Les Cahiers du GERAD ISSN: 0711–2440

A stochastic proximal method for nonsmooth regularized finite sum optimization

D. Lakhmiri, D. Orban, A. Lodi

G–2022–27

June 2022

GERAD HEC Montréal
3000, chemin de la Côte-Sainte-Catherine
Montréal (Québec) Canada H3T 2A7

Tél. : 514 340-6053
Téléc. : 514 340-5665
info@gerad.ca
www.gerad.ca

A stochastic proximal method for nonsmooth regularized finite sum optimization

Dounia Lakhmiri [a, b, c]

Dominique Orban [a, c]

Andrea Lodi [a, b, d]

[a] *GERAD, Montréal (Qc), Canada, H3T 1J4*

[b] *Canada Excellence Research Chair in "Data Science for Real-time Decision-making", Montréal (Qc), Canada, H3T 2A7*

[c] *Department of Mathematics and Industrial engineering, Polytechnique Montréal, Montréal (Qc), Canada, H3T 2A7*

[d] *Jacobs Technion-Cornell Institute, Cornell University, Ithaca, NY 14850, United States*

dounia.lakhmiri@polymtl.ca
dominique.orban@polymtl.ca
andra.lodi@cornell.edu

June 2022
Les Cahiers du GERAD
G–2022–27

Abstract : We consider the problem of training a deep neural network with nonsmooth regularization to retrieve a sparse and efficient sub-structure. Our regularizer is only assumed to be lower semi-continuous and prox-bounded. We combine an adaptive quadratic regularization approach with proximal stochastic gradient principles to derive a new solver, called SR2, whose convergence and worst-case complexity are established without knowledge or approximation of the gradient's Lipschitz constant. We formulate a stopping criteria that ensures an appropriate first-order stationarity measure converges to zero under certain conditions. We establish a worst-case iteration complexity of $\mathcal{O}(\epsilon^{-2})$ that matches those of related methods like ProxGEN, where the learning rate is assumed to be related to the Lipschitz constant. Our experiments on network instances trained on CIFAR-10 and CIFAR-100 with ℓ_1 and ℓ_0 regularizations show that SR2 consistently achieves higher sparsity and accuracy than related methods such as ProxGEN and ProxSGD.

Keywords : Pruning neural networks, regularization, proximal stochastic methods, nonsmooth nonconvex optimization, finite sum optimization

Acknowledgements: This work has been supported by the NSERC Alliance grant 544900–19 in collaboration with Huawei-Canada, the Canada Excellence Research Chair in "Data Science for Real-time Decision-making", and Cornell Tech.

1 Introduction

We focus on the problem of training neural networks with regularization expressed as

$$\min_x F(x) := f(x) + \mathcal{R}(x), \qquad f(x) := \frac{1}{N}\sum_{i=1}^{N} f_i(x), \tag{1}$$

where $x \in \mathbb{R}^n$ are the parameters, f is the loss function, and $\mathcal{R}$ may be nonsmooth, nonconvex, and take infinite values. Instances of (1) are often used as approximations of

$$\min_x \mathbb{E}_{\omega\sim\mathcal{P}}[f(x,\omega)] + \mathcal{R}(x),$$

where ω follows a distribution $\mathcal{P}$. In (1), $\mathcal{R}$ helps select a solution with desirable features among all potential minimizers of f. Examples include the weight decay technique, which uses $\mathcal{R}(x) := \|x\|_2$ to avoid over-fitting the training data (Krogh and Hertz, 1992; Zhou et al., 2021). Other applications employ a specific regularizer, whether convex, such as $\|\cdot\|_1$, or nonconvex, such as $\|\cdot\|_0$, to retrieve a sparse sub-network for network pruning (Hoefler et al., 2021; Wang et al., 2019; Yang et al., 2019a) or quantization (Bai et al., 2018; Wess et al., 2018). For the rest of this work, we focus on sparsity-promoting $\mathcal{R}$.

