x_t - \eta_t g_t)$$

where x_0 is a given initialization, $g_k \coloneqq \mathcal{G}(x_k)$, and $\mathrm{Proj}_{\mathcal{X}}(\cdot)$ is the Euclidean projection onto $\mathcal{X}$. To succinctly state and analyze DoG, we define the following quantities:

$$r_t \coloneqq \|x_t - x_0\| \ , \ \bar{r}_t = \max_{k \le t} r_k \vee r_\epsilon \text{ and } G_t \coloneqq \sum_{k=0}^{t} \|g_t\|^2,$$

where $a \vee b \coloneqq \max\{a, b\}$ and r_ϵ is a small user-specified initial movement size parameter. With this notation, we define a family of DoG-like learning rate schedules.

Definition 1. A step size schedule is DoG-*like* if

$$\eta_t = \frac{\bar{r}_t}{\sqrt{G'_t}}$$

for a positive nondecreasing sequence G'_t that depends only on $x_0, g_0, \ldots, g_t$ and satisfies $G'_t \ge G_t$.

DoG corresponds to simply setting $G'_t = G_t$; in Section 3.3 we consider a theoretical (or tamed) DoG-like algorithm for which we guarantee bounded iterates by making G'_t larger than G_t by polylogarithmic factors. Throughout, we bound the error of the weighted average sequence

$$\bar{x}_t \coloneqq \frac{1}{\sum_{k=0}^{t-1} \bar{r}_k} \sum_{k=0}^{t-1} \bar{r}_k x_k. \tag{3}$$

Finally, to streamline the analysis we define:

$$d_t \coloneqq \|x_t - x_\star\| \ , \ \bar{d}_t \coloneqq \max_{k \le t} d_k \ , \ \bar{\ell}_t \coloneqq \max_{k \le t} \ell(x_k),$$

and

$$\theta_{t,\delta} \coloneqq \log\left(\frac{60 \log(6t)}{\delta}\right).$$

Logarithm conventions. Throughout the paper log is base e and $\log_+(\cdot) \coloneqq 1 + \log(\cdot)$.

3.2 Optimality gap bounds assuming bounded iterates

In this section, we bound the optimality gap attained by any DoG-like algorithm. Our bounds depend on the quantities $\bar{r}_T$ and $\bar{\ell}_T$, and are nearly optimal when $\bar{r}_T = O(d_0)$ (i.e., the DoG iterates don't move too far away from x_0) and G'_T is not much larger than G_T. In the next section we describe a specific DoG-like algorithm that is guaranteed to satisfy both requirements.

Convexity and Jensen's inequality imply that $\bar{x}_t$ satisfies

$$f(\bar{x}_t) - f_\star \le \frac{1}{\sum_{k=0}^{t-1} \bar{r}_k} \sum_{k=0}^{t-1} \bar{r}_k \langle \nabla f(x_k), x_k - x_\star \rangle. \tag{4}$$

The sum in the RHS decomposes to two components:

$$\underbrace{\sum_{k=0}^{t-1} \bar{r}_k \langle g_k, x_k - x_\star \rangle}_{\text{weighted regret}} - \underbrace{\sum_{k=0}^{t-1} \bar{r}_k \langle \Delta_k, x_k - x_\star \rangle}_{\text{noise}}, \tag{5}$$

where $\Delta_k \coloneqq g_k - \nabla f(x_k)$. We give probability 1 bounds for the weighted regret (Lemma 1) and high probability bounds for the noise term (Lemma 2). In each case, the key challenge is replacing usual a-priori bounds on d_0 (or the domain size) with the empirically observed $\bar{r}_T$. We present and discuss each lemma in turn.

Lemma 1 (Weighted regret bound)**.** *If $\mathcal{X}$ is a closed convex set then any* DoG*-like scheme (Definition 1) satisfies* $\sum_{k=0}^{t-1} \bar{r}_k \langle g_k, x_k - x_\star \rangle \le \bar{r}_t(2\bar{d}_t + \bar{r}_t)\sqrt{G'_{t-1}}$, $\forall t \ge 1$.

Proof. Using $x_{k+1} = \mathrm{Proj}_{\mathcal{X}}(x_k - \eta_k g_k)$ we obtain the standard inequality $d_{k+1}^2 \le \|x_k - \eta_k g_k - x_\star\|^2 = d_k^2 - 2\eta_k \langle g_k, x_k - x_\star \rangle + \eta_k^2 \|g_k\|^2$. Rearranging this gives:

$$\langle g_k, x_k - x_\star \rangle \le \frac{d_k^2 - d_{k+1}^2}{2\eta_k} + \frac{\eta_k \|g_k\|^2}{2}. \tag{6}$$

Therefore, $\sum_{k=0}^{t-1} \bar{r}_k \langle g_k, x_k - x_\star \rangle$ is at most

$$\underbrace{\frac{1}{2}\sum_{k=0}^{t-1} \frac{\bar{r}_k}{\eta_k}(d_k^2 - d_{k+1}^2)}_{(A)} + \underbrace{\frac{1}{2}\sum_{k=0}^{t-1} \bar{r}_k \eta_k \|g_k\|^2}_{(B)}.$$

We will bound the terms (A) and (B) in turn, beginning with the former:

$$\begin{aligned}
(A) &= \sum_{k=0}^{t-1} \sqrt{G'_k}(d_k^2 - d_{k+1}^2) = d_0^2\sqrt{G'_0} - d_t^2\sqrt{G'_{t-1}} + \sum_{k=1}^{t-1} d_k^2\left(\sqrt{G'_k} - \sqrt{G'_{k-1}}\right) \\
&\overset{(i)}{\le} \bar{d}_t^2\sqrt{G'_0} - d_t^2\sqrt{G'_{t-1}} + \bar{d}_t^2 \sum_{k=1}^{t-1}\left(\sqrt{G'_k} - \sqrt{G'_{k-1}}\right) = \sqrt{G'_{t-1}}\left(\bar{d}_t^2 - d_t^2\right) \overset{(ii)}{\le} 4\bar{r}_t \bar{d}_t \sqrt{G'_{t-1}}.
\end{aligned}$$

Inequality (i) uses $d_k \le \bar{d}_t$ and that G'_k is nondecreasing as per Definition 1. Inequality (ii) holds since, for $s \in \arg\max_{k \le t} d_k$, we have $\bar{d}_t^2 - d_t^2 = d_s^2 - d_t^2 = (d_s - d_t)(d_s + d_t) \le \|x_s - x_t\|(d_s + d_t) \le (\bar{r}_s + \bar{r}_t)(d_s + d_t) \le 4\bar{r}_t\bar{d}_t$. Bounding the second term (B), we have:

$$(B) = \sum_{k=0}^{t-1} \frac{\bar{r}_k^2 \|g_k\|^2}{\sqrt{G'_k}} \le \sum_{k=0}^{t-1} \frac{\bar{r}_k^2 \|g_k\|^2}{\sqrt{G_k}} \le \bar{r}_t^2 \sum_{k=0}^{t-1} \frac{\|g_k\|^2}{\sqrt{G_k}} \le 2\bar{r}_t^2\sqrt{G_{t-1}},$$

where the final inequality uses the standard Lemma 4 with $a_k = G_k = \sum_{i \le k} \|g_i\|^2$. □

While the proof of Lemma 1 is similar to the analysis of adaptive SGD where $\eta_t = \frac{\rho}{\sqrt{G_t}}$ [31], there are a couple of key differences. First, the DoG step sizes can increase, which typically makes adaptive gradient methods difficult to analyze [75]. We bypass this difficulty by considering regret weighted by $\bar{r}_k$, which factors out the increasing portion of the step size. Second, the standard adaptive SGD analysis yields a bound proportional to $\bar{d}_t^2$ (typically further bounded using the domain diameter) rather than $\bar{r}_t\bar{d}_t$ as in our bound. This is a crucial difference, since—as we soon argue—$\bar{r}_t$ "cancels" when dividing through by $\sum_{k<t} \bar{r}_k$, while $\bar{d}_t$ does not. We obtain the improved result by keeping around the term $-d_t^2\sqrt{G'_{t-1}}$ in the bound for (A) above; a trick similar to Carmon and Hinder [10, Lemma 1].

Next, we handle the noise term in (5), recalling the notation $\Delta_t \coloneqq g_t - \nabla f(x_t)$ and $\theta_{t,\delta} \coloneqq \log\frac{60\log(6t)}{\delta}$.

Lemma 2 (Noise bound)**.** *Under Assumption 2, for all $\delta \in (0,1)$, $T \in \mathbb{N}$ and $L > 0$ we have*

$$\mathbb{P}\left(\exists t \le T : \left|\sum_{k=0}^{t-1} \bar{r}_k \left\langle \Delta_k, x_k - x_\star \right\rangle\right| \ge 8\bar{r}_{t-1}\bar{d}_{t-1}\sqrt{\theta_{t,\delta}G_{t-1} + \theta_{t,\delta}^2 L^2}\right) \le \delta + \mathbb{P}\big(\bar{\ell}_T > L\big).$$

The proof of Lemma 2 appears in Appendix C.1 and is based on a new concentration bound, Lemma 7, which allows us to bound the noise term despite having no deterministic bound on the magnitude of the martingale difference sequence $\bar{r}_k \left\langle \Delta_k, x_k - x_\star \right\rangle$. The proof of Lemma 7 involves combining time-uniform Bernstein bounds [37] and a general bound on the cumulative sums of sequence products (Lemma 5), which may be of independent interest.

Combining the above results, we obtain the following.

Proposition 1. *For all $\delta \in (0,1)$ and $L > 0$, if Assumption 1, Assumption 2, and Definition 1 hold then with probability at least $1 - \delta - \mathbb{P}\big(\bar{\ell}_T > L\big)$, for all $t \le T$ the optimality gap $f(\bar{x}_t) - f_\star$ is upper bounded by*

$$O\left(\frac{(d_0 + \bar{r}_t)\sqrt{G'_{t-1} + G_{t-1}\theta_{t,\delta} + L^2\theta_{t,\delta}^2}}{\sum_{i<t} \bar{r}_i / \bar{r}_t}\right).$$

Proof. Follows from Equations (4) and (5), Lemma 1, Lemma 2 and the fact that $\bar{d}_t \le d_0 + \bar{r}_t$. □

The following algebraic fact shows that there is always an iteration $\tau \le T$ where the denominator $\sum_{i<t} \frac{\bar{r}_i}{\bar{r}_t} \ge \Omega(T / \log \frac{\bar{r}_T}{r_\epsilon})$; see Appendix B.3 for proof.

Lemma 3. *Let $s_0, s_1, \ldots, s_T$ be a positive increasing sequence. Then*

$$\max_{t \le T} \sum_{i<t} \frac{s_i}{s_t} \ge \frac{1}{e}\left(\frac{T}{\log_+(s_T/s_0)} - 1\right).$$

Combining Proposition 1 and Lemma 3 yields the following (see short proof in Appendix C.2).

Corollary 1. *Under the setting of Proposition 1, let $\tau \in \arg\max_{t \le T} \sum_{i<\tau} \frac{\bar{r}_i}{\bar{r}_t}$. Then, with probability at least $1 - \delta - \mathbb{P}(\bar{\ell}_T > L)$, the optimality gap $f(\bar{x}_\tau) - f_\star$ is*

$$O\left(\log_+\left(\frac{\bar{r}_\tau}{r_\epsilon}\right)\frac{(d_0 + \bar{r}_\tau)\sqrt{G'_{\tau-1} + G_{\tau-1}\theta_{\tau,\delta} + L^2\theta_{\tau,\delta}^2}}{T}\right).$$

Corollary 1 is immediately useful is when $\mathcal{X}$ is bounded but its exact diameter unknown, for example when $\mathcal{X}$ is a polytope as is common in two-stage stochastic programming [59].

Simplifying the bound for typical DoG trajectories. Suppose that DoG iterates satisfy $\bar{r}_T \le 3d_0$, which implies that $\bar{\ell}_T \le L_\star$ and therefore (for DoG) $G'_t = G_t \le L_\star^2 T$. Substituting into Corollary 1 yields an optimality gap bound of $O\Big(\frac{d_0 L_\star}{\sqrt{T}}\theta_{T,\delta}\log\frac{\bar{r}_T}{r_\epsilon}\Big)$, which is minimax optimal up a term double-logarithmic in T and logarithmic in $\frac{1}{r_\epsilon}$ [2].

Furthermore, in realistic DoG trajectories, even the multiplicative term $\log\frac{\bar{r}_T}{r_\epsilon}$ is likely too pessimistic. This is because $\bar{r}_t$ typically increases rapidly for $t_0 < 1000$ steps and then plateaus (see Figure 6 in the appendix). Consequently, $\bar{r}_i / \bar{r}_t \ge 1/10$ for most of the optimization trajectory, and $\sum_{i<t} \frac{\bar{r}_i}{\bar{r}_t} \ge t/10 - t_0$. Substituting back into Proposition 2, we get that $\bar{x}_T$ is $O\Big(\frac{d_0 L_\star}{\sqrt{T - t_0}}\theta_{T,\delta}\Big)$ suboptimal.

DoG can run wild. While DoG is empirically stable, there exist (non-stochastic) examples where $\bar{r}_t$ grows much larger than d_0: in Appendix C.3 we describe a variant of Nermirovski's function [60] for which $\bar{r}_t = r_\epsilon\sqrt{t}$ and therefore $\bar{r}_t/d_0$ diverges as t grows. Next, we show that by slightly decreasing the DoG step sizes we can guarantee that $\bar{r}_T/d_0 \le 3$ with high probability.

3.3 Iterate stability bound

This section introduces a new DoG-*like* step size scheme whose iterates are guaranteed to remain bounded with high probability. We call this scheme T-DoG, where the T stands for "theoretical" or "tamed." The step sizes depend on the iteration budget T, the failure probability δ, and an upper bound L on $L_\star$ (defined in Equation (2)), and are given by $\eta_t = \bar{r}_t/\sqrt{G'_t}$, where

$$G'_t = 8^4\theta_{T,\delta}\log_+^2(t+1)(G_{t-1} + 16\theta_{T,\delta}\|g_t\|^2 \vee L^2). \tag{T-DoG}$$

(recalling that $a \vee b := \max\{a,b\}$ and using $G_{-1} := 0$). The dependencies on T, δ and L are weak, since $\theta_{t,\delta} := \log\left(\frac{\log(6t)}{\delta}\right)$ and the L-dependent term (that barely grows with t) will typically be smaller than the term proportional to G_{t-1}.

With the definition of T-DoG in hand, we are ready to state its key property: guaranteed iterate stability.

Proposition 2. *Suppose that Assumptions 1 and 2 hold and $r_\epsilon \le 3d_0$. For any $\delta \in (0,1)$, $T \in \mathbb{N}$ and $L \ge L_\star$, the iterations of T-DoG satisfy $\mathbb{P}(\bar{r}_T > 3d_0) \le \delta$.*

The proof of Proposition 2 uses the standard inequality (6) combined with a time-uniform empirical Bernstein concentration result (Corollary 3, based on Howard et al. [37]) to bound the effect of gradient noise. We defer the full proof to Appendix C.4 and proceed to highlight the key argument by proving the result in the noiseless case.

Proof of Proposition 2 in the noiseless case. In the noiseless case we have $g_k = \nabla f(x_k)$ and therefore $\langle g_k, x_k - x_\star\rangle \ge f(x_k) - f_\star \ge 0$. Substituting into (6) and rearranging gives $d_{k+1}^2 - d_k^2 \le \eta_k^2\|g_k\|^2$. Assuming by induction that $\bar{r}_t \le 3d_0$ and telescoping yields

$$d_{t+1}^2 - d_0^2 \le \bar{r}_t^2\sum_{k=0}^{t}\frac{\|g_k\|^2}{G'_k} \overset{(i)}{\le} \frac{\bar{r}_t^2}{8^4}\sum_{k=0}^{t}\frac{G_k - G_{k-1}}{(G_k + L^2)\log_+^2\frac{G_k+L^2}{L^2}} \overset{(ii)}{\le} \frac{\bar{r}_t^2}{8^4} \overset{(iii)}{\le} \frac{9d_0^2}{8^4} \implies d_{t+1} \le 2d_0,$$

where (i) uses that $\|g_k\|^2 = G_k - G_{k-1}$ (with the shorthand $G_{-1} := 0$) and

$$G'_k \ge 8^4(G_{k-1} + \|g_k\|^2 + L^2)\log_+^2(k+1) \ge 8^4(G_k + L^2)\log_+^2\frac{G_k + L^2}{L^2}$$

by the inductive assumption that $\bar{r}_t \le 3d_0$ and hence $G_k \le kL^2$ for all $k \le t$, (ii) uses Lemma 6 with $a_k = G_k + L^2$, and (iii) uses $\bar{r}_t \le 3d_0$ again. Therefore, $r_{t+1} \le d_{t+1} + d_0 \le 3d_0$ by the triangle inequality, completing the induction step. □

Finally, we state the main guarantee for T-DoG.

Theorem 1. *Suppose that Assumptions 1 and 2 hold. For any $\delta \in (0,\frac{1}{2})$, $T \in \mathbb{N}$ and $L \ge L_\star$, consider T iterations of T-DoG with $r_\epsilon \le 3d_0$. Then for $\tau \in \arg\max_{t\le T}\sum_{i<\tau}\bar{r}_i/\bar{r}_t$ we have, with probability at least $1-2\delta$, that $f(\bar{x}_\tau) - f_\star$ is upper bounded by*

$$O\left(c_{\delta,r_\epsilon,T}\frac{d_0\sqrt{G_\tau + L^2}}{T}\right) \le O\left(c_{\delta,r_\epsilon,T}\frac{d_0(L_\star + L/\sqrt{T})}{\sqrt{T}}\right),$$

where $c_{\delta,r_\epsilon,T} = \log_+(T)\log_+\left(\frac{d_0}{r_\epsilon}\right)\log\left(\frac{\log_+(T)}{\delta}\right)$.

Proof. Follows by Corollary 1, Proposition 2 and (T-DoG). □

Theorem 1 yields the optimal convergence bound [2] up to logarithmic factors. To the best of our knowledge this is the first parameter-free stochastic optimization method that does not require the stochastic gradients to be uniformly bounded across the domain $\mathcal{X}$ and instead produces a bound whose leading order term depends on the 'local' gradient bound $L_\star$.

4 Experiments

To test DoG in practical scenarios, we perform extensive experiments over a diverse set of tasks and model architectures in both the vision and language domains. We construct a testbed that consists of over 20 tasks and 7 model architecture, covering natural language understanding and computer vision (Section 4.1). In this testbed we compare DoG to SGD and Adam (Section 4.2), showing that DoG performs on par with tuned SGD, but not as well as tuned Adam. Nevertheless, a per-layer version of DoG (defined below) closes much of this gap with Adam without requiring tuning. We also use our testbed to analyze the sensitivity of DoG to its fixed parameters (Section 4.3), and demonstrate its effectiveness in convex logistic regression settings (Section 4.4). Finally, we experiment with fine-tuning a CLIP model on ImageNet (Section 4.5) and training a CIFAR10 model from scratch (Section 4.6). PyTorch implementation of DoG is available at `https://github.com/formll/dog`.

Layer-wise DoG. Neural models in general and transformer-based models in particular often benefit from using a per-parameter or per-layer step sizes [45, 98]. With this in mind, we consider a per-layer version of DoG, which we call L-DoG, where we apply the (DoG) formula separately for every layer. Namely, if we consider x_t^l to be the weights in layer l at time t,[3] then we set the learning rate for that layer to be $\eta_t^l = \frac{\max_{i \le t} \|x_i^l - x_0^l\|}{\sqrt{\sum_{i \le t} \|g_i^l\|^2 + \epsilon}}$, where $\epsilon = 10^{-8}$ is added to the denominator for numerical stability. While we do not provide theoretical guarantees for L-DoG, we show below that it performs well in practice.

4.1 Fine-tuning testbed

Our main experiments focus on fine-tuning pre-trained models, which allows us to experiment with advanced models while also thoroughly tuning the learning rate for the baseline optimizers, using an academic computational budget.

Common hyperparameters. For each baseline algorithm, we use best-practice learning rate schedule (cosine annealing for all experiments, with a warm-up stage for language experiments) and sweep over the peak learning rate for each model/task pair. We give each pair a fixed step budget designed to suffice for convergence, performing evaluation throughout the training. In all cases, we use polynomial decay averaging[4] as proposed by Shamir and Zhang [78], and select the best checkpoint (either averaged or not) based on evaluation performance. We repeat relevant learning setups with 5 different seeds, and report the mean performance across the seeds. For simplicity, we do not use weight decay throughout. The complete set of hyper-parameters appears in Appendix D.

[3] More precisely, our implementation treats each element in the PyTorch `.parameters()` list as a separate layer.

[4] We apply the weight averaging with a fixed parameter ($\gamma = 8$, following [49]); we did not try any other parameter in our experiments.

Natural language understanding (NLU). To test DoG's efficacy in modern NLU, we use it to fine-tune transformer language models [84] on the well-studied GLUE benchmark [88] which measures models' performance on diverse text classification tasks (listed in Appendix D.3).

Additionally, we fine-tune models on SQuAD 1.1, a question answering dataset [74]. We fine-tune a RoBERTa-base [51] checkpoint and T5-base [73].[5] For each task, we use the official evaluation metrics defined in by Wang et al. [88] and Rajpurkar et al. [74] as well as their original proposed splits, and report the results over the evaluation set.

Computer vision. We also fine-tune 5 models architectures on 12 different computer vision tasks from the VTAB benchmark [100] (see Appendix D.3); of the other 7 tasks in VTAB, 5 are trivial (accuracy greater than 99%) and 2 have small validation splits leading to unreliable model selection. We follow the training, validation and test splits defined in VTAB, and report performance on the test split (using the validation split for model selection). We fine-tune 5 models: VGG11 [80], ResNet50 [35], Densenet121 [38], ViT-B/32 [26], and ConvNeXt-T [52], where the ViT model is pre-trained on ImageNet 21K and the others are trained on ImageNet 1K [24].

Normalized performance metric. Since the performance metrics in our testbed vary substantially across tasks and models, they are challenging to compare in aggregate. To address this, we consider the following notion of *relative error difference* (RED), that provides a normalized measure of a difference between two model's performance. In particular, given a task and a model architecture, let err_x be the error[6] of the model when trained with optimizer x (Adam or SGD with a certain learning rate, or L-DoG) and let $\mathrm{err}_{\mathrm{DoG}}$ be the error when trained with DoG. Then

$$\mathrm{RED}(\mathrm{err}_x, \mathrm{err}_{\mathrm{DoG}}) \coloneqq \frac{\mathrm{err}_{\mathrm{DoG}} - \mathrm{err}_x}{\mathrm{err}_{\mathrm{DoG}}}.$$

A positive RED value indicates that optimizer x is better than DoG, and a negative value indicates the opposite. When the absolute value of RED is beneath a few percentage points, the compared methods are nearly equivalent.

Setting r_ϵ. Our theoretical analysis suggest that the particular choice of r_ϵ does not matter as long as it is sufficiently small relative to the distance between the weight initialization x_0 to the optimum. Consequently, for vision experiments we set $r_\epsilon = \alpha \cdot (1 + \|x_0\|)$ for $\alpha = 10^{-4}$, assuming that the distance to the optimum is more than 0.01% of the initialization norm. For language experiments, this assumption turned out to be wrong (causing DoG to diverge in some cases), and we decreased α to 10^{-6} for DoG and to 10^{-8} for L-DoG, where the additive 10^{-6} term was too large in some layers. We believe that 10^{-6} and 10^{-8} should be good defaults for DoG and L-DoG, respectively, though networks with batch normalization or different initialization schemes could require a larger value; see Section 4.3 for additional discussion.

4.2 Comparison of fine-tuning performance

Figure 2 depicts the median, IQR (inter-quantile range) and mean RED of each model, when trained with SGD and Adam with different peak learning rates. The figure shows that, when comparing across models, there is no good default learning rate for neither SGD nor Adam. Moreover, even for a single model only very specific SGD learning rate performs well, while most are considerably

[5] Throughout the paper we often use the shorthand names RoBERTa-b and T5-b, respectively.

[6] We consider the error to be 1 minus the respective performance metric, as detailed in Table 3.

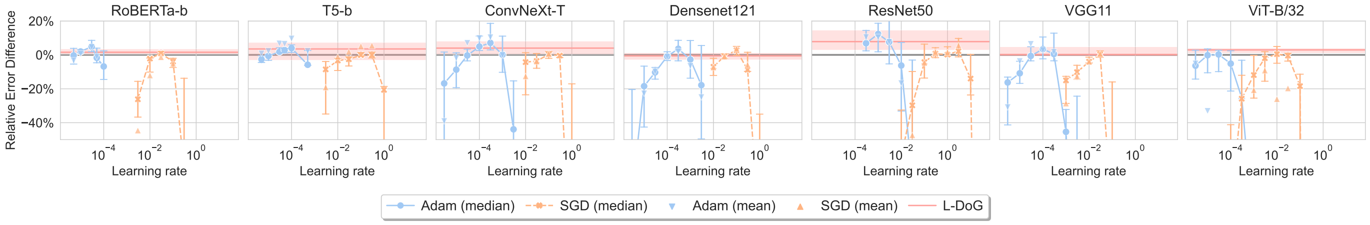


Figure 2: Relative error difference statistics (median, mean, and error bars showing IQR) across tasks for each model, as a function of peak learning rate. The red horizontal line and shaded region indicate the median and IQR RED for L-DoG, respectively.

inferior to using DoG. Even when tuned to the best *fixed* learning-rate value per model (which we refer to as *model tuned LR*), some tasks may still fail (compared to DoG) as indicated by the large IQR and the gap between the mean (triangles) and the median RED (circles) in models such as ViT-B/32 ad Densenet121. While Adam also requires tuning, it is somewhat less sensitive than SGD to the choice of peak learning rate. For a full breakdown of performance per task, see Figure 7 and Tables 4 and 5 in Appendix E.1.

DoG performs similarly to well-tuned SGD in 79 out of the 80 model/task combinations in our testbed. The one exception is tuning T5-b on CoLA, where DoG behaves erratically while SGD succeeds only with a few learning rates. In contrast, both Adam and L-DoG achieved reasonable performance consistently. DoG's poor performance on CoLA results in high RED measures for this case, which draw the mean RED (triangles) above the median one in Figure 2 for T5-b. We further analyze this exception in Appendix E.2 and show that choosing significantly smaller r_ϵ for DoG alleviates the problem.

Figure 3 (top) compares DoG to SGD with model tuned LR as defined above, as well as *instance tuned LR*, where for each model/task pair we select the best learning rate, at a computational expense 6–7 times larger than running DoG. The performance of DoG remains close to that of SGD with instance-tuned LR, with the largest median RED observed for ResNet50 and ViT-B/32.

Figure 3 (bottom) compares DoG to model-tuned and instance-tuned Adam, as well as to L-DoG. In a few cases (namely ResNet50 and ConvNeXt-T) the gaps between DoG and Adam are significant, and favor Adam. We hypothesize this is due to Adam's per-parameter step-sizes and momentum mechanisms, which DoG does not exploit. We note that L-DoG, which has per-layer steps, has positive median RED for all models, and closes a good deal of the gap between DoG and Adam, particularly for ResNet50.

4.3 Sensitivity of DoG's fixed parameters

Initial movement size r_ϵ. Our theory suggests that all sufficiently small choices of r_ϵ should perform similarly, but choosing r_ϵ too large (compared to the initial distance to the optimum) can hurt the performance of the algorithm. In Figure 4 (left) we plot the test performance as a function of r_ϵ for 8 model/task combinations. For 7 out of the 8, DoG is highly robust to the value of r_ϵ as long as it small enough, as predicted. However, ResNet50 on CIFAR-100 (bottom left) is an exception, where smaller values of r_ϵ result in an accuracy drop. We hypothesize this is due to scale invariance introduced by batch normalization (BN), and provide supporting evidence for that in Appendix E.3 (Figure 9), where we show that DoG is insensitive to r_ϵ when we turn off BN. In the appendix we also provide a complementary diagnostic for r_ϵ sensitivity by plotting η_t vs. η_0 for different values of t (see Figure 8).

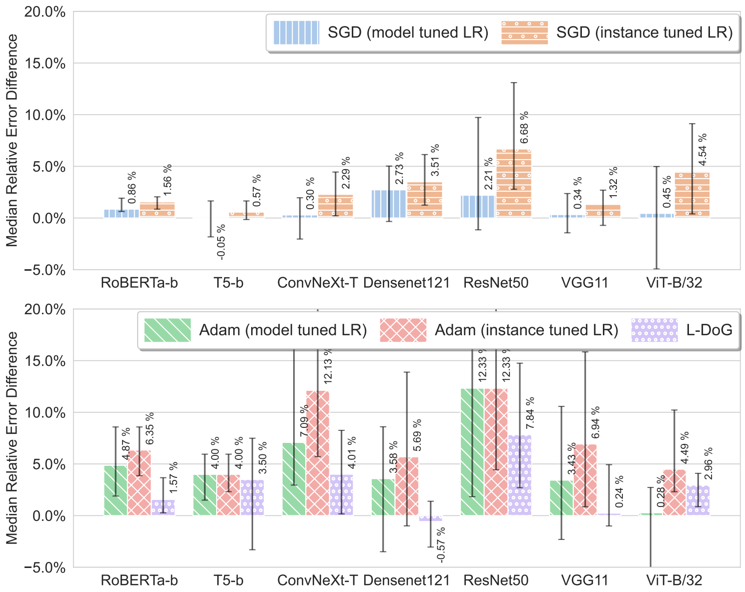


Figure 3: RED median (bar chart) and IQR (error bars) of each model on the set of applicable tasks. **Top**: Comparison with SGD when the LR is optimally tuned per model (*model tuned LR*) or per task (*instance tuned LR*). DoG is competitive with model-tuned SGD and often performs nearly as well as instance-tuned SGD. **Bottom**: Comparison of DoG with adaptive optimizers. L-DoG closes most of the gap to Adam.

Base learning rate. For this experiment only, we consider variants of DoG with different values of base learning, i.e., step sizes of the form $\eta_t = c \cdot \frac{\max_{i \le t} \|x_i - x_0\|}{\sqrt{\sum_{i \le t} \|g_i\|^2}}$ with different values of c. We expect optimal performance when c is close to 1. More specifically, we expect the algorithm to be unstable when $c > 1$ and to be slower to converge (and less likely to generalize well) when $c < 1$. As can be observed in Figure 4 (right), values around $c = 1$ perform well for all models. For smaller values, there is indeed inferior performance in some models (mainly ResNet50 and RoBERTa-b), while larger values result in divergence (in 6 out of 8 cases). We conclude that the useful range for c is very narrow (about [0.5, 1.5]) and tuning it is not likely to produce significant improvements. This is in contrast to Adam and SGD which generally require searching over a space spanning a few orders of magnitude to properly train a model.

4.4 Convex optimization

We also evaluate DoG on convex optimization tasks, matching the assumptions of our theoretical analysis. To do so, we perform multi-class logistic regression on features obtained from the computer vision models in our testbed, i.e., we perform linear probes. We find that model-tuned SGD performs on-par or worse than DoG, while instance-tuned SGD barely gains any advantage over DoG (Figure 5), with RED values well under 1% (corresponding to the difference between accuracies 90% and 90.1%). Moreover, even in this simple setting, SGD is sensitive to the choice of learning rate, which differ significantly between models (Figure 6).

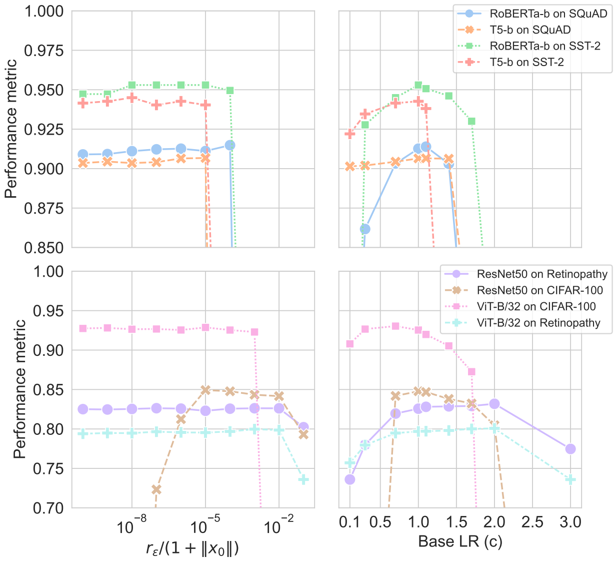


Figure 4: Performance metrics of models trained with DoG as a function of η_0 (left) or the base learning rate (right).

4.5 Fine-tuning on ImageNet

To complement our main fine-tuning testbed, we perform a more limited experiment involving ImageNet as a downstream task, which is more expensive to tune due its larger scale. We fine-tune a ViT-B/32 CLIP model [72] and compare DoG and L-DoG to training with SGD as well as to an AdamW [54] training prescription similar to Wortsman et al. [95]; see Appendix D.7 for additional details. Table 1 shows the ImageNet top-1 validation accuracies of the final model checkpoints, with and without the polynomial decay averaging used throughout our experiments. DoG performs similarly to SGD, but both algorithm perform significantly worse than Adam, perhaps due to an insufficient iteration budget. L-DoG performs well in this setting, improving on AdamW by a little over 1 point.

4.6 Training from scratch

We conduct a preliminary experiment with training a model from scratch, specifically a Wide ResNet 28-10 [99] on CIFAR-10 [47]; see Appendix D.8 for details. Table 2 shows the test accuracy of the final checkpoint, with and without the polynomial averaging used throughout our experiments. Here DoG performs on par with the setting's canonical training prescription of SGD with momentum 0.9 and learning rate 0.1 [18]. In this setting Adam produces poorer results, and L-DoG is 0.5 point worse than tuned Adam with the best learning rate, perhaps due to not reaching convergence.