We introduce SR2,[1] a stochastic variant of the quadratic regularization method that solves (1) for nonsmooth, nonconvex regularizers. Our main contributions are

1. to the best of our knowledge, the first stochastic adaptive quadratic regularization method for (1) with weak assumptions on $\mathcal{R}$;
2. the formulation of a stopping criterion and a first order stationarity measure adapted to nonsmooth, non-convex stochastic optimization problems;
3. the convergence of a first-order stationarity measure to zero without assuming knowledge of the Lipschitz constant of ∇f, and worst-case $\mathcal{O}(\epsilon^{-2})$ iteration complexity;
4. numerical experiments on multiple instances of deep neural networks (DNNs) to retrieve a sparse sub-network. In most cases, SR2 achieves high sparsity levels without post-treatment. A comparison against two related proximal solvers, ProxSGD and ProxGEN, in terms of accuracy and sparsity of the solution is favorable for SR2.

1.1 Background and related work

The stochastic gradient (SG) method (Kiefer and Wolfowitz, 1952; Robbins and Monro, 1951), and its variants (Ruder, 2016; Kingma and Ba, 2015; Nguyen et al., 2017), are a common approach for (1) when $\mathcal{R} = 0$. At iteration t, SG selects a sample set $\xi_t \subseteq \{1,\dots,n\}$, computes the sampled gradient $g_t = \frac{1}{|\xi_t|}\sum_{i\in\xi_t}\nabla f_i(x_t)$, and updates

$$x_{t+1} \leftarrow x_t - \alpha_t g_t, \tag{2}$$

where $\alpha_t > 0$ is the step size, or learning rate. SG and variants typically accept every step regardless of whether the objective decreases or not. For this reason, we do not refer to it as SGD, where D would stand for *descent*. SG can be shown to converge in expectation under certain assumptions on the learning rate and on the quality of g_t (Bottou et al., 2018).

Proximal gradient descent (PGD) (Fukushima and Mine, 1981) is suited to the structure of (1), i.e., when $\mathcal{R} \neq 0$. At iteration t, it computes a step

$$s_t \in \operatorname*{argmin}_s \tfrac{1}{2}\|s+\alpha_t g_t\|^2 + \alpha_t\mathcal{R}(x_t+s) = \operatorname*{argmin}_s g_t^T s + \tfrac{1}{2}\alpha_t^{-1}\|s\|^2 + \mathcal{R}(x_t+s) := \operatorname*{prox}_{\alpha_t\mathcal{R}(x_t+\cdot)}(-\alpha_t g_t) \tag{3}$$

for a prescribed $\alpha_t > 0$, followed by the update $x_{t+1} := x_t + s_t$.

[1] https://github.com/DouniaLakhmiri/SR2

Observe that due to the nonsmoothness and/or nonconvexity of $\mathcal{R}$, the right-hand side of (3) may contain several elements. The key point is that a closed form solution of (3) is known for a wide range of choices of $\mathcal{R}$ (Beck, 2017; Rockafellar and Wets, 1998). PGD has been substantially studied in the deterministic case and is provably convergent to first-order stationary points under weak assumptions (Karimi et al., 2016; Teboulle, 1997). In the case $g_t = \nabla f(x_t)$, s_t is guaranteed to result in a decrease in F provided that $\alpha_t \leq 1/L$ (Bolte et al., 2014, Lemma 2), L being the Lipschitz constant of ∇f.

Several variants have been successfully adapted to training deep networks and often provide proof of convergence towards critical solutions (Davis and Drusvyatskiy, 2019; Pham et al., 2020; Xu et al., 2019; Yang et al., 2019b; Yun et al., 2021). They differ in the way they solve (3), in whether α_t is fixed or adaptive, in the use of a momentum term, a preconditioner, and other ML techniques that speed up convergence during training.

1.2 Motivation and proposed approach

One notable and common assumption behind the convergence proof of the variants of SG and PGD is the initial learning rate $\alpha_0 \leq 1/L$. In practice, however, especially in deep learning, L is unknown.

In the *adaptive quadratic regularization* method, to which we will refer as R2, α_t is adjusted based on the objective decrease observed at iteration t. R2 was initially proposed for the case with $\mathcal{R} = 0$ and the term *regularization* in its name should not be confused with the nonsmooth term $\mathcal{R}$ in (1). About x_t, a step s_t is computed that minimizes the linear model $\varphi(s; x_t) := f(x_t) + \nabla f(x_t)^T s \approx f(x_t + s)$ to which we add the quadratic regularization term $\frac{1}{2}\sigma_t \|s\|^2$, where $\sigma_t > 0$ is a regularization parameter. The larger σ_t, the shorter we may expect s_t to be. Conversely, small values of σ_t may allow us to compute large steps and make fast progress. By completing the square, note that minimizing $\varphi(s; x_t) + \frac{1}{2}\sigma_t \|s\|^2$ amounts to minimizing $\frac{1}{2}\sigma_t \|s + \sigma_t^{-1} \nabla f(x_t)\|^2$, which corresponds to (3) with $\alpha_t := 1/\sigma_t$ and may be viewed as gradient descent with adaptive step size.

Lotfi et al. (2020, 2021) propose stochastic variants of R2 along with second-order methods for large scale machine learning when $\mathcal{R} = 0$. The fact that R2 appears closely relaxed to PGD motivated Aravkin et al. (2021) to generalize it to nonsmooth regularized problems with especially weak assumptions on $\mathcal{R}$. In the convergence analysis, the value of L is never explicitly needed.

Organization The rest of the manuscrip is organized as follows. Section 2 gives a brief overview of ProxSGD and ProxGEN, two proximal methods related to SR2. Section 3 develops SR2 and justifies the methodology. Section 4 establishes the convergence guarantees towards a first-order stationary point w.p.1 and an iteration complexity analysis. In Section 5, we present numerical results and experiments. We conclude with a discussion in Section 6.

Notation $\|x\|$ is the Euclidean norm of $x \in \mathbb{R}^n$. $|\mathcal{S}|$ is the number of elements in the set $\mathcal{S}$. We introduce a stochastic variable $\xi : \mathbb{N} \to \mathcal{P}(\{1, \ldots, N\}) \setminus \varnothing$, whose domain represents an iteration counter, and which takes values in the set of nonempty samples of the sum in (1). For a realization $\xi_t := \xi(t)$ of ξ at iteration t we denote

$$f(x, \xi_t) := \frac{1}{|\xi_t|} \sum_{j \in \xi_t} f_j(x), \quad g_t := \nabla f(x, \xi_t) \; = \frac{1}{|\xi_t|} \sum_{j \in \xi_t} \nabla f_j(x)$$

the sampled, or stochastic, objective and gradient. We also write $F(x, \xi_t) := f(x, \xi_t) + \mathcal{R}(x)$. We note $\mathbb{E}_\xi$ the expectation over the distribution of ξ, while $\mathbb{E}_{\xi_t}$ represents the expectation over the distribution of ξ that yields a success knowing x_t. The abbreviation w.p.1 means "with probability one".

2 Overview of ProxSGD and ProxGEN

ProxSGD (Yang et al., 2019b) and ProxGEN (Yun et al., 2021) are two approaches based on the adaptation of the proximal gradient method, although neither is a descent method. Both consider a variant of (3) with a momentum term v_t instead of g_t, and a preconditioner.

ProxSGD assumes that $\mathcal{R}$ is convex, and computes

$$s_t \in \operatorname*{argmin}_s \; v_t^T s + \tfrac{1}{2} s^T D_t s + \mathcal{R}(x_t + s), \tag{4a}$$

$$x_{t+1} = x_t + \alpha_t s_t, \tag{4b}$$

where D_t is a positive-definite diagonal matrix. Note that ProxSGD does not exactly fit in the framework (3). Yang et al. (2019b) show convergence to a first-order stationary point w.p.1., but do not provide a complexity bound.