5 Related work

Previous attempts to design theoretically principled and practical optimization algorithms that do not require learning rate tuning approach the problem from a variety of perspectives, resulting in a large variety of proposed algorithms. Rolinek and Martius [76], Vaswani et al. [85], Paquette

Algorithm	LR	Acc. w/o averaging	Acc. with averaging
SGD	1e-03	60.70%	60.49%
	3e-03	73.62%	73.54%
	1e-02	76.82%	76.80%
	3e-02	77.51%	77.54%
	1e-01	75.73%	75.71%
DoG	-	74.78%	77.22%
AdamW	1e-05	78.23%	78.25%
	3e-05	79.04%	79.01%
	1e-04	75.02%	74.97%
L-DoG	-	78.20%	**80.12%**

Table 1: ImageNet top-1 validation accuracies after fine-tuning a CLIP ViT-B/32 model for 25K training steps, with and without polynomial decay averaging (see Section 4.5).

Algorithm	LR	Acc. w/o averaging	Acc. with averaging
SGD	0.1	94.9%	94.9%
	0.3	95.8%	95.6%
	1	**96.4%**	84.4%
	3	95.9%	21.7%
	10	10.0%	10.0%
SGD w/ mom. 0.9	0.01	95.0%	95.1%
	0.03	95.8%	95.7%
	0.1 †	96.3%	88.5%
	0.3	95.8%	27.5%
	1	42.0%	63.4%
DoG	-	85.2%	**96.4%**
Adam	3e-05	91.1%	91.1%
	1e-04	94.0%	94.0%
	3e-04	93.5%	93.8%
	1e-03	91.4%	91.6%
L-DoG	-	83.2%	93.5%

Table 2: CIFAR-10 test accuracies after training a Wide ResNet 28-10 model from scratch for 200 epochs, with and without polynomial decay averaging (see Section 4.6). † denotes the standard training configuration for this setting [cf. 18, Table 2].

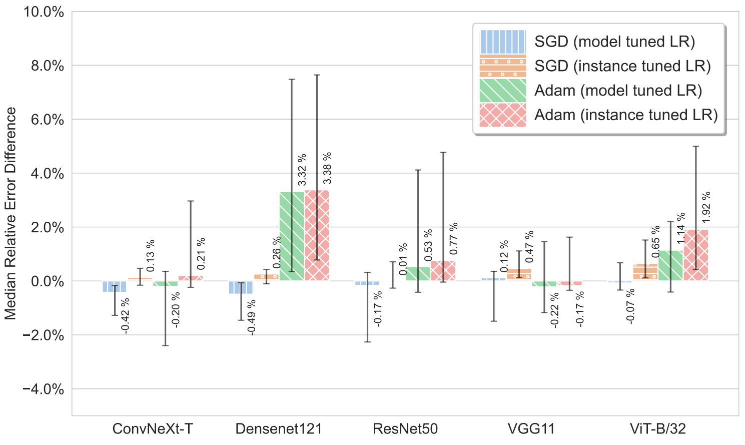


Figure 5: RED median and IQR (as in Figure 3) in tn the *convex optimization* setting (Section 4.4).
also

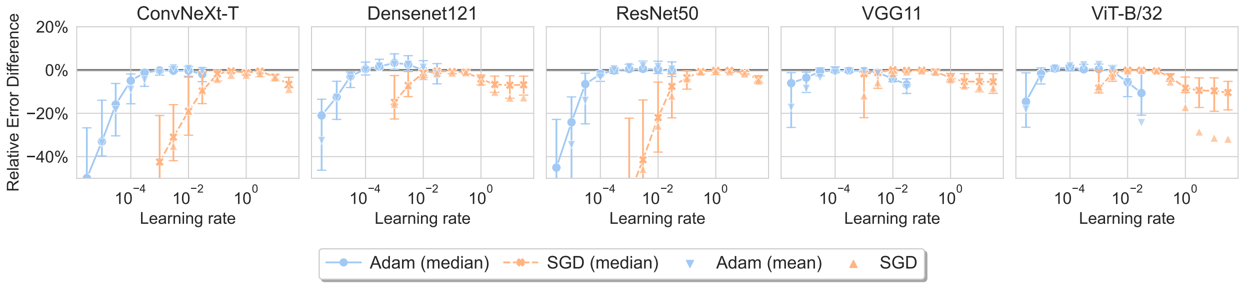


Figure 6: Per-learning rate RED statistics (as in Figure 2) in the *convex optimization* setting (Section 4.4).

and Scheinberg [67] lift classical line search technique from non-stochastic optimization to the stochastic setting, while Berrada et al. [8], Loizou et al. [53] do the same for the classical Polyak step size [71, 34]. Asi and Duchi [4] develop a class of algorithms based on stochastic proximal methods and demonstrate their improved robustness both theoretically and empirically. Schaul et al. [77] use a stochastic quadratic approximation for designing learning rates that maximize the expected one-step objective decrease. Chandra et al. [13] nest hypergradient descent to make a method that is insensitive to initial hyper-parameter choices. However, none of these results are parameter-free in the same sense as DoG: they either do not have converges guarantees, or have suboptimality bounds that blow up polynomially when the method's parameters do not match a problem-dependent value. In contrast, parameter-free methods have converges rates that depend at most logarithmically on algorithmic parameters.

While the parameter-free optimization literature has focused mainly on theoretical schemes, Orabona and Tommasi [65] build on coin-betting schemes to design an algorithm for training neural networks that has AdaGrad-style convergence guarantees for quasi-convex functions. However, the algorithm is fairly different in structure from standard stochastic gradient methods, and underperforms tuned gradient methods in terms of test error. In recent work Chen et al. [14] design an improved algorithm that leverages coin betting and truncated linear models, showing more promising empirical performance. However, this method lacks theoretical guarantees.

Finally, in recent independent work Defazio and Mishchenko [22] propose a parameter-free dynamic step size schedule of dual averaging. While our work has the same motivation and shares a number of technical similarities (including the use of weighted regret bounds and an independently obtained Lemma 3) the proposed algorithms are quite different, and dual averaging is rarely used in training neural networks. Moreover, Defazio and Mishchenko [22] only prove parameter-free rates of convergence in the non-stochastic setting, while we establish high probability guarantees in the stochastic setting. Concurrently with our work, Defazio and Mishchenko [23] heuristically extended their dual averaging scheme to SGD- and Adam-like algorithms, reporting promising experimental results. Since our experimental testbed is fairly different from that of [23], empirical comparison of the algorithms will require further work.

6 Limitations and outlook

Our theoretical and empirical results place DoG as a promising step toward a new generation of principled and efficient tuning-free optimization algorithms. However, much additional work is necessary for these algorithms to become ubiquitous. First, it is important to understand how to correctly combine DoG with proven technique such as momentum, per-parameter learning rates, and learning rate annealing—this is a challenge both from a theoretical and a practical perspective. Second, it is important to gain a better understanding of situations where DoG is more sensitive to the choice of r_ϵ than theory would have us expect. Our preliminary investigations suggest a connection to batch normalization, and following that lead could lead to even more robust training methods. Finally, while our experiments aim to cover a broad range of tasks and architectures, future work needs to explore DoG in additional settings, particularly those involving training from scratch.

Acknowledgments

We thank Mitchell Wortsman and Simon Kornblith for their insightful comments. This research was partially supported by the Israeli Science Foundation (ISF) grant no. 2486/21, the Alon Fellowship, the Yandex Initiative for Machine Learning, the Israeli council of higher education, and the Len Blavatnik and the Blavatnik Family Foundation.

References

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL https://www.tensorflow.org/. Software available from tensorflow.org.

[2] Alekh Agarwal, Peter L Bartlett, Pradeep Ravikumar, and Martin J Wainwright.

Information-theoretic lower bounds on the oracle complexity of stochastic convex optimization. *IEEE Transactions on Information Theory*, 58(5):3235–3249, 2012.

[3] Kenneth J Arrow and Alain C Enthoven. Quasi-concave programming. *Econometrica: Journal of the Econometric Society*, pages 779–800, 1961.

[4] Hilal Asi and John C Duchi. The importance of better models in stochastic optimization. *Proceedings of the National Academy of Sciences*, 116(46):22924–22930, 2019.

[5] Roy Bar Haim, Ido Dagan, Bill Dolan, Lisa Ferro, Danilo Giampiccolo, Bernardo Magnini, and Idan Szpektor. The second PASCAL recognising textual entailment challenge. 2006.

[6] Charles Beattie, Joel Z Leibo, Denis Teplyashin, Tom Ward, Marcus Wainwright, Heinrich Küttler, Andrew Lefrancq, Simon Green, Víctor Valdés, Amir Sadik, et al. Deepmind lab. *arXiv preprint arXiv:1612.03801*, 2016.

[7] Luisa Bentivogli, Ido Dagan, Hoa Trang Dang, Danilo Giampiccolo, and Bernardo Magnini. The fifth PASCAL recognizing textual entailment challenge. 2009.

[8] Leonard Berrada, Andrew Zisserman, and M Pawan Kumar. Training neural networks for and by interpolation. In *International Conference on Machine Learning*, pages 799–809, 2020.

[9] Aditya Bhaskara, Ashok Cutkosky, Ravi Kumar, and Manish Purohit. Online learning with imperfect hints. In *International Conference on Machine Learning*, pages 822–831. PMLR, 2020.

[10] Yair Carmon and Oliver Hinder. Making SGD parameter-free. *Conference on Learning Theory*, 2022.

[11] Yair Carmon, Aditi Raghunathan, Ludwig Schmidt, John C Duchi, and Percy S Liang. Unlabeled data improves adversarial robustness. *Advances in Neural Information Processing Systems*, 2019.

[12] Daniel Matthew Cer, Mona T. Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. Semeval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *International Workshop on Semantic Evaluation*, 2017.

[13] Kartik Chandra, Audrey Xie, Jonathan Ragan-Kelley, and Erik Meijer. Gradient descent: The ultimate optimizer. *Advances in Neural Information Processing Systems*, 2022.

[14] Keyi Chen, John Langford, and Francesco Orabona. Better parameter-free stochastic optimization with ode updates for coin-betting. *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022.

[15] Gong Cheng, Junwei Han, and Xiaoqiang Lu. Remote sensing image scene classification: Benchmark and state of the art. *Proceedings of the IEEE*, 105(10):1865–1883, 2017.

[16] Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *CVPR*, pages 3606–3613, 2014.

[17] Comet.ML. Comet.ML home page, 2021. URL https://www.comet.ml/.

[18] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. AutoAugment: Learning augmentation strategies from data. In *Conference on Computer Vision and Pattern Recognition*, pages 113–123, 2019.

[19] Ashok Cutkosky and Francesco Orabona. Black-box reductions for parameter-free online learning in Banach spaces. In *Conference On Learning Theory*, pages 1493–1529, 2018.

[20] Ido Dagan, Oren Glickman, and Bernardo Magnini. The PASCAL recognising textual entailment challenge. In *Machine learning challenges. evaluating predictive uncertainty, visual object classification, and recognising tectual entailment*, pages 177–190. Springer, 2006.

[21] Damek Davis and Dmitriy Drusvyatskiy. Stochastic model-based minimization of weakly convex functions. *SIAM Journal on Optimization*, 29(1):207–239, 2019.

[22] Aaron Defazio and Konstantin Mishchenko. Parameter free dual averaging: Optimizing lipschitz functions in a single pass. In *OPT 2022: NeurIPS Workshop on Optimization for Machine Learning*, 2022.

[23] Aaron Defazio and Konstantin Mishchenko. Learning-rate-free learning by D-adaptation. *arXiv:2301.07733*, 2023.

[24] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[25] William B. Dolan and Chris Brockett. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*, 2005. URL https://aclanthology.org/I05-5002.

[26] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *ArXiv*, abs/2010.11929, 2020.

[27] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(7), 2011.

[28] Matthew Faw, Isidoros Tziotis, Constantine Caramanis, Aryan Mokhtari, Sanjay Shakkottai, and Rachel Ward. The power of adaptivity in sgd: Self-tuning step sizes with unbounded gradients and affine variance. In *Conference on Learning Theory (COLT)*, 2022.

[29] Li Fei-Fei, Rob Fergus, and Pietro Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In *CVPR Workshop*, 2004.

[30] Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and Bill Dolan. The third PASCAL recognizing textual entailment challenge. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, pages 1–9, Prague, 2007. Association for Computational Linguistics. URL https://aclanthology.org/W07-1401.

[31] Vineet Gupta, Tomer Koren, and Yoram Singer. A unified approach to adaptive regularization in online and stochastic optimization. *arXiv:1706.06569*, 2017.

[32] Vineet Gupta, Tomer Koren, and Yoram Singer. Shampoo: Preconditioned stochastic tensor optimization. In *International Conference on Machine Learning (ICML)*, 2018.

[33] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, 2020. doi: 10.1038/s41586-020-2649-2. URL https://doi.org/10.1038/s41586-020-2649-2.

[34] Elad Hazan and Sham Kakade. Revisiting the Polyak step size. *arXiv:1905.00313*, 2019.

[35] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2015.

[36] Oliver Hinder, Aaron Sidford, and Nimit Sohoni. Near-optimal methods for minimizing star-convex functions and beyond. In *Conference on Learning Theory*, pages 1894–1938, 2020.

[37] Steven R Howard, Aaditya Ramdas, Jon McAuliffe, and Jasjeet Sekhon. Time-uniform, nonparametric, nonasymptotic confidence sequences. *The Annals of Statistics*, 49(2):1055–1080, 2021.

[38] Gao Huang, Zhuang Liu, and Kilian Q. Weinberger. Densely connected convolutional networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2261–2269, 2016.

[39] Shankar Iyer, Nikhil Dandekar, and Kornel Csernai. First quora dataset release: Question pairs, 2017. URL https://data.quora.com/First-Quora-Dataset-Release-Question-Pairs.

[40] Andrew Jacobsen and Ashok Cutkosky. Parameter-free mirror descent. 2022.

[41] Justin Johnson, Bharath Hariharan, Laurens Van Der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. CLEVR: A diagnostic dataset for compositional language and elementary visual reasoning. In *CVPR*, 2017.

[42] Kwang-Sung Jun and Francesco Orabona. Parameter-free online convex optimization with sub-exponential noise. In *Conference on Learning Theory*, pages 1802–1823. PMLR, 2019.

[43] Kaggle and EyePacs. Kaggle diabetic retinopathy detection, July 2015. URL https://www.kaggle.com/c/diabetic-retinopathy-detection/data.

[44] Hamed Karimi, Julie Nutini, and Mark Schmidt. Linear convergence of gradient and proximal-gradient methods under the polyak-łojasiewicz condition. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2016, Riva del Garda, Italy, September 19-23, 2016, Proceedings, Part I 16*, pages 795–811. Springer, 2016.

[45] Diederik P Kingma and Jimmy Ba. ADAM: A method for stochastic optimization. *International Conference on Learning Representations*, 2015.

[46] Bobby Kleinberg, Yuanzhi Li, and Yang Yuan. An alternative view: When does sgd escape local minima? In *International conference on machine learning*, pages 2698–2707. PMLR, 2018.

[47] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

[48] Hector J Levesque, Ernest Davis, and Leora Morgenstern. The Winograd schema challenge. In *AAAI Spring Symposium: Logical Formalizations of Commonsense Reasoning*, volume 46, page 47, 2011.

[49] Daniel Levy, Yair Carmon, John C Duchi, and Aaron Sidford. Large-scale methods for distributionally robust optimization. *Advances in Neural Information Processing Systems*, pages 8847–8860, 2020.

[50] Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, Joe Davison, Mario Šaško, Gunjan Chhablani, Bhavitvya Malik, Simon Brandeis, Teven Le Scao, Victor Sanh, Canwen Xu, Nicolas Patry, Angelina McMillan-Major, Philipp Schmid, Sylvain Gugger, Clément Delangue, Théo Matussière, Lysandre Debut, Stas Bekman, Pierric Cistac, Thibault Goehringer, Victor Mustar, François Lagunas, Alexander Rush, and Thomas Wolf. Datasets: A community library for natural language processing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 175–184, Online and Punta Cana, Dominican Republic, 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-demo.21. URL https://aclanthology.org/2021.emnlp-demo.21.

[51] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692, 2019.

[52] Zhuang Liu, Hanzi Mao, Chaozheng Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11966–11976, 2022.

[53] 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 *International Conference on Artificial Intelligence and Statistics*, pages 1306–1314, 2021.

[54] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2018.

[55] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2019.

[56] Olvi L Mangasarian. Pseudo-convex functions. In *Stochastic optimization models in finance*, pages 23–32. Elsevier, 1975.

[57] Loic Matthey, Irina Higgins, Demis Hassabis, and Alexander Lerchner. dsprites: Disentanglement testing sprites dataset. https://github.com/deepmind/dsprites-dataset/, 2017.

[58] Zakaria Mhammedi and Wouter M Koolen. Lipschitz and comparator-norm adaptivity in online learning. In *Conference on Learning Theory*, pages 2858–2887, 2020.

[59] Arkadi Nemirovski, Anatoli Juditsky, Guanghui Lan, and Alexander Shapiro. Robust stochastic approximation approach to stochastic programming. *SIAM Journal on optimization*, 19 (4):1574–1609, 2009.

[60] Arkadii Nemirovskii and David Borisovich Yudin. Problem Complexity and Method Efficiency in Optimization. 1983.

[61] Yurii Nesterov and Boris T Polyak. Cubic regularization of Newton method and its global performance. *Mathematical Programming*, 108(1):177–205, 2006.

[62] Yuval Netzer, Tao Wang, Adam Coates, A. Bissacco, Bo Wu, and A. Ng. Reading digits in natural images with unsupervised feature learning. 2011.

[63] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing*, pages 722–729. IEEE, 2008.

[64] Francesco Orabona and Dávid Pál. Coin betting and parameter-free online learning. *Advances in Neural Information Processing Systems*, 29:577–585, 2016.

[65] Francesco Orabona and Tatiana Tommasi. Training deep networks without learning rates through coin betting. *Advances in Neural Information Processing Systems*, 30:2160–2170, 2017.

[66] The pandas development team. pandas-dev/pandas: Pandas, 2020. URL https://doi.org/10.5281/zenodo.3509134.

[67] Courtney Paquette and Katya Scheinberg. A stochastic line search method with expected complexity analysis. *SIAM Journal on Optimization*, 30(1):349–376, 2020.

[68] Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and CV Jawahar. Cats and dogs. In *CVPR*, 2012.

[69] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. URL http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[70] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[71] Boris T. Polyak. *Introduction to Optimization.* Optimization Software, Inc, 1987.

[72] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763, 2021.

[73] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67, 2020.

[74] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas, 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1264. URL https://aclanthology.org/D16-1264.

[75] Sashank J Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of adam and beyond. In *International Conference on Learning Representations*, 2018.

[76] Michal Rolinek and Georg Martius. L4: Practical loss-based stepsize adaptation for deep learning. *Advances in Neural Information Processing Systems*, 2018.

[77] Tom Schaul, Sixin Zhang, and Yann LeCun. No more pesky learning rates. In *International Conference on Machine Learning*, pages 343–351, 2013.

[78] Ohad Shamir and Tong Zhang. Stochastic gradient descent for non-smooth optimization: Convergence results and optimal averaging schemes. In *International Conference on Machine Learning*, pages 71–79, 2013.

[79] Noam Shazeer and Mitchell Stern. Adafactor: Adaptive learning rates with sublinear memory cost. In *International Conference on Machine Learning*, pages 4596–4604, 2018.

[80] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[81] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA, 2013. Association for Computational Linguistics. URL https://aclanthology.org/D13-1170.

[82] Matthew Streeter and H Brendan McMahan. No-regret algorithms for unconstrained online convex optimization. *Advances in Neural Information Processing Systems*, pages 2402–2410, 2012.

[83] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[84] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

[85] Sharan Vaswani, Aaron Mishkin, Issam Laradji, Mark Schmidt, Gauthier Gidel, and Simon Lacoste-Julien. Painless stochastic gradient: Interpolation, line-search, and convergence rates. *Advances in Neural Information Processing Systems*, 2019.

[86] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020. doi: 10.1038/s41592-019-0686-2.

[87] Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. Superglue: A stickier benchmark for general-purpose language understanding systems. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 3261–3275, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/4496bf24afe7fab6f046bf4923da8de6-Abstract.html.

[88] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=rJ4km2R5t7.

[89] Rachel Ward, Xiaoxia Wu, and Leon Bottou. AdaGrad stepsizes: Sharp convergence over nonconvex landscapes. In *International Conference on Machine Learning*, pages 6677–6686, 2019.

[90] Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641, 2019. doi: 10.1162/tacl_a_00290. URL https://aclanthology.org/Q19-1040.

[91] Wes McKinney. Data Structures for Statistical Computing in Python. In Stéfan van der Walt and Jarrod Millman, editors, *Proceedings of the 9th Python in Science Conference*, pages 56 – 61, 2010. doi: 10.25080/Majora-92bf1922-00a.

[92] Ross Wightman. Pytorch image models. https://github.com/rwightman/pytorch-image-models, 2019.

[93] Adina Williams, Nikita Nangia, and Samuel Bowman. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana, 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1101. URL https://aclanthology.org/N18-1101.

[94] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-demos.6. URL https://aclanthology.org/2020.emnlp-demos.6.

[95] Mitchell Wortsman, Gabriel Ilharco, Samir Yitzhak Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, et al. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International Conference on Machine Learing*, 2022.

[96] Jianxiong Xiao, James Hays, Krista A Ehinger, Aude Oliva, and Antonio Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *CVPR*, pages 3485–3492, 2010.

[97] Jianxiong Xiao, Krista A Ehinger, James Hays, Antonio Torralba, and Aude Oliva. Sun database: Exploring a large collection of scene categories. *International Journal of Computer Vision*, 119(1):3–22, 2016.

[98] Yang You, Jing Li, Sashank Reddi, Jonathan Hseu, Sanjiv Kumar, Srinadh Bhojanapalli, Xiaodan Song, James Demmel, Kurt Keutzer, and Cho-Jui Hsieh. Large batch optimization for deep learning: Training BERT in 76 minutes. In *International Conference on Learning Representations*, 2020.

[99] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *British Machine Vision Conference*, 2016.

[100] Xiaohua Zhai, Joan Puigcerver, Alexander Kolesnikov, Pierre Ruyssen, Carlos Riquelme, Mario Lucic, Josip Djolonga, André Susano Pinto, Maxim Neumann, Alexey Dosovitskiy, Lucas Beyer, Olivier Bachem, Michael Tschannen, Marcin Michalski, Olivier Bousquet, Sylvain Gelly, and Neil Houlsby. A large-scale study of representation learning with the visual task adaptation benchmark. *arXiv: Computer Vision and Pattern Recognition*, 2019.

[101] Jiujia Zhang and Ashok Cutkosky. Parameter-free regret in high probability with heavy tails. In *Advances in Neural Information Processing Systems*, 2022.

[102] Yi Zhou, Junjie Yang, Huishuai Zhang, Yingbin Liang, and Vahid Tarokh. SGD converges to global minimum in deep learning via star-convex path. In *International Conference on Learning Representations*, 2019.

A Relaxing the convexity assumption

This section describes relaxations of convexity under which our main theoretical results still hold. In particular, our results naturally extend to star-convex functions [61] which satisfy

$$f(x) - f_\star \le \langle \nabla f(x), x - x_\star \rangle \quad \text{for all } x \in \mathcal{X}.$$

Our results also extend (with changed constants) to quasarconvex functions [36], which require that $f(x) - f_\star \le c \langle \nabla f(x), x - x_\star \rangle$ holds for some $c < \infty$ and all $x \in \mathcal{X}$. A further relaxation of star convexity requires it to hold only along the optimization trajectory:

Assumption 3. *[102, Definition 2] There exists $x_\star \in \arg\min_x f(x)$ and constant $\gamma \in (0, 1]$ such that the iterates of SGD satisfy*

$$f(x_k) - f_\star \le \langle \nabla f(x_k), x_k - x_\star \rangle \quad \textit{for all} \quad k$$

almost surely.

Zhou et al. [102] introduce this notion of a "star-convex path" and provide some empirical evidence that it may hold when training deep neural networks with SGD (see also Kleinberg et al. [46] for a related assumption). Zhou et al. [102] also prove that the assumption suffices to prove that SGD converges to the global minimizer; it suffices for DoG for similar reasons.

When substituting Assumption 1 with Assumption 3 our analysis goes through unchanged, except we can no longer use Jensen's inequality to argue directly about the suboptimality of the point $\bar{x}_\tau$. Instead, Theorem 1 with Assumption 3 says that, with probability at least $1 - \delta$,

$$\sum_{k=0}^{\tau-1} \omega_k (f(x_k) - f_\star) \le O\left(c_{\delta, r_\epsilon, T} \cdot \frac{d_0 \sqrt{G_\tau + L^2}}{T} \right),$$

with $\omega_k := \frac{\bar{r}_k}{\sum_{i=0}^{t-1} \bar{r}_i}$ and τ and $c_{\delta, r_\epsilon, T}$ as defined in Theorem 1. (Note that Assumption 3 implies $\sum_{k=0}^{t-1} \omega_k (f(x_k) - f_\star) \le \sum_{k=0}^{t-1} \omega_k \langle \nabla f(x_k), x_k - x_\star \rangle$ which replaces (4)).

We can turn the above bound into a constnat-probability guarantee for a specific T-DoG iterate x_K by sampling $K \sim \omega$ and using Markov's inequality:

$$\mathbb{P}\left(f(x_K) - f_\star \le e \sum_{k=0}^{\tau-1} \omega_k (f(x_k) - f_\star) \right) \le e^{-1}.$$

To obtain a high probability guarantee, we can make $l = \lceil \log \frac{1}{\delta} \rceil$ independent draws from ω, denoted $K_1, \ldots, K_l$ and use the fact that

$$\mathbb{P}\left(\min_{i \le l} f(x_{K_i}) - f_\star \le e \sum_{k=0}^{\tau-1} \omega_k (f(x_k) - f_\star) \right) \le \delta.$$

Finding the i that minimizes $f(x_{K_i})$ requires a logarithmic number of evaluations of the exact objective. When this is not feasible, we can instead consider a statistical learning setup where we have sample access to stochastic functions $F(x)$ such that $\mathbb{E} F(x) = f(x)$ for all x and, almost surely, F is $L_\star$ Lipschitz in a ball of radius $3d_0$ around x_0. (The stochastic subgradient oracle $\mathcal{G}(x)$ is then implemented by sampling F and returning its subgradient at x). We can then sample T new stochastic functions $F_1, \ldots, F_T$ and select $K^\star \in \arg\min_{k\in\{K_1,\ldots,K_l\}} \sum_{i=1}^T F_i(x_k)$. Straightforward application of Hoeffding's inequality shows that (when $\bar{r}_T \le 3d_0$)

$$f(x_{K^\star}) - f_\star \le \min_{i\le l} f(x_{K_i}) - f_\star + O\left(\frac{L_\star d_0}{\sqrt{T}}\sqrt{\log\frac{1}{\delta}}\right)$$

with probability at least $1-\delta$.

We remark that the literature contains a plethora of other convexity relaxations such as quasiconvexity [3], pseudoconvexity [56], Polyak-Łojasiewicz conditons [44] and weak convexity [21]. Exploring the convergence of DoG under these additional convexity relaxations is left to future work.

B Useful algebraic facts

B.1 Lemma 4

Lemma 4. *Let $a_0, \ldots, a_t$ be a nondecreasring sequence of nonnegative numbers. Then*

$$\sum_{k=1}^{t} \frac{a_k - a_{k-1}}{\sqrt{a_k}} \le 2(\sqrt{a_t} - \sqrt{a_0}).$$

Proof. We have

$$\sum_{k=1}^{t} \frac{a_k - a_{k-1}}{\sqrt{a_k}} = \sum_{k=1}^{t} \frac{(\sqrt{a_k} - \sqrt{a_{k-1}})(\sqrt{a_k} + \sqrt{a_{k-1}})}{\sqrt{a_k}} \le 2\sum_{k=1}^{t}\big(\sqrt{a_k} - \sqrt{a_{k-1}}\big) \le 2(\sqrt{a_t} - \sqrt{a_0}).$$

□

B.2 Lemma 5

Lemma 5. *Let $a_1, \ldots, a_T$ and $b_1, \ldots, b_T$ be sequences in $\mathbb{R}$ such that $\{a_i\}$ is nonnegative and nondecreasing. Then, for all $t \le T$,*

$$\left|\sum_{i=1}^{t} a_i b_i\right| \le 2a_t \max_{i\le t}\left|\sum_{i=1}^{t} b_i\right|.$$

Proof. Let $a_i' = a_i - a_{i-1}$ and $B_i = \sum_{j\le i} b_i$. Then (by discrete integration by parts)

$$\sum_{i=1}^{t} a_i b_i = \sum_{i=1}^{t} a_i\left(B_i - B_{i-1}\right) = \sum_{i=1}^{t-1}\left(a_i - a_{i+1}\right)B_i + a_t B_t = a_t B_t - \sum_{i=1}^{t-1} a_{i+1}' B_i.$$

Therefore

$$\left|\sum_{i=1}^{t} a_i b_i\right| \overset{(i)}{\le} |a_t B_t| + \left(\sum_{i=1}^{t-1}\left|a_{i+1}'\right|\right)\max_{i<t}|B_i| \le \left(|a_t| + \sum_{i=1}^{t-1}|a_{i+1} - a_i|\right)\max_{i\le t}|B_i| \overset{(ii)}{\le} 2a_t \max_{i\le t}|B_i|,$$

where we used (i) the triangle and Hölder's inequality, and (ii) that a_t is nonnegative and nondescreasing and therefore $\sum_{i=1}^{t-1}|a_{i+1} - a_i| = a_t - a_1 \le a_t$. □

B.3 Proof of Lemma 3

Proof. Define $K := \lceil \log(s_T/s_0) \rceil$, and $n := \lfloor \frac{T}{K} \rfloor$. Then, we have

$$\log\left(\frac{s_T}{s_0}\right) \geq \sum_{k=0}^{K-1} \log\left(\frac{s_{n(k+1)}}{s_{nk}}\right) \geq K \min_{k<K} \log\left(\frac{s_{n(k+1)}}{s_{nk}}\right).$$

Rearranging and using the definition of K gives

$$\min_{k<K} \log\left(\frac{s_{n(k+1)}}{s_{nk}}\right) \leq \frac{\log\left(\frac{s_T}{s_0}\right)}{K} \leq 1 \implies \min_{k<K} \frac{s_{n(k+1)}}{s_{nk}} \leq e.$$

Therefore,

$$\max_{t \leq T} \sum_{i \leq t} \frac{s_i}{s_t} \geq \max_{t \leq T} n \frac{s_{t-n}}{s_t} \geq \max_{k \leq K} n \frac{s_{n(k-1)}}{s_{nk}} \geq e^{-1} n \geq e^{-1} \frac{T}{\log(s_T/s_0)+1} - e^{-1}.$$

□

B.4 Lemma 6

Recall that $\log_+(z) := 1 + \log(t)$.

Lemma 6. *Let $a_{-1}, a_0, a_1, \ldots, a_t$ be a nondecreasing sequence of nonnegative numbers, then*

$$\sum_{k=0}^{t} \frac{a_k - a_{k-1}}{a_k \log_+^2(a_k/a_{-1})} \leq 1.$$

Proof. We have

$$\begin{aligned}\sum_{k=0}^{t} \frac{a_k - a_{k-1}}{a_k \log_+^2(a_k/a_{-1})} &\leq \sum_{k=0}^{t} \int_{a_{k-1}/a_0}^{a_k/a_{-1}} \frac{d\alpha}{\alpha \log_+^2(\alpha)} = \int_1^{a_t/a_{-1}} \frac{d\alpha}{\alpha \log_+^2(\alpha)} \\ &\leq \int_1^{\infty} \frac{d\alpha}{\alpha \log_+^2(\alpha)} = \left[\frac{1}{1+\log(\alpha)}\right]_1^{\infty} = 1.\end{aligned}$$

□

C Proofs for Section 3

C.1 Proof of Lemma 2

We begin by citing the following corollary of a general bound due to Howard et al. [37].

Corollary 2 (Carmon and Hinder [10, Corollary 1])**.** *Let X_t be adapted to $\mathcal{F}_t$ such that $|X_t| \leq 1$ with probability 1 for all t. Then, for every $\delta \in (0,1)$ and any $\hat{X}_t \in \mathcal{F}_{t-1}$ such that $|\hat{X}_t| \leq 1$ with probability 1,*

$$\mathbb{P}\left(\exists t < \infty : \left|\sum_{s=1}^{t} (X_s - \mathbb{E}[X_s \mid \mathcal{F}_{s-1}])\right| \geq \sqrt{A_t(\delta) \sum_{s=1}^{t} \left(X_s - \hat{X}_s\right)^2 + B_t(\delta)}\right) \leq \delta,$$

where $A_t(\delta) = 16 \log\left(\frac{60 \log(6t)}{\delta}\right)$ and $B_t(\delta) = 16 \log^2\left(\frac{60 \log(6t)}{\delta}\right)$.

Building on Corollary 2 we prove the following result, which allows for the sequence X_t to be almost-surely bounded by a random (rather than deterministic) quantity. (Recall that $\theta_{t,\delta} \coloneqq \log \frac{60\log(6t)}{\delta}$).