Although Yun et al. (2021) do not explicitly mention their assumptions on $\mathcal{R}$, they mention that ProxGEN does not require it to be convex. ProxGEN may be seen as a proximal generalization of several SG variants like Adam, Adagrad, etc. that matches (3) more closely than (4). It computes

$$s_t \in \operatorname*{argmin}_s \; v_t^T s + \tfrac{1}{2} \alpha_t^{-1} s^T D_t s + \mathcal{R}(x_t + s), \tag{5a}$$

$$x_{t+1} = x_t + s_t. \tag{5b}$$

The authors show convergence to a first-order stationary point, and a worst-case complexity of $\mathcal{O}(\epsilon^{-2})$ in terms of iterations and $\mathcal{O}(\epsilon^{-4})$ overall to achieve $\mathbb{E}_a[\operatorname{dist}(0, \hat{\partial} F(x_a))] \leq \epsilon$ when the batch size is fixed, where x_a is an iterate drawn uniformly randomly from $\{x_1, \ldots, x_T\}$, and T is the maximum number of iterations.

The method we propose in the next section, SR2, has convergence results similar to ProxGEN but the version we present includes neither a momentum term nor a preconditioner, and it relies on an implicit assumption on the batch size—see Assumption 3 below.

3 Stochastic quadratic regularization: SR2

Recall that $\mathcal{R} : \mathbb{R}^n \to \mathbb{R} \cup \{\pm\infty\}$ is proper if it never takes the value $-\infty$ and $\mathcal{R}(x) < \infty$ for at least one $x \in \mathbb{R}^n$, lower semi-continuous at $\bar{x} \in \mathbb{R}^n$ if $\liminf_{x \to \bar{x}} \mathcal{R}(x) \geq \mathcal{R}(\bar{x})$, and prox-bounded if there exists $x \in \mathbb{R}^n$ and $\lambda_x > 0$ such that $\inf_w \{\frac{1}{2}\lambda_x^{-1}\|x - w\|^2 + \mathcal{R}(w)\} > -\infty$. The supremum of all such λ_x is the threshold of prox-boundedness of $\mathcal{R}$, which we also refer to as λ_x. Any function that is bounded below is prox-bounded with $\lambda_x = +\infty$, but certain unbounded regularizers, such as $-\|x\|$ or $-\|x\|^2$, are also prox-bounded. Our assumptions on (1) are as follow.

Assumption 1. There exists $L > 0$ such that f is L-smooth, i.e., $\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|$ for all x, $y \in \mathbb{R}^n$. In addition, $\mathcal{R}$ is proper and lower semi-continuous at all $x \in \mathbb{R}^n$, and $s \mapsto \mathcal{R}(x_t + s)$ is prox-bounded for each x_t encountered during the iterations.

Under the previous assumption, the appropriate concept of subdifferential is the following.

Definition 1. The Fréchet subdifferential $\hat{\partial}\phi(\bar{x})$ of $\phi : \mathbb{R}^n \to \mathbb{R} \cup \{\pm\infty\}$ at $\bar{x}$ where ϕ is finite is the set of $v \in \mathbb{R}^n$ such that

$$\liminf_{\substack{x \to \bar{x} \\ x \neq \bar{x}}} \frac{\phi(x) - \phi(\bar{x}) - v^T(x - \bar{x})}{\|x - \bar{x}\|} \geq 0.$$

Assumption 2. $\mathcal{R}$ is such that $\hat{\partial}\mathcal{R} \neq \emptyset$, which implies $\hat{\partial} F = \nabla f + \hat{\partial}\mathcal{R} \neq \emptyset$,

Our assumptions on $\mathcal{R}$ are satisfied for many sparsity-promoting regularizers of interest, including $\|x\|_0$, $\|x\|_p$, $\|x\|_p^p$ for $0 < p < 1$, and the indicator of $\{x \mid \|x\|_0 \leq k\}$ for fixed $k \in \{0, \ldots, n\}$. Note that Assumption 2 excludes regularizers such as $-\|x\|$ or $-\|x\|^2$.

As in the deterministic version R2 (Aravkin et al., 2021), SR2 uses a linear model of f defined at each iteration t as $\varphi(s; x_t) = f(x_t, \xi_t) + {g_t}^T s$, such that $\varphi(0; x_t) = f(x_t, \xi_t)$ and $\nabla_s \varphi(0; x_t) = g_t$. Let

$$\psi(s; x_t) := \varphi(s; x_t) + \mathcal{R}(x_t + s). \tag{6}$$

Note that the analysis of Aravkin et al. (2021) makes provision for using a model of $\mathcal{R}$ about x_t. In the interest of clarity, we use the ideal $\mathcal{R}(x_t + s)$ in the sequel, but our analysis below could just as easily accommodate a model.

For a regularization parameter $\sigma_t > 0$, we also define

$$m(s; x_t, \sigma_t) = \psi(s; x_t) + \tfrac{1}{2}\sigma_t \|s\|^2. \tag{7}$$

SR2 starts the iteration with computing a step s_t that minimizes (7), which is equivalent to computing a proximal stochastic gradient step with step size ${\sigma_t}^{-1}$:

$$s_t \in \operatorname*{argmin}_s m(s; x_t, \sigma_t) = \operatorname*{prox}_{{\sigma_t}^{-1}\mathcal{R}}({\sigma_t}^{-1} g_t). \tag{8}$$

Because the Lipschitz constant of $\nabla\varphi(.; x_t)$ is zero, s_t is guaranteed to result in a decrease in $\psi(\cdot; x_t)$ (Bolte et al., 2014, Lemma 2). However, the latter does not necessarily correlate with a decrease in F. Therefore, SR2 compares the ratio ρ_t of the decrease in $F(.)$ to that in $\psi(\cdot; x_t)$ between x_t and $x_t + s_t$ to decide on the acceptance of the step. The value of ρ_t, which is indicative of the adequacy of the model $\psi(\cdot; x_t)$ along s_t, also guides the update of σ_t. The procedure is stated in Algorithm 1.

Algorithm 1: SR2: Stochastic nonsmooth quadratic regularization.

Input: $0 < \eta_1 \le \eta_2 < 1$, $0 < \gamma_3 \le 1 < \gamma_1 \le \gamma_2$, $x_0 \in \mathbb{R}^n$ where $\mathcal{R}$ is finite, $\sigma_0 \ge \sigma_{\min} > 0$
for $t = 1, \ldots$ **do**
  Draw ξ_t and define g_t ;
  Define $m(s; x_t, \sigma_t)$ as in (7) ;
  Compute $s_t \in \operatorname{argmin}_s m(s; x_t, \sigma_t)$;
  if *ξ_t does not satisfy Assumption 2* **then**
    set $s_t = 0$;
  else
    Compute $\Delta F_t := F(x_t) - F(x_t + s_t)$;
    Compute $\Delta\psi_t := \psi(0; x_t) - \psi(s_t; x_t)$;
    Compute $\rho_t := \Delta F_t / \Delta\psi_t$;
    if $\rho_t \ge \eta_1$ **then**
      set $x_{t+1} = x_t + s_t$; accept step
    else
      set $x_{t+1} = x_t$; reject step
  Set $\sigma_{t+1} \in \begin{cases} [\max(\sigma_{\min}, \gamma_3\sigma_t), \sigma_t] & \text{if } \rho_t \ge \eta_2 \quad (\sigma_t \searrow) \\ [\sigma_t, \gamma_1\sigma_t] & \text{if } \eta_1 \le \rho_t < \eta_2 \quad (\sigma_t \approx) \\ [\gamma_1\sigma_t, \gamma_2\sigma_t] & \text{if } \rho_t < \eta_1 \quad (\sigma_t \nearrow) \end{cases}$

The importance of prox-boundedness in Algorithm 1 resides in the update of σ_t. If $\sigma_t < 1/\lambda_{x_t}$, (7) is unbounded below, so that $\Delta m_t = +\infty$. Because $\mathcal{R}$ is proper, ΔF_t is either finite or $+\infty$. Either way, the rules of extended arithmetic in nonsmooth optimization imply $\rho_t = 0$, and therefore the step is rejected and σ_t is increased. After a finite number of such increases, $\sigma_t \ge 1/\lambda_{x_t}$ and a step that yields finite Δm_t can be assessed. A key result stated as Theorem 1 below is that as soon as σ_t is sufficiently large, the step will be accepted.