Corollary 3. *Let $C_t \in \mathcal{F}_{t-1}$ and let X_t be a martingale difference sequence adapted to $\mathcal{F}_t$ such that $|X_t| \le C_t$ with probability 1 for all t. Then, for all $\delta \in (0,1)$, $c > 0$, and $\hat{X}_t \in \mathcal{F}_{t-1}$ such that $|\hat{X}_t| \le C_t$ with probability 1,*

$$\mathbb{P}\left(\exists t \le T : \left|\sum_{s=1}^{t} X_s\right| \ge 4\sqrt{\theta_{t,\delta}\sum_{s=1}^{t}\left(X_s - \hat{X}_s\right)^2 + c^2\theta_{t,\delta}^2}\right) \le \delta + \mathbb{P}(\exists t \le T : C_t > c).$$

Proof. Define the random variables

$$W_t \coloneqq \frac{X_t}{\max\{c, C_t\}} \quad \text{and} \quad \hat{W}_t \coloneqq \frac{\hat{X}_t}{\max\{c, C_t\}}$$

and note that they satisfy the requirements of Corollary 2: W_t is a martingale difference sequence adapted to $\mathcal{F}_t$ while $\hat{W}_t \in \mathcal{F}_{t-1}$ and they both have absolute value bounded by 1 almost surely.

Next, define the events

$$E_T \coloneqq \left\{\exists t < T : \left|\sum_{s=1}^{t} X_s\right| \ge 4\sqrt{\theta_{t,\delta}\sum_{s=1}^{t}\left(X_s - \hat{X}_s\right)^2 + c^2\theta_{t,\delta}^2}\right\} \quad \text{and} \quad H_T \coloneqq \{\exists t \le T : C_t > c\}.$$

Then we have,

$$\mathbb{P}(E_T) = \mathbb{P}(E_T \cap \neg H_T) + \mathbb{P}(E_T \cap H_T) \le \mathbb{P}(E_T \cap \neg H_T) + \mathbb{P}(H_T).$$

Writing $\bar{C}_t = \max\{c, C_t\}$ for short, we have

$$\begin{aligned}
\mathbb{P}(E_T \cap \neg H_T) &= \mathbb{P}\left(\exists t \le T : \left|\sum_{s=1}^{t} \bar{C}_s W_s\right| \ge 4\sqrt{\theta_{t,\delta}\sum_{s=1}^{t} \bar{C}_s^2\left(W_s - \hat{W}_s\right)^2 + c^2\theta_{t,\delta}^2} \;\cap\; \neg H_T\right) \\
&\overset{(i)}{=} \mathbb{P}\left(\exists t \le T : \left|\sum_{s=1}^{t} W_s\right| \ge 4\sqrt{\theta_{t,\delta}\sum_{s=1}^{t}\left(W_s - \hat{W}_s\right)^2 + \theta_{t,\delta}^2} \;\cap\; \neg H_T\right) \\
&\le \mathbb{P}\left(\exists t \le T : \left|\sum_{s=1}^{t} W_s\right| \ge 4\sqrt{\theta_{t,\delta}\sum_{s=1}^{t}\left(W_s - \hat{W}_s\right)^2 + \theta_{t,\delta}^2}\right) \overset{(ii)}{\le} \delta,
\end{aligned}$$

where (i) uses the fact that $\bar{C}_s = c$ for all $s \le T$ when $\neg H_T$ holds, and (ii) uses Corollary 2 and $\theta_{t,\delta} \coloneqq \log \frac{60\log(6t)}{\delta}$. □

Next, we connect Corollary 3 with a handy algebraic fact (Lemma 5) to obtain the following result, which underpins Lemma 2.

Lemma 7. *Let S be the set of nonnegative and nondecreasing sequences. Let $C_t \in \mathcal{F}_{t-1}$ and let X_t be a martingale difference sequence adapted to $\mathcal{F}_t$ such that $|X_t| \le C_t$ with probability 1 for all t. Then, for all $\delta \in (0,1)$, $c > 0$, and $\hat{X}_t \in \mathcal{F}_{t-1}$ such that $|\hat{X}_t| \le C_t$ with probability 1,*

$$\mathbb{P}\left(\exists t \le T, \exists \{y_i\}_{i=1}^{\infty} \in S : \left|\sum_{i=1}^{t} y_i X_i\right| \ge 8y_t\sqrt{\theta_{t,\delta}\sum_{i=1}^{t}\left(X_i - \hat{X}_i\right)^2 + c^2\theta_{t,\delta}^2}\right) \le \delta + \mathbb{P}(\exists t \le T : C_t > c).$$

Proof. Follows from Lemma 5 (with y_i and X_i taking the roles of a_i and b_i, respectively), and Corollary 3 that bounds $\max_{i\le t}\left|\sum_{i\le t} X_i\right|$ for all $t \le T$. □

Proof of Lemma 2. For $k \in [T]$ define the random variables:

$$Y_k = \bar{r}_k \bar{d}_k, \quad X_k = \left\langle \Delta_k, \frac{x_k - x_\star}{\bar{d}_k} \right\rangle, \quad \text{and} \quad \hat{X}_k = -\left\langle \nabla f(x_k), \frac{x_k - x_\star}{\bar{d}_k} \right\rangle.$$

From these definitions we get

$$\sum_{k=0}^{t-1} Y_k X_k = \sum_{k=0}^{t-1} \bar{r}_k \left\langle \Delta_k, x_k - x_\star \right\rangle.$$

Therefore,

$$\mathbb{P}\left(\exists t \le T : \left| \sum_{k=0}^{t-1} \bar{r}_k \left\langle \Delta_k, x_k - x_\star \right\rangle \right| \ge 8 \bar{r}_{t-1} \bar{d}_{t-1} \sqrt{\theta_{t,\delta} G_{t-1} + L^2 \theta_{t,\delta}^2} \right)$$
$$\le \mathbb{P}\left(\exists t \le T : \left| \sum_{k=0}^{t-1} Y_k X_k \right| \ge 8 Y_t \sqrt{\theta_{t,\delta} \sum_{k=0}^{t-1} \left(X_k - \hat{X}_k \right)^2 + L^2 \theta_{t,\delta}^2} \right) \le \delta + \mathbb{P}\big(\bar{\ell}_T > L\big)$$

where the last inequality uses Lemma 7. □

C.2 Proof of Corollary 1

Proof. If $T > 2\log_+(\bar{r}_T / r_\epsilon)$ then the corollary follows by Proposition 1 and Lemma 3 with $s_t = \bar{r}_t$. For the corner case when $T \le 2\log_+(\bar{r}_T / r_\epsilon)$ we use that $f(\bar{x}_\tau) - f_\star \le O(L\bar{d}_\tau) \le O(L(\bar{r}_\tau + d_0))$ where the first inequality uses (4), Cauchy-Schwarz and that $\|\nabla f(x_t)\| \le L$; the second inequality uses the triangle inequality. □

C.3 DoG can diverge on a pathological instance

Consider the following variant of Nemirovski's function [60] defined on $\mathbb{R}^m$:

$$f(x) = \max_{i \le m} \max \left\{ [x]_i, -\frac{1}{\sqrt{m}} [x]_i \right\},$$

where $[x]_i$ denotes the i'th coordinate of x and $[x_0]_i = 10 r_\epsilon / \sqrt{m}$ for all i, so that $d_0 = 10 r_\epsilon > r_\epsilon$. We show that applying DoG on this function gives $\bar{r}_T / d_0 = \sqrt{T}/10$ for all $T \le m$, meaning that the ratio $\bar{r}_T / d_0$ can be made arbitrarily large by increasing T and m.

Define

$$i(x) \coloneqq \min \arg\max_{i \le m} \left\{ [x]_i, -\frac{[x]_i}{\sqrt{m}} \right\},$$

i.e., $i(x)$ is the smallest coordinate which is candidate for providing a subgradient. Using this notation, a valid subgradient for f is:

$$\nabla f(x) = \begin{cases} e_{i(x)} & x_{i(x)} > 0 \\ -\frac{1}{\sqrt{m}} e_{i(x)} & \text{otherwise} \end{cases}$$

where e_j is a vector with one in the jth entry and zero elsewhere. With this subgradient choice for $k \le m$ the iterates become:

$$[x_k]_j = \begin{cases} 10r_\epsilon/\sqrt{m} - r_\epsilon & j < k \\ 10r_\epsilon/\sqrt{m} & j \ge k \end{cases} \tag{7}$$

and therefore $\bar{r}_k = \sqrt{k}r_\epsilon = \sqrt{k}d_0/10$ as claimed. We confirm (7) by induction. Since $[x_0]_i = 10r_\epsilon/\sqrt{m}$ for all i, the expression (7) holds for $k = 0$. If (7) holds for all $k \le n < m$ then

$$\nabla f(x_k) = e_k$$

and therefore $G_n = n$ so that $\eta_n = \frac{r_\epsilon\sqrt{n}}{\sqrt{n}} = r_\epsilon$ and $x_{n+1} = x_n - \frac{\sqrt{n}}{\sqrt{n}}r_\epsilon e_n$, meaning that

$$[x_{n+1}]_j = \begin{cases} 10r_\epsilon/\sqrt{m} - r_\epsilon & j < n+1 \\ 10r_\epsilon/\sqrt{m} & j \ge n+1 \end{cases}$$

which completes the induction.

C.4 Proof of Proposition 2

To show iterate boundedness in the stochastic setting we define the stopping time

$$\tau = \min\{t : \bar{r}_t > 3d_0\},$$

so that the event $\{\bar{r}_T \le 3d_0\}$ is the same as $\{\tau > T\}$. Let η_k denote the sequence of T-DoG step sizes (for given L, T and δ). To facilitate our analysis we also define the following truncated step size sequence:

$$\tilde{\eta}_k \coloneqq \begin{cases} \eta_k & k < \tau \\ 0 & \text{otherwise.} \end{cases} \tag{8}$$

Truncating the step size allows us to rigorously handle the possibility that $\bar{r}_T$ exceeds $3d_0$ and we no longer have any bound on the magnitude of the stochastic gradients. In particular, the following holds for $\{\tilde{\eta}_k\}$ but not for $\{\eta_k\}$. (Recall that $\Delta_t \coloneqq g_t - \nabla f(x_k)$).

Lemma 8. *Under Assumption 2, if $L \ge L_\star$ then the truncated* T-DoG *step sizes (8) satisfy, for all $t \le T$,*

$$\tilde{\eta}_t \in \sigma(g_0, \ldots, g_{t-1}) \ , \tag{9}$$

$$|\tilde{\eta}_t \langle \gamma, x_t - x_\star \rangle| \le \frac{6d_0^2}{8^2\theta_{T,\delta}} \ \textit{for}\ \gamma \in \{g_t, \nabla f(x_t), \Delta_t\} \ , \tag{10}$$

$$\sum_{k=0}^{t} \tilde{\eta}_k^2 \|g_k\|^2 \le \frac{9d_0^2}{8^4\theta_{T,\delta}} \ , \ \textit{and} \tag{11}$$

$$\sum_{k=0}^{t} (\tilde{\eta}_k \langle g_k, x_k - x_\star \rangle)^2 \le \frac{12^2 d_0^4}{8^4\theta_{T,\delta}} \tag{12}$$

Proof. To see that (9) holds, note that $g_0, \ldots, g_{t-1}$ determine $x_0, \ldots, x_t$ and therefore they determine whether $\tau > t$. If $\tau > t$ holds, then $\bar{r}_t \le 3d_0$ and therefore $L \ge \|g_t\|$, meaning that η_t is a function of $g_0, \ldots, g_{t-1}$ (through G_{t-1}) and not g_t. Alternatively, if $\tau \le t$ then, $\tilde{\eta}_t = 0$ and again not a function of g_t.

To see the bound (10), first note that either $\tilde{\eta}_t = 0$ (and the bound holds trivially) or $\bar{r}_t \leq 3d_0$ and hence $\ell(x_t) \leq L_\star \leq L$ implying that $\|\Delta_k\| \leq \|g_k\| + \|\nabla f(x_k)\| \leq 2\ell(x_t) \leq 2L$. Since $G'_t \geq 4^2 8^4 L^2 \theta_{T,\delta}^2$ for all t, the Cauchy-Schwartz inequality gives

$$|\tilde{\eta}_t \langle \Delta_t, x_t - x_\star \rangle| \leq \frac{\bar{r}_t}{\sqrt{G'_t}} \|\Delta_t\| d_t \leq \frac{1}{2 \cdot 8^2 \theta_{T,\delta}} \bar{r}_T d_t \leq \frac{6d_0^2}{8^2 \theta_{T,\delta}},$$

where the last inequality uses again $\bar{r}_t \leq 3d_0$ and $d_t \leq d_0 + \bar{r}_t$ by the triangle inequality. Bounds for $|\tilde{\eta}_t \langle \gamma, x_t - x_\star \rangle|$ for $\gamma \in \{g_t, \nabla f(x_t)\}$ follow by the same argument.

To establish (11), first note that $\sum_{k=0}^{t} \tilde{\eta}_k^2 \|g_k\|^2 \leq \sum_{k=0}^{\tau-1} \eta_k^2 \|g_k\|^2$ by the definition of $\tilde{\eta}_k$. Furthermore

$$\sum_{k=0}^{\tau-1} \eta_k^2 \|g_k\|^2 = \sum_{k=0}^{\tau-1} \frac{\bar{r}_k^2 \|g_k\|^2}{G'_k} \overset{(i)}{\leq} \frac{\bar{r}_{\tau-1}^2}{8^4 \theta_{T,\delta}} \sum_{k=0}^{\tau-1} \frac{G_k - G_{k-1}}{(G_k + L^2) \log_+^2 \frac{G_k + L^2}{L^2}} \overset{(ii)}{\leq} \frac{9d_0^2}{8^4 \theta_{T,\delta}},$$

where (i) uses that $\|g_k\|^2 = G_k - G_{k-1}$ (with the shorthand $G_{-1} \coloneqq 0$) and

$$G'_k \geq 8^4 \theta_{T,\delta} (G_k + L^2) \log_+^2 (k+1) \geq 8^4 \theta_{T,\delta} (G_k + L^2) \log_+^2 \frac{G_k + L^2}{L^2};$$

for all $k < \tau$, since $\bar{r}_k \leq 3d_0$ and hence $\|g_k\| \leq L_\star \leq L$ and $G_k \leq kL^2$, while (ii) uses Lemma 6 with $a_k = G_k + L^2$ and $\bar{r}_{\tau-1} \leq 3d_0$.

The final bound (12) follows immediately from (11) by noting that

$$\sum_{k=0}^{t} (\tilde{\eta}_k \langle g_k, x_k - x_\star \rangle)^2 \leq \sum_{k=0}^{t} \tilde{\eta}_k^2 \|g_k\|^2 d_k^2 \leq (4d_0)^2 \sum_{k=0}^{t} \tilde{\eta}_k^2 \|g_k\|^2,$$

where the first inequality follows from Cauchy-Schwartz and the second inequality from the fact that only terms with $k < \tau$ contribute to the sum. □

The above properties allow us to establish the following concentration bound.

Lemma 9. *In settings of Lemma 8,*

$$\mathbb{P}\left(\exists t \leq T : \sum_{k=0}^{t-1} \tilde{\eta}_k \langle \Delta_k, x_\star - x_k \rangle > d_0^2 \right) \leq \delta.$$

Proof. Consider the filtration $\mathcal{F}_t = \sigma(g_0, \ldots, g_t)$ and define $X_t = \tilde{\eta}_t \langle \Delta_t, x_\star - x_t \rangle$ and $\hat{X}_t = -\tilde{\eta}_t \langle \nabla f(x_t), x_\star - x_t \rangle$. Then, by (9) we have that X_t is martingale difference sequence adapted to $\mathcal{F}_t$ and $\hat{X}_t \in \mathcal{F}_{t-1}$. Moreover, by (10) we have that $\max\{|X_t|, |\hat{X}_t|\} \leq c$ almost surely for $c = \frac{24d_0^2}{8^4 \theta_{T,\delta}}$. Substituting into Corollary 3 (and shifting the start of the summation from 1 to 0) we have

$$\mathbb{P}\left(\exists t \leq T : \left| \sum_{k=0}^{t-1} X_k \right| \geq 4 \sqrt{\theta_{t,\delta} \sum_{k=0}^{t-1} \left(X_k - \hat{X}_k \right)^2 + c^2 \theta_{t,\delta}^2} \right) \leq \delta.$$

Noting that $X_t - \hat{X}_t = \langle g_t, x_\star - x_t \rangle$ and substituting the definition of c and the bound (12) gives, for every $t < T$,

$$4 \sqrt{\theta_{t,\delta} \sum_{k=0}^{t-1} \left(X_k - \hat{X}_k \right)^2 + c^2 \theta_{t,\delta}^2} \leq 4 \sqrt{\theta_{t,\delta} \frac{12^2 d_0^4}{8^4 \theta_{T,\delta}} + \left(\frac{6\theta_{t,\delta} d_0^2}{8^2 \theta_{T,\delta}} \right)^2} \leq d_0^2,$$

concluding the proof of lemma. □

Finally, we show that the event defined in Lemma 9 implies the desired distance bound.

Lemma 10. *In the setting of Proposition 2, if* $\sum_{k=0}^{t-1} \tilde{\eta}_k \langle \Delta_k, x_\star - x_k \rangle \le d_0^2$ *for all* $t \le T$ *then* $\tau > T$*, i.e.,* $\bar{r}_T \le 3d_0$*.*

Proof. To condense notation, let $B_t \coloneqq \max_{t' \le t} \sum_{k=0}^{t'-1} \tilde{\eta}_k \langle \Delta_k, x_\star - x_k \rangle$, so that the claim becomes $B_t \le d_0^2$ implies $\tau > t$ for all $t \le T$. We prove the claim by induction on t. The basis of the induction is that $\tau > 0$ always holds since $\bar{r}_0 = r_\epsilon \le 3d_0$ by assumption. For the induction step, we assume that B_{t-1} implies $\tau \ge t$ and show that $B_t \le d_0^2$ implies $\tau > t$. To that end, we use $\langle f(x_t), x_t - x_\star \rangle \ge f(x_t) - f_\star \ge 0$ to rearrange (6) and obtain

$$d_{k+1}^2 - d_k^2 \le \eta_k^2 \|g_k\|^2 + 2\eta_k \langle \Delta_k, x_\star - x_k \rangle$$

for all k. Summing this inequality from $k = 0$ to $k = t - 1$, we get

$$d_t^2 - d_0^2 \le \sum_{k=0}^{t-1} \eta_k^2 \|g_k\|^2 + 2 \sum_{k=0}^{t-1} \eta_k \langle \Delta_k, x_\star - x_k \rangle = \sum_{k=0}^{t-1} \tilde{\eta}_k^2 \|g_k\|^2 + 2 \sum_{k=0}^{t-1} \tilde{\eta}_k \langle \Delta_k, x_k - x_\star \rangle ,$$

where the equality holds since $\tau > t - 1$ and therefore $\eta_k = \tilde{\eta}_k$ for all $k \le t - 1$. Now, by the bound (11) we have $\sum_{k=0}^{t-1} \tilde{\eta}_k^2 \|g_k\|^2 \le \frac{9^2}{8^4 \theta_{T,\delta}} d_0^2 \le d_0^2$. Moreover, $\sum_{k=0}^{t-1} \tilde{\eta}_k \langle \Delta_k, x_k - x_\star \rangle \le B_t \le d_0^2$ by definition and assumption, from which we conclude that $d_t^2 \le 4d_0^2$ and hence $r_t \le d_0 + d_t \le 3d_0$. Since $\bar{r}_t = \max\{\bar{r}_{t-1}, r_t\}$ and $\bar{r}_{t-1} \le 3d_0$ by the induction assumption, we have that $\bar{r}_t \le 3d_0$ as well, concluding the proof. □

Proposition 2 follows immediately from Lemmas 9 and 10.

D Experiment details

D.1 Environment settings

All experiments were based on PyTorch [69] (version 1.12.0).

Language experiments were done with the *transformers* [94] library (version 4.21.0) and tracked using the *Comet.ML* [17]. All datasets were provided by the *Datasets* library [50] (version 2.4.0) and were left as is, including train-eval-test splits.

Vision experiments were based on the *pytorch-image-models* (`timm`, version0.7.0dev0) repository [92], with *TensorFlow datasets* (version 4.6.0) as a dataset backend [1].

To support the training and analysis of the results, we used *numpy* [33], *scipy* [86], *pandas* [91, 66] and *scikit-learn* [70].

D.2 Implementation details

Whenever possible, we used existing scripts and recipes provided by `timm` and *transformers* to fine-tune the models. We implemented DoG, L-DoG and the polynomial model averaging as a subclass of PyTorch *Optimizer* interface. We provide implementation of both in `https://github.com/formll/dog`.

Task	Batch size	Steps	Metric	LR warmup	LR annealing	Grad. clipping
VTAB datasets	128	20K	Accuracy	None	Cosine	None
SQuAD	48	5475	F_1	10%	Cosine	1
SST-2	32	31407	Accuracy	10%	Cosine	1
CoLA	32	1000	Matthews correlation	10%	Cosine	1
MRPC	32	1734	F_1	10%	Cosine	1
STSB	32	3281	Pearson correlation	10%	Cosine	1
QNLI	32	49218	Accuracy	10%	Cosine	1
RTE	32	10000	Accuracy	10%	Cosine	1
QQP	32	160625	F_1	10%	Cosine	1
MNLI	32	184218	Accuracy	10%	Cosine	1

Table 3: Configuration used for each dataset in our testbed (Section 4.1). For all language tasks, we used the batch size as in Liu et al. [51], and at least 150% the number of steps used there, in order to ensure convergence. Learning rate (LR) warmup and annealing refers to tuning with SGD and Adam. In all cases, both DoG and L-DoG used neither warmup nor annealing.

D.3 Datasets

The datasets used in the language experiments are: **CoLA** [90], **SST-2** [81], **MRPC** [25], **QQP** [39], **STS-B** [12], **MNLI** [93], **QNLI** [74], and **RTE** [20, 5, 30, 7]. Following Liu et al. [51], we discard WNLI [48] as it was found to be ill-defined and was reformulated differently in SuperGLUE [87].

The datasets used in the vision experiments are: The tasks are **Caltech101** [29], **CIFAR-100** [47], **CLEVR-Dist** [41], **DMLab** [6], **dSprites-Ori** [57], **DTD** [16], **Flowers102** [63], **Pets** [68], **Resisc45** [15], **Retinopathy** [43], **Sun397** [96, 97], and **SVHN** [62].

D.4 Models

When fine-tuning RoBERTa (from the ‘roberta-base’ checkpoint) on classification tasks, we follow the common technique of prepending an *CLS* token to the input, and feeding its final representation to a one hidden-layer, randomly initialized MLP that is used as a classification head. For SQuAD, the classification head is tasked with multi-label classification, predicting the probability that each word (token) in the input is the beginning/end of the answer span, and we then used the span that has the maximum likelihood as the model’s output. When fine-tuning T5 (from the ‘t5-base’ checkpoint), we treated all tasks as sequence-to-sequence tasks, translating classification labels to appropriate words (e.g. 0/1 to positive/negative) and then evaluated accuracy with exact match. The computer vision pre-trained models were accessed via `timm`, and had randomly initialized classification heads. The strings used to load the models were: ‘convext_tiny’, ‘resnet50’, ‘densenet121’, ‘vit_base_patch32_224_in21k’ and ‘vgg11’.

D.5 Hyper-parameters

We trained each model/tasks combination a fixed number of steps (see Table 3), performing evaluation every 500 update steps (except for the smaller datasets Caltech101, DTD, Flowers102 and Pets where we evaluated every 200) with both the latest checkpoint, and the polynomial averaged one (see below). We did not use any weight decay. For language models, we left dropout at its default value in the transformers library. We used batch sizes as is common practice for each task, as detailed in Table 3.

Data augmentation in vision experiments. The VTAB suite [100] divides its datasets into three categories: natural, specialized and structured, and we uses a suitable data augmentation strategy for each of the categories. In particular, for structured datasets we simply resizes the images to a (224, 224) resolution, while for the natural and specialized datasets we uses the standard "inception crop" [83] at training time and a 0.875 center crop at test time. For natural datasets we additionally applied a color jitter operation with parameter 0.4 (as implemented in `timm`). Finally, we applied a random horizontal flip for all datasets except SVHN and dSprites-Ori, where such augmentation clearly interferes with the task.

Model selection in vision experiments. For computer vision experiments, we used the VTAB evaluation splits to select the best checkpoint, and then reported performance on the training split. Unlike the experiments accompanying the VTAB suite [100], we did not retrain selected models on the combination of training and validation data.

Repeated runs. To account for randomness, we repeated our fine-tuning experiments using multiple seeds. In most cases (with exceptions listed below) we repeated each DoG and L-DoG training 5 times. For SGD and Adam repeating the learning with all learning rates was computationally prohibitive, so instead for each task / model pair we repeated 5 times only the best-performing LR (i.e., instance-tuned LR) and the best-performing LR across all tasks for that model (i.e., model-tuned LR) according the validation split. A few experiments were too computationally expensive to repeat: for QQP and MNLI (which require a large step budget) we have only 1–3 repetitions per training configuration, and for ConvNeXt-T (which takes a long time per step) we did repeat the training runs.

Each relative error difference (RED) score combines the error of two optimization methods (one being DoG) on a particular model task combination. Given multiple seeds for each optimization method, we computed the RED scores for each possible seed combination. In Figures 2, 3, 5 and 6 (which aggregate multiple tasks) we average over those multiple RED values and compute the statistics of the average RED. In per-task breakdowns such as Figure 7, **??**, and tables 4 and 5 we report the statistics over the multiple RED values.

Baseline optimizers. For both SGD and Adam, we used cosine learning rate decay, and searched over appropriate values for peak learning rate The base learning rate search space used when performing fine-tuning for each model/task combination can be found in Tables 4 and 5. We did not use momentum for SGD. For Adam we used $\beta_1 = 0.9$ for all experiments, and $\beta_2 = 0.999$ for language experiments and $\beta_2 = 0.99$ for vision experiments. For language models only, we used warmup of 10% of the maximum steps count, and gradient clipping at global norm 1. We did not perform learning warmup or gradient clipping for the vision experiment since we did not encounter any training stability issues there.

Setting r_ϵ. As explained in Section 4.1, setting $r_\epsilon \coloneqq \alpha(1 + \|x_0\|)$ generally works well for $\alpha = 10^{-4}$. However, in some cases such as with T5, $\|x_0\|$ can be very large, causing destructively large first updates, with η_t increasing exponentially and the model diverging. This is easily detectable early during training, as usually η_t exceeds 1000 within the first 100 steps. Since the theory requires r_ϵ to be small, we simply decreased α by a factor of 100. While preliminary experimetns with RoBERTa indicated that DoG also performed well with $\alpha = 10^{-4}$, for the sake of consistency we use the same values in all models of the same domain. Thus, models fine-tuned on vision tasks used $\alpha = 10^{-4}$, while language models used 10^{-6} for DoG and 10^{-8} for L-DoG.

Model averaging. As mentioned in Section 4.1, we used the polynomial decay averaging as proposed by Shamir and Zhang [78]. Namely, we kept an additional copy of the model weights, and in every update step we updated our running average of the model parameters as follows:

$$\bar{x}_t^\gamma = \left(1 - \frac{1+\gamma}{t+\gamma}\right)\bar{x}_{t-1}^\gamma + \frac{1+\gamma}{t+\gamma}x_t \tag{13}$$

The vector $\bar{x}_t^\gamma$ roughly corresponds to an average of the last t/γ iterates preceding iteration t. For all models, we set $\gamma = 8$. We did not perform any tuning of the parameter γ; we chose the value 8 because 1/8 seemed like a good fraction of iterates to average, and because it worked well in the experiments of [49].

To ensure that iterate averaging is never harmful, for each optimization method we selected the best-performing checkpoint across both x_t and $\bar{x}_t^\gamma$ (i.e., with or without averaging).

D.6 Figure 1 details

We generated Figure 1 as part of our fine-tuning testbed. In particular, SGD used a cosine learning rate annealing (without warmup), both algorithms use polynomial decay averaging, and we report test performance on the best checkpoint selected on a validation set.

D.7 Fine-tuning ImageNet

Our training setup mostly followed the default configuration in Wortsman et al. [95]. In particular, we used batch size 512 and the default `timm` augmentation (as in our main computer vision experiments) which Wortsman et al. [95] refer to as 'medium aug.' We trained for 25K steps, corresponding to roughly 10 passes over the data. However (keeping with our computer vision testbed setting) we did not perform learning rate warmup or gradient clipping, and we initialized the classification head to be random.

For AdamW [54] we used weight decay 0.1 and cosine learning rate annealing as in Wortsman et al. [95]. We obtained accuracies within 0.5% of the numbers reported in Appendix L of Wortsman et al. [95].

DoG and L-DoG we used weight decay 0 since the value 0.1 is meant for decoupled weight decay and we did not wish to re-tune a weight decay parameter. We set r_ϵ to be $10^{-6} \cdot (1 + \|x_0\|)$ without trying different values of this parameter.

For SGD we used cosine learning rate annealing and set weight decay to 0 for a more direct comparison to DoG.

D.8 Training from scratch

Our training setup followed the basic training configuration of Cubuk et al. [18], which is typical for training ResNets on CIFAR-10: data augmentations comprising a random crop after 4 pixel padding and random horizontal flip, batch size of 128, weight decay of 0.0005 and 200 epochs of training. SGD used cosine learning weight annealing and (when applicable) Nesterov momentum. We did not use dropout or any other additional form of regularization. For DoG and L-DoG, we set $r_\epsilon = 10^{-4} \cdot (1 + \|x_0\|)$ without trying different values of this parameter. The accuracies we obtained using SGD and DoG are consistent (and slightly better) than the baseline number reported in Table 2 of Cubuk et al. [18] and within 0.1% of the one reported in Table 1 of Carmon et al. [11].

Model	**Optimizer**	**LR**	CoLA	MNLI	MRPC	QNLI	QQP	RTE	SQuAD	SST-2	STS-B	Avg.
RoBERTa-b	**Adam**	**5e-06**	60.8	87.8	89.8	93.0	88.7	77.6	90.3	95.1 (0.34)	90.4	85.46
		1e-05	63.4	87.9 (0.05)	90.5	93.1 (0.22)	89.0 (0.11)	77.6	91.5	95.1	90.8	86.22
		3e-05	63.5 (1.33)	87.3	92.3 (0.44)	92.9 (0.15)	88.8	80.6 (1.19)	92.3 (0.05)	94.8 (0.16)	91.1 (0.19)	86.94
		5e-05	61.8	86.8	92.0	92.3	88.0	78.7	92.4 (0.07)	94.3	90.9	85.93
		0.0001	57.5	86.2	91.8	91.3	0.0	79.4	91.9	94.7	89.8	75.33
	SGD	**0.003**	56.3	86.4	81.9	91.5	85.1	75.5	79.4	93.6	87.0	81.44
		0.01	59.1	87.4	89.7	92.5	87.0	79.0 (0.69)	86.2	94.8	90.3 (0.25)	84.81
		0.03	62.3 (1.38)	87.8 (0.04)	91.8 (0.21)	92.7 (0.18)	88.3	78.9 (0.79)	89.5 (0.12)	95.0 (0.15)	90.7 (0.12)	86.30
		0.1	58.7	87.4	91.0	92.2	88.7 (0.06)	78.3	91.0	94.0	90.4	85.52
		0.3	0.0	86.0	81.2	85.3	87.7	64.6	91.3 (0.11)	92.8	27.0	68.14
		1.0	0.0	81.5	81.2	83.8	79.4	52.7	82.8	89.7	13.0	62.22
	DoG	**-**	62.8 (1.17)	87.7 (0.12)	91.6 (0.29)	92.6 (0.15)	88.2 (0.02)	78.5 (2.91)	91.3 (0.17)	94.9 (0.26)	90.5 (0.33)	86.46
	L-DoG	**-**	63.3 (0.32)	87.7 (0.12)	91.5 (0.19)	92.8 (0.28)	88.7 (0.14)	80.1 (1.00)	91.8 (0.18)	94.8 (0.54)	90.6 (0.34)	86.81
T5-b	**Adam**	**5e-06**	53.4 (0.93)	86.8	91.4	93.4	88.0	79.8	90.3	93.9	90.4	84.82
		1e-05	56.0 (0.63)	86.9	91.2	93.5	88.2	80.5	90.4	94.2	90.6	85.33
		3e-05	58.9 (1.10)	87.1	91.6	93.4 (0.16)	88.8	82.3	90.8	94.8 (0.24)	90.7	86.12
		5e-05	58.9 (0.80)	87.3	91.8	93.3	89.0	80.9	90.7	94.8	90.8 (0.10)	85.97
		0.0001	58.3 (0.80)	86.9	92.9 (0.35)	93.5 (0.10)	89.2	82.5 (0.48)	90.9 (0.15)	94.9 (0.26)	90.8 (0.18)	86.53
		0.0005	55.4 (0.45)	86.1	92.3	92.7	88.8	81.2 (1.48)	90.0	94.6	89.7	85.29
	SGD	**0.003**	22.9 (1.64)	85.8	90.1	92.7	87.5	66.8	90.3	92.1	90.3	79.43
		0.01	49.4 (0.27)	86.4	92.2	93.1	87.4	80.9	90.3	93.0	90.5	84.49
		0.03	56.4 (0.52)	86.5	92.0	93.2 (0.03)	88.1	80.9	90.5	93.6	90.6	85.40
		0.1	58.9 (0.82)	86.8	91.8	93.1	88.7	84.1	90.7 (0.04)	93.7	90.6 (0.09)	86.13
		0.3	56.7 (0.83)	86.1	92.8 (0.47)	93.0 (0.13)	88.7	82.8 (1.24)	90.7 (0.05)	93.9 (0.15)	90.7 (0.21)	86.07
		1.0	0.0 (0.00)	32.8	81.2	91.7	56.9	78.7	90.1	92.1	88.7	67.56
	DoG	**-**	7.3 (6.78)	86.9 (0.21)	92.8 (0.35)	93.1 (0.09)	88.5	81.7 (3.06)	90.6 (0.05)	94.1 (0.19)	90.7 (0.09)	80.58
	L-DoG	**-**	59.9 (1.43)	87.3 (0.10)	91.9 (0.32)	93.6 (0.02)	87.8	83.1 (0.78)	90.3 (0.02)	95.0 (0.19)	90.5 (0.05)	86.51

Table 4: Average (std) performance of RoBERTa-b and T5-b on language tasks, when fine-tuned with different optimization algorithms and their respective base learning rate (when applicable). DoG uses $r_\epsilon = 10^{-6}(1 + \|x_0\|)$ and L-DoG uses $r_\epsilon = 10^{-8}(1 + \|x_0\|)$. Scores are reported as mean across seeds, measured in the corresponding performance metric as detailed in Table 3.