Section 4 establishes the convergence properties of SR2, for which we require assumptions that ensure g_t behaves somewhat similarly to $\nabla f(x_t)$. Comparable conditions appear in (Bottou et al., 2018; Bollapragada et al., 2018).

Assumption 3. There exists $\kappa_m > 0$ such that for all t,

$$|f(x_t + s_t) - f(x_t) - {g_t}^T s_t| \le \kappa_m \|s_t\|^2.$$

In addition $\mathbb{E}_\xi[f(x_t, \xi)] = f(x_t)$, $\mathbb{E}_\xi[g_t] = \nabla f(x_t)$

Assumption 3 states that the stochastic gradient should behave similarly to a full gradient, which implicitly involves a condition on the batch size. If the assumption is not respected, the batch-size should be increased. The process is finite because due to Assumption 1, the inequality of Assumption 3 holds with $\kappa_m = \frac{1}{2}L$ when $g_t = \nabla f(x_t)$. This is similar in spirit to implementing a variance reduction strategy, a standard condition for the convergence of stochastic gradient methods (Bottou et al., 2018).

4 Convergence analysis

Under Assumption 1, x^* is first-order stationary for (1) if $0 \in \hat{\partial} F(x^*) = \nabla f(x^*) + \hat{\partial}\mathcal{R}(x^*)$ (Rockafellar and Wets, 1998, Theorem 10.1).

The following result mirrors (Aravkin et al., 2021, Theorem 6.2) and shows that SR2 cannot generate a infinite number of failed iterations unless the step is zero. We require the following final assumption stating that $s \mapsto \mathcal{R}(x_t + s)$ are *uniformly* prox-bounded. The assumption is trivially satisfied for any $\mathcal{R}$ that is bounded below.

Assumption 4. There exists $\lambda > 0$ such that $\lambda_{x_t} \geq \lambda$ for all x_t encountered during the iterations.

Theorem 1. *Let Assumptions 1, 3 and 4 hold. If $s_t \neq 0$ and $\sigma_t \geq \sigma_{\text{succ}} := \max(2\kappa_m/(1-\eta_2), 1/\lambda)$, then s_t is accepted and $\sigma_t \leq \sigma_t$.*

Proof. As explained above, we assume that $\sigma_t \geq 1/\lambda \geq 1/\lambda_{x_t}$ to ensure that Δm_t is finite. By definition of s_t, $m_t(s_t, x_t, \sigma_t) \leq m_t(0, x_t, \sigma_t)$, i.e.,

$$g_t{}^T s_t + \mathcal{R}(x_t + s_t) + \tfrac{1}{2}\sigma_t \|s_t\|^2 \leq \mathcal{R}(x_t). \tag{9}$$

The definition of ρ_t, Assumption 3 and (9) yield

$$|\rho_t - 1| = \left| \frac{f(x_t) + g_t{}^T s_t - f(x_t + s_t)}{\mathcal{R}(x_t) - g_t{}^T s_t - \mathcal{R}(x_t + s_t)} \right| \leq \frac{2\kappa_m \|s_t\|^2}{\sigma_t \|s_t\|^2} = \frac{2\kappa_m}{\sigma_t} \leq \frac{2\kappa_m}{\sigma_{\text{succ}}} = 1 - \eta_2.$$

Thus, $\rho_t \geq \eta_2$ and $\sigma_t \leq \sigma_t$. □

As a consequence of Theorem 1, there is a constant $\sigma_{\max} := \min\{\sigma_0, \gamma_2 \sigma_{\text{succ}}\} > 0$ such that for all $t, \sigma_t \leq \sigma_{\max}$.

Next, we analyze the scenario where SR2 only generates a finite number of successes, and show that the method converges to a first order stationary point w.p.1 in this case.

Theorem 2. *Let Assumptions 1, 3 and 4 hold. If Algorithm 1 only generates a finite number of successes, $x_t = x_{t^*}$ for all sufficiently large t and x_{t^*} is first-order stationary w.p.1.*

Proof. If Algorithm 1 results in a finite number of successful iterations, there exists t_1 so that for all $t \geq t_1$, iteration t fails. Consequently, $\rho_t < \eta_1$ and $\sigma_{t+1} \geq \gamma_1 \sigma_t$.