E Additional experiment results

E.1 Full breakdown of main experiment results

Figure 7 as well as Tables 4 and 5 provide the full breakdown of our main fine-tuning results, comparing DoG and L-DoG to SGD and Adam with different learning rates for each model/task combination.

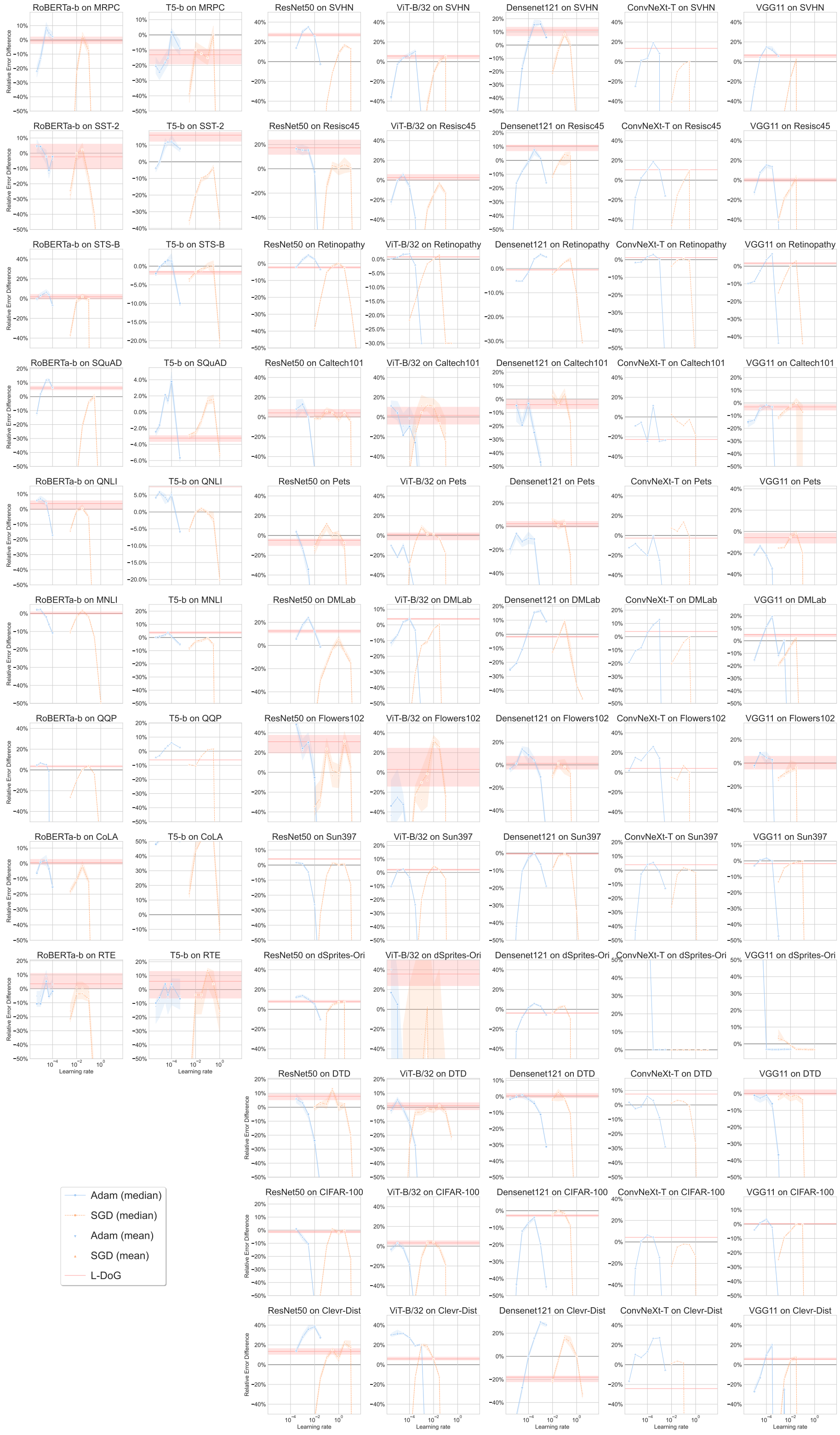


Figure 7: Relative error difference (RED) statistics across seeds (median, mean and IQR shown as shaded region) for all model/task combinations. The red horizontal line shows the median RED of L-DoG.

Model	Optimizer	LR	Caltech101	CIFAR-100	Clevr-Dist	DMLab	dSprites-Ori	DTD	Flowers102	Pets	Resisc45	Retinopathy	Sun397	SVHN	Avg.
ConvNeXt-T	**Adam**	**3e-06**	-	70.0	89.2	70.5	86.5	72.1	92.5	93.1	91.2	-	36.7	-	-
		1e-05	89.0	84.7	91.7	72.7	95.7	70.9	93.5	93.3	94.7	83.2	64.2	96.6	85.33
		3e-05	89.4	87.8	91.4	73.2	96.3	71.3	93.3	92.9	95.6	83.2	74.3	97.3	86.75
		0.0001	87.5	88.5	91.9	76.0	96.4	73.3	93.9	92.6	95.9	83.7	76.0	97.4	87.25
		0.0003	91.1	88.2	93.2	77.5	7.6	72.5	94.3	93.8	96.3	83.9	76.3	97.8	80.58
		0.001	87.5	85.9	93.2	78.5	7.6	69.1	93.4	92.3	95.9	83.5	74.7	97.5	79.42
		0.003	87.6	73.7	90.2	22.2	7.6	63.4	87.9	88.3	94.7	73.6	71.7	19.6	64.50
	SGD	**0.01**	90.2	85.0	90.8	70.3	7.6	72.0	92.0	94.3	93.4	82.9	68.4	96.2	78.25
		0.03	89.5	87.2	91.1	72.2	7.6	72.6	91.8	94.1	94.8	83.4	74.2	97.0	79.25
		0.1	89.1	87.5	90.9	74.3	7.6	72.3	92.9	94.7	95.4	83.6	75.4	97.2	79.58
		0.3	89.7	87.4	24.5	75.2	7.6	71.4	92.4	93.8	95.9	83.4	75.1	97.3	74.00
		1.0	88.1	86.1	24.5	22.2	7.6	64.0	0.4	13.4	2.1	73.6	74.5	19.6	39.33
		3.0	0.4	1.0	20.0	22.2	7.4	2.1	0.3	2.7	2.2	73.6	0.5	6.7	11.25
		10.0	-	1.0	20.0	22.2	7.4	2.1	0.3	2.7	2.2	-	0.5	-	-
	DoG	-	89.9	87.7	90.7	75.3	7.6	71.6	92.4	93.8	95.5	83.5	75.0	97.3	79.50
	L-DoG	-	87.7	88.2	88.5	76.3	96.4	73.8	92.7	93.7	95.9	83.7	75.9	97.7	86.92
Densenet121	**Adam**	**3e-06**	-	62.3	84.3	65.0	84.8	65.9	88.1	88.4	90.6	-	37.9	-	-
		1e-05	86.5 (1.24)	76.9	86.9	66.3	95.4	66.4	88.6	89.7	93.9	81.1	59.6	95.6	81.71
		3e-05	85.2	82.0	89.0	69.0	95.9	66.4 (0.76)	90.0	89.0	94.4	81.0	68.7	96.8	83.70
		0.0001	86.7 (1.40)	82.8 (0.26)	91.4 (0.16)	72.7 (0.69)	96.3 (0.07)	65.8 (0.39)	89.4 (1.08)	89.4 (0.57)	94.8 (0.16)	81.7 (0.10)	70.8 (0.45)	97.3 (0.05)	84.92
		0.0003	84.5	83.3	92.7	76.3	96.4 (0.03)	65.1	88.9	89.2	95.1 (0.20)	82.7	71.6	97.7 (0.08)	84.93
		0.001	81.8	80.8	93.9	76.6 (0.39)	96.4	62.8	87.2	84.7	94.9	83.0 (0.08)	69.7	97.7	83.55
		0.003	76.0	76.7	93.7 (0.17)	74.6	96.0	56.1	81.5	82.7	93.9	82.8	66.2	97.4	81.06
	SGD	**0.01**	87.7 (0.74)	83.6	89.5	68.5	96.1	65.7	87.4	90.9	94.2	81.6	68.9	96.7	83.73
		0.03	87.0	84.0 (0.08)	91.1	71.4	96.3 (0.07)	66.6 (1.59)	88.6	90.5 (0.47)	94.7	82.0	71.1	97.1	84.87
		0.1	87.9 (0.57)	83.7 (0.31)	92.8 (0.26)	74.5 (0.53)	96.4 (0.05)	65.9 (0.49)	88.3 (0.82)	90.6 (0.28)	94.9 (0.36)	82.5 (0.08)	71.5 (0.21)	97.4 (0.09)	85.53
		0.3	85.5	82.5	92.4 (0.56)	69.1 (3.67)	95.9	62.9	87.6	87.9	95.0 (0.25)	82.6 (0.28)	70.9	97.2	83.67
		1.0	61.6	65.1	91.4	61.9	7.6	35.4	55.4	64.6	82.6	80.0	62.0	95.6	63.17
		3.0	50.1	61.6	88.5	59.2	7.6	23.4	44.1	39.8	85.8	76.5	45.6	94.7	55.92
	DoG	-	87.4 (0.65)	84.0 (0.18)	91.4 (0.19)	71.9 (0.43)	96.2 (0.04)	66.1 (0.90)	88.5 (0.92)	90.3 (0.40)	94.8 (0.12)	82.0 (0.08)	71.6 (0.22)	97.2 (0.13)	85.12
	L-DoG	-	86.9 (0.27)	83.5 (0.18)	89.8 (0.31)	71.5 (0.24)	96.1 (0.03)	66.4 (0.61)	88.7 (0.70)	90.6 (0.13)	95.3 (0.17)	81.9 (0.10)	71.4 (0.25)	97.5 (0.09)	84.97
ResNet50	**Adam**	**0.0003**	87.8 (1.52)	84.8	91.0	73.3	96.2	68.5 (0.97)	92.7	93.1 (0.27)	95.5	82.2	73.9	97.0	86.03
		0.001	88.3 (1.18)	83.9 (0.31)	92.4 (0.21)	76.5 (0.53)	96.3 (0.04)	67.2 (0.94)	89.2 (1.83)	92.0 (0.37)	95.5 (0.25)	83.0 (0.18)	73.7 (0.33)	97.6 (0.07)	86.30
		0.003	86.9	83.1	93.3 (0.37)	78.3 (0.30)	96.1	64.4	90.1	90.2	95.4	83.4 (0.15)	72.2	97.7	85.67
		0.01	79.9	75.9	93.5	75.0	95.9	58.0	85.1	85.0	94.4	83.0	66.5	97.4	82.08
		0.03	62.8	64.9	92.4	71.3	95.3	46.9	63.1	67.0	89.3	82.0	50.8	96.4	73.08
	SGD	**0.01**	86.7	62.9	83.7	52.5	68.3	66.2	80.9	91.9	84.0	75.9	48.0	80.2	72.92
		0.03	86.7	77.0	88.0	63.0	88.5	67.2	82.1	92.8	90.5	78.8	64.7	91.4	80.50
		0.1	87.6 (0.66)	82.8	90.2	67.0	95.5	67.5 (1.71)	89.2	93.5	93.8	81.7	71.6	95.0	84.26
		0.3	87.4	84.8	91.1	70.7	95.9	70.5	85.8 (2.54)	92.9	94.7	82.3	73.9	96.1	84.98
		1.0	86.6 (0.50)	84.5 (0.46)	90.1 (0.51)	72.6 (1.56)	96.0 (0.08)	66.4 (1.16)	85.5 (2.39)	93.0 (0.30)	94.6 (0.28)	82.6 (0.09)	73.6 (0.44)	96.8 (0.06)	85.19
		3.0	87.4	84.7	91.8	70.1	96.0	66.9	90.3	92.1	94.8 (0.49)	82.1	73.7	97.0 (0.10)	85.23
		10.0	86.2	81.2	91.4 (0.71)	67.2	7.6	59.4	86.6	82.2	94.6	78.2	70.0	96.9	74.78
		30.0	0.4	12.1	20.0	22.2	7.4	24.1	43.6	16.5	83.6	73.6	3.1	6.7	25.75
	DoG	-	86.8 (0.62)	84.8 (0.37)	89.3 (0.58)	71.4 (0.77)	95.7 (0.09)	66.4 (1.48)	85.8 (2.64)	92.9 (0.38)	94.6 (0.33)	82.6 (0.16)	73.5 (0.41)	96.5 (0.10)	85.02
	L-DoG	-	87.6 (1.15)	84.6 (0.38)	90.8 (0.38)	75.0 (0.44)	96.0 (0.05)	69.1 (1.30)	90.2 (2.02)	92.4 (0.36)	95.6 (0.46)	82.2 (0.15)	74.6 (0.32)	97.4 (0.08)	86.29
VGG11	**Adam**	**3e-06**	80.2 (0.71)	-	-	-	-	-	-	-	-	79.8 (0.07)	-	93.8 (0.15)	-
		1e-05	80.5	73.7	89.3	63.5	94.3	61.5	82.1	87.3	91.4	80.0	65.4	95.3	80.00
		3e-05	82.2 (0.48)	74.9	90.5	67.7	96.1 (0.06)	61.4 (0.84)	84.0	88.1 (0.17)	92.9	81.1	66.6	96.4	81.48
		0.0001	82.6	75.6 (0.23)	92.4	71.9	7.6	61.6	83.5 (1.06)	87.2	93.5 (0.23)	82.3	66.9 (0.12)	96.8	74.79
		0.0003	82.3	74.1	93.2 (0.25)	74.4 (0.47)	7.5	59.6	83.0	86.0	93.4	82.9 (0.08)	66.3	96.8 (0.14)	74.77
		0.001	61.7	61.1	24.5	64.6	7.5	48.0	53.7	65.0	89.1	73.6	50.5	96.5	57.58
		0.003	-	1.0	89.4	68.4	7.6	2.1	0.5	2.7	76.2	-	30.5	-	-
		0.01	-	1.0	24.5	22.2	7.6	2.1	1.2	2.7	2.2	-	2.0	-	-
	SGD	**0.001**	81.0	68.5	85.0	62.3	14.8 (3.59)	61.3	80.3	88.0	89.3	78.9	61.9	92.0	71.65
		0.003	81.8	72.4	90.3	64.2	12.3	62.2	80.9	88.0	90.9	80.3	64.9	94.4	73.08
		0.01	82.4	73.5	91.9 (0.19)	67.0	9.8	61.2	82.0 (0.99)	89.0 (0.37)	91.7	81.7	65.8	95.7	73.91
		0.03	83.0 (0.50)	74.7 (0.21)	92.1 (0.19)	69.1 (0.36)	7.6 (0.04)	61.9 (0.41)	82.3 (1.09)	89.4 (0.32)	92.5 (0.28)	82.2 (0.06)	66.1 (0.10)	96.4 (0.08)	74.77
		0.1	49.6 (44.91)	74.6 (0.08)	20.0	22.2	7.6	60.5	0.3	87.5	2.2	73.6	53.2 (29.49)	6.7	37.87
		0.3	-	1.0	20.0	22.2	7.4	2.1	0.3	2.7	2.2	-	0.5	-	-
		1.0	-	1.0	20.0	22.2	7.4	2.1	0.3	2.7	2.2	-	0.5	-	-
	DoG	-	82.9 (0.45)	74.7 (0.22)	91.5 (0.17)	68.4 (0.53)	10.4 (0.82)	62.5 (1.19)	82.8 (0.95)	89.5 (0.23)	92.4 (0.22)	81.6 (0.07)	66.3 (0.33)	96.3 (0.09)	74.94
	L-DoG	-	82.4 (0.60)	74.8 (0.16)	92.0 (0.11)	69.9 (0.44)	92.1 (8.59)	62.6 (0.95)	82.7 (1.15)	88.9 (0.62)	92.4 (0.15)	81.9 (0.18)	65.8 (0.09)	96.5 (0.12)	81.83
ViT-B/32	**Adam**	**3e-06**	90.7 (0.76)	92.3 (0.22)	89.5 (0.35)	66.0 (0.69)	94.0 (0.24)	74.9 (0.24)	98.5 (0.53)	91.6 (0.11)	95.5 (0.07)	79.7 (0.08)	75.5 (0.23)	96.8 (0.06)	87.08
		1e-05	90.2 (0.47)	92.8 (0.21)	89.9 (0.52)	67.5	51.5 (48.10)	76.9	98.7 (0.29)	90.7	96.3	79.8	78.0 (0.12)	97.6	83.84
		3e-05	88.6	92.5	89.7	70.1	7.6	75.3 (0.24)	98.7	91.6 (0.21)	96.5 (0.13)	80.1 (0.09)	78.3	97.7	80.21
		0.0001	89.5	91.2	89.1	70.8 (0.32)	7.6	72.9	98.1	90.0	96.1	80.1	77.0	97.8 (0.07)	79.80
		0.0003	87.9	87.3	87.9	68.5	7.6	69.0	94.9	87.9	94.9	79.3	72.5	97.9	77.33
		0.001	80.3	62.1	88.1	51.2	7.6	50.4	71.0	58.3	88.6	73.6	53.6	96.3	64.75
		0.003	-	13.5	64.6	29.3	7.6	14.1	26.7	12.1	71.9	-	19.5	-	-
	SGD	**3e-05**	-	6.1	52.7	40.8	32.7	45.9	61.4	80.3	59.7	-	5.2	-	-
		0.0001	85.0	73.7	71.3	50.4	57.0	69.0	98.1	90.1	83.0	75.3	24.7	79.1	71.17
		0.0003	88.7	88.7	83.1	60.3	69.6	74.7	98.8	91.9	90.2	76.6	59.6	90.7	80.50
		0.001	90.8	91.7	88.1	65.6	87.5	74.8	98.7 (0.51)	93.0 (0.21)	93.5	78.2	73.4	95.2	85.48
		0.003	90.9 (0.89)	92.8 (0.10)	87.3 (1.28)	66.3 (0.40)	85.6 (15.77)	75.3 (0.29)	98.9 (0.31)	92.5 (0.27)	95.3 (0.10)	79.4 (0.02)	77.3 (0.16)	96.6 (0.17)	86.52
		0.01	90.7 (0.63)	92.9 (0.14)	85.9	68.8	56.1	75.2	99.3	92.5	95.8	79.7	78.9 (0.08)	97.4	84.04
		0.03	89.8	92.5	83.1	69.7 (0.29)	65.8	75.8 (0.59)	99.3	92.2	96.2 (0.06)	78.9 (2.51)	78.3	97.7 (0.05)	84.69
		0.1	88.0	91.2	25.2	22.2	7.6	74.9	98.7	91.0	95.9	73.6	76.8	97.8	69.75
		0.3	0.4	1.0	24.5	22.2	7.6	70.5	1.8	2.7	2.3	73.6	0.5	19.6	18.42
	DoG	-	89.5 (1.26)	92.5 (0.22)	85.0 (0.27)	69.5 (0.11)	67.7 (36.66)	75.5 (0.71)	98.9 (0.25)	92.4 (0.16)	96.4 (0.10)	79.7 (0.01)	77.8 (0.13)	97.7 (0.08)	85.22
	L-DoG	-	89.6 (0.81)	92.8 (0.15)	86.0 (0.40)	70.7 (0.29)	95.3 (0.08)	75.8 (0.71)	99.0 (0.26)	92.3 (0.45)	96.5 (0.17)	79.8 (0.07)	78.3 (0.26)	97.8 (0.04)	87.82

Table 5: Average (std) test accuracy across seeds for vision tasks, when fine-tuned with different optimization algorithms and their respective base learning rate when applicable. DoG and L-DoG use $r_\epsilon = 10^{-4}(1 + \|x_0\|)$.

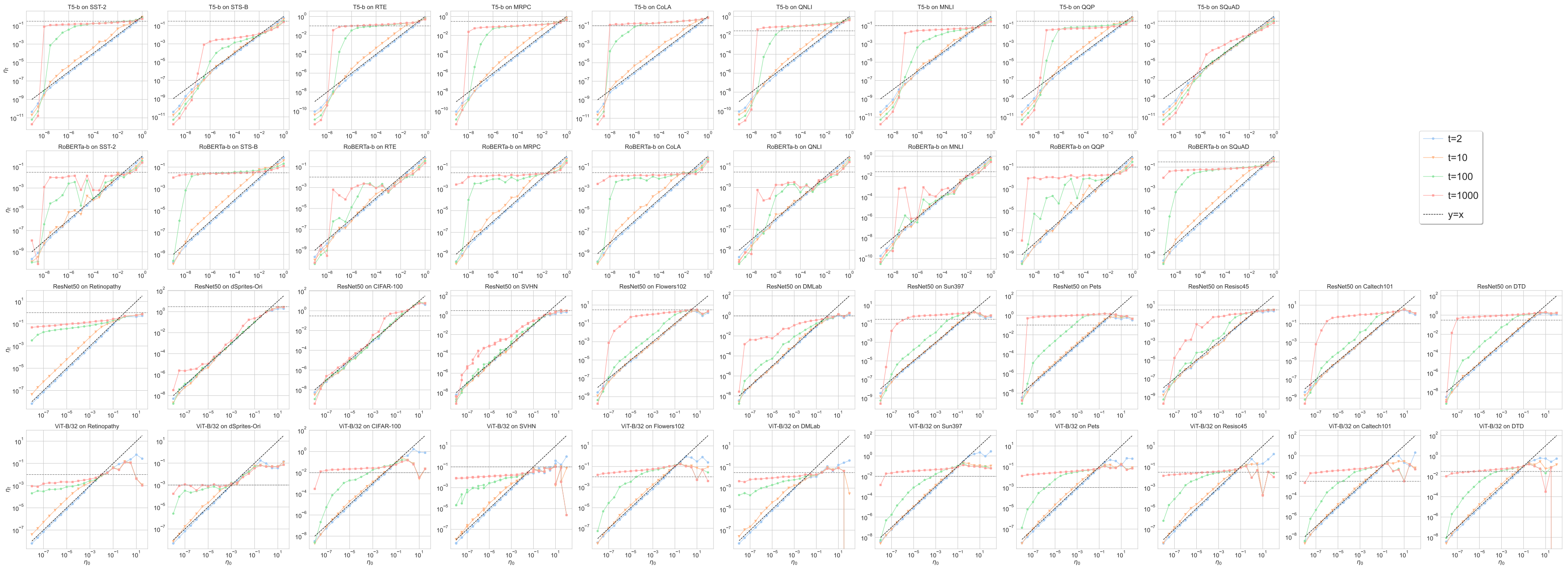


Figure 8: Stabilizing behavior of DoG on η_t as a function of η_0 (x-axis) and t (color). In most cases η_t quickly stabilizes around a value close to the optimal SGD base learning rate for all sufficiently small $\eta_0 = r_\epsilon / \|g_0\|$. The main exceptions (where η_t depends strongly on η_0) are dSprites-Ori, CIFAR-100 and SVHN when trained with ResNet50; see E.3 for further discussion.

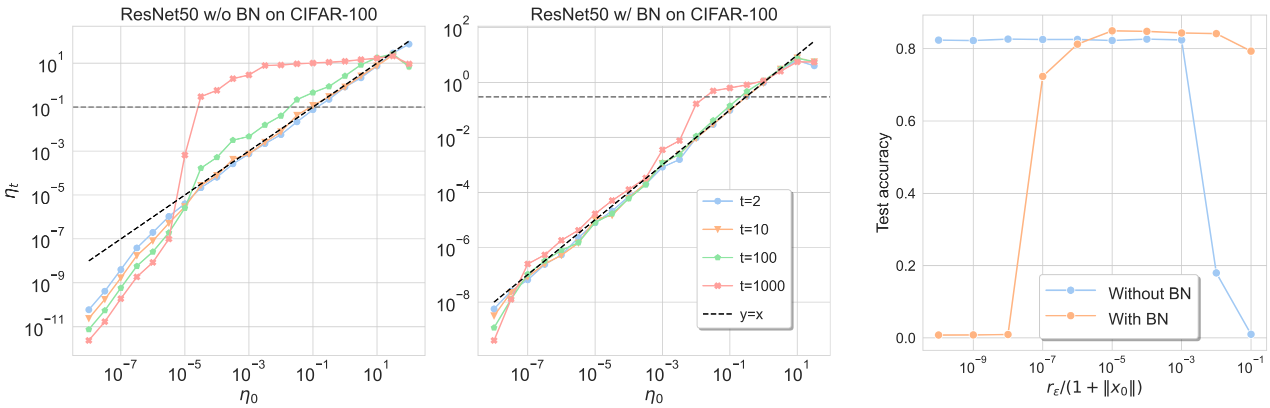
E.2 Fine-tuning CoLA

As discussed in Section 4.2, DoG with $r_\epsilon = 10^{-6}(1 + \|x_0\|)$ failed in fine-tuning T5-b on CoLA. To investigate this issue, we ran DoG and L-DoG with different choices of r_ϵ. Figure 10 depicts the results of this test as well as the performance of SGD and Adam with different learning rates. The figure shows that using lower values of r_ϵ allows DoG to reach reasonable results, but with some seeds still failing. In contrast, L-DoG shows consistent and superior performance across a large range of r_ϵ values. We leave further investigations on the cause of failure in CoLA to future work.

E.3 Sensitivity of DoG to r_ϵ and the effect of batch normalization

In Section 4.3, we discuss DoG's insensitivity to the choice of r_ϵ as long as it is small enough. Here, we expand on this analysis by testing how the DoG step size at iteration t, denoted η_t, depends on its initial step size $\eta_0 = r_\epsilon / \|g_0\|$. For for each task and in our testbed and 4 models, we perform short training runs with a large numbers of η_0 values. In Figure 8 we plot η_t vs. η_0 for $t \in \{2, 10, 100, 1000\}$. We also show a horizontal line for the best peak step size of SGD, and the $y = x$ diagonal. The figure shows that for most model/task combinations, η_t converges quickly (within the first 1000 steps) to a value near the optimal one for SGD, and mostly independent of η_0 as long as it is small enough.

However, we also observe some failure cases where η_t strongly depends on η_0, such as fine-tuning ResNet50 on CIFAR-100. This provides a complementary perspective on the fact DoG is sensitive to r_ϵ in this setting, as already shown in Figure 3: when η_0 is to low, DoG fails to reach a suitable value of η_t in a reasonable time. We hypothesize that this is due to the batch normalization (BN) layers in the model causing many different step size to "look" like solutions to the implicit equation motivating DoG. To test this hypothesis, we repeat the CIFAR-100 training experiment but without BN (we disable BN by fine-tuning the model in evaluation mode). Figure 9(a) shows that removing BN allows DoG to recover its stabilizing behavior. Moreover, Figure 9(b) further shows that without batch normalization, the performance of DoG again becomes insensitive to the choice of r_ϵ provided it is sufficiently small. Unsurprisingly, we also observe that removing BN slightly hurts generalization performance in this task. As mentioned in Section 6, improving DoG to be more robust in the presence of normalization layers in general and batch normalization in particular is an important direction for future research.



(a) (b)

Figure 9: ResNet50 fine-tuned on CIFAR-100 with and without batch normalization. **(a)** Stabilizing behavior of DoG on η_t as a function of η_0 (x-axis) and t (color). Turning off batch normalization (left) mitigates the sensitivity of η_t to η_0 observed in batch normalized model (right). **(b)** Accuracies of models trained with DoG (for 20K steps) as a function of r_ϵ. Without batch normalization, DoG is robust to smaller values of r_ϵ.

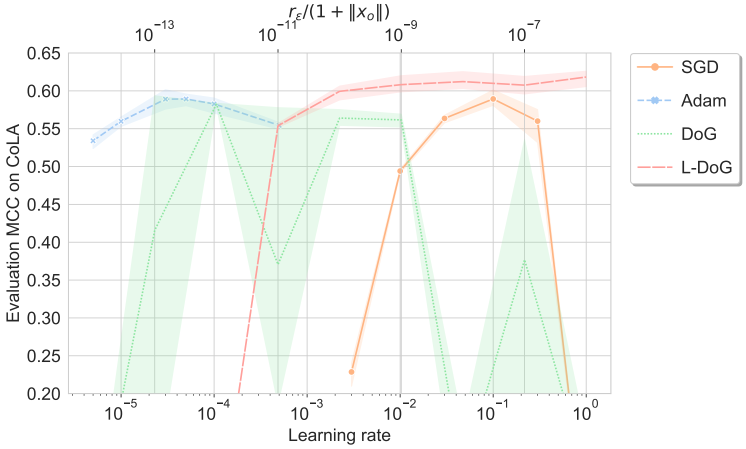


Figure 10: Matthews correlation of T5-base fine-tuned on CoLA with SGD and Adam with different base learning rates (bottom axis), as well as with DoG and L-DoG with different r_ϵ (top axis). Only L-DoG and Adam perform consistently well across different parameters. The lines and shaded regions show the average Matthews correlation and the min-max range, respectively, computed over 3 seeds.

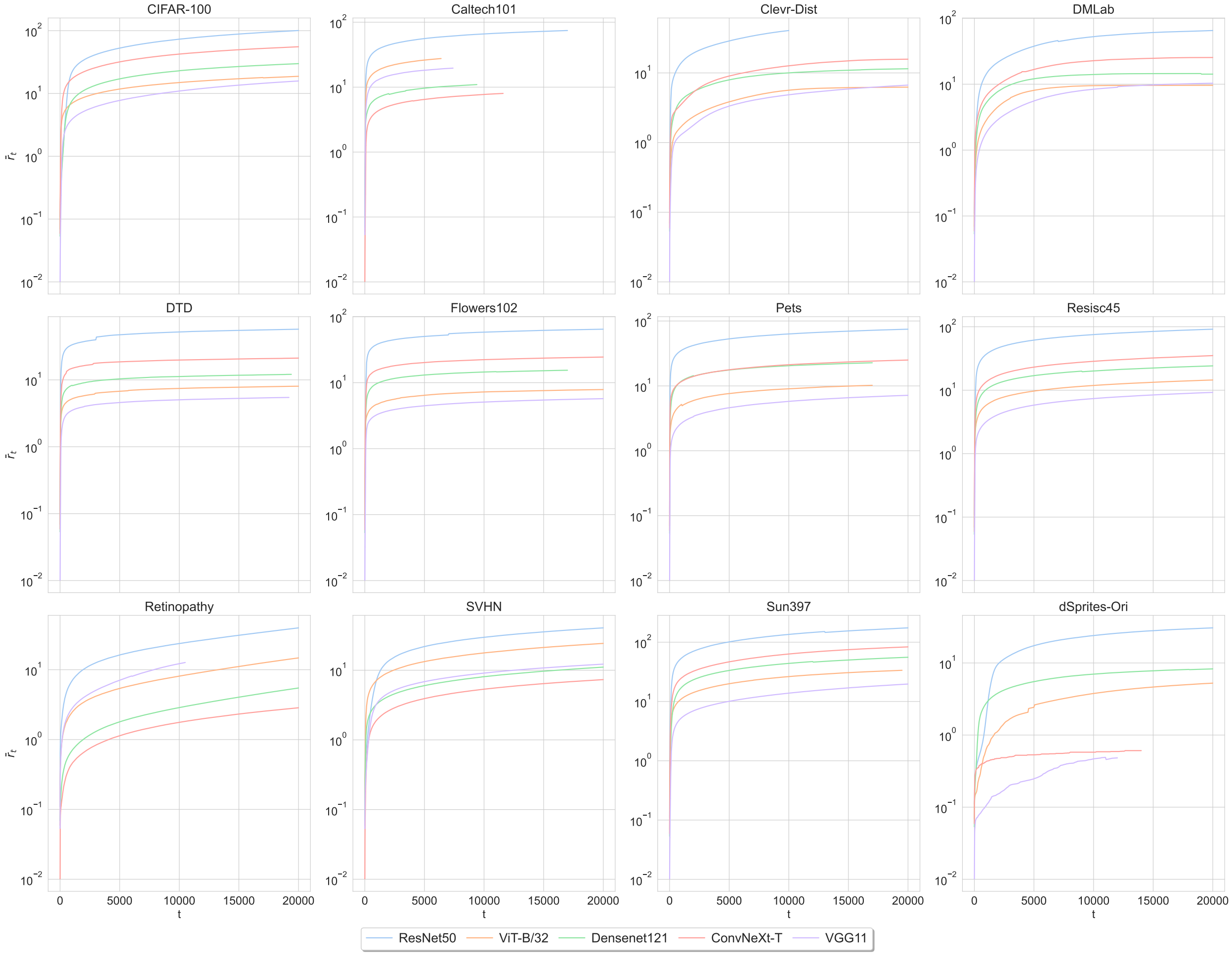


Figure 11: The quantity $\bar{r}_t = \max_{i \leq t} \|x_i - x_0\|$ as a function of the number of steps t in our computer vision testbed. The value of $\bar{r}_t$ grows rapidly at first and then almost plateaus.