Necessarily, there exists a $t_2 \geq t_1$ such that $\sigma_t \geq \sigma_{\text{succ}}$ for all $t \geq t_2$.

If there existed $t \geq t_2$ such that $s_t \neq 0$, Theorem 1 would ensure that iteration t is successful, which contradicts our assumption. Therefore, $s_t = 0$ and $0 \in \operatorname{argmin}_s m_{t_2}(s; x_{t_2}, \sigma_{t_2})$.

Since $\mathcal{R}$ is prox-bounded, $\hat{\partial}\mathcal{R}$ closed and convex (Rockafellar and Wets, 1998, Propositions 8.6 and 8.46), and therefore, $-g_t \in \hat{\partial}\mathcal{R}(x_{t_2})$, for all $t \geq t_2$.

We now show that $-\nabla f(x_{t_2}) \in \hat{\partial}\mathcal{R}(x_{t_2})$. The empirical mean of the next m stochastic gradients satisfies

$$-\bar{g}_m = -\frac{1}{m} \sum_{i=t_2}^{t_2+m} g_i \in \hat{\partial}\mathcal{R}(x_{t_2}),$$

because $\hat{\partial}\mathcal{R}(x_{t_2})$ is convex.

According to the law of large numbers and Assumption 3,

$$\lim_{m\to\infty} \bar{g}_m = \mathbb{E}[g] = \nabla f(x_{t_2}) \quad \text{w.p.1.}$$

Because $\hat{\partial}\mathcal{R}(x_{t_2})$ is closed, $-\nabla f(x_{t_2}) \in \hat{\partial}\mathcal{R}(x_{t_2})$, i.e.,

$$0 \in \nabla f(x_{t_2}) + \hat{\partial}\mathcal{R}(x_{t_2}) = \hat{\partial}F(x_{t_2}),$$

and x_{t_2} is a first order stationary point w.p.1. □

We now focus on the case where SR2 generates infinitely many successes. By analogy with the deterministic and smooth case where $s_t = -\sigma_t^{-1}\nabla f(x_t)$, our criticality measure is $\mathbb{E}_{\hat{\xi}_t}[\|s^\xi\|^2] \leq \epsilon^2$, where $\mathbb{E}_{\hat{\xi}_t}$ denotes the expectation taken over the distribution of the ξ that yields a success knowing x_t. The first iteration that satisfies the latter condition is noted $t(\epsilon)$.

We start by studying the complexity of reaching this termination criteria. To that effect, let us define

$$\mathcal{S} = \{t \in \mathbb{N} \mid \rho_t \geq \eta_1\}, \tag{10}$$
$$\mathcal{S}(\epsilon) = \{t \in \mathcal{S} \mid t \leq t(\epsilon)\}, \tag{11}$$
$$\mathcal{U}(\epsilon) = \{t \in \mathbb{N} \mid t < t(\epsilon) \text{ and } \rho_t < \eta_1\}. \tag{12}$$

Lemma 1. *Let Assumptions 1, 3 and 4 hold. If Algorithm 1 generates an infinite number of successes and if there exists $F_{\text{low}} \in \mathbb{R}$ such that $F(x_t) \geq F_{\text{low}}$ for all $t \geq 0$, then for any $\epsilon \in (0,1)$, $|\mathcal{S}(\epsilon)| = \mathcal{O}(\epsilon^{-2})$.*

Proof.

When $t \in \mathcal{S}(\epsilon)$, $\rho_t \geq \eta_1$.

Using (9), the facts that $t < t(\epsilon)$ and $\sigma_{\min} \leq \sigma_t \leq \sigma_{\max}$, we have

$$F(x_t) - F(x_t + s_t) \geq \eta_1 \Delta\psi_t \geq \tfrac{1}{2}\eta_1\sigma_{\min}||s_t||^2.$$

This inequality holds for every s^ξ derived from ξ that yields a success at iteration t. We can therefore introduce the expectation over the distribution of the ξ that yield a success given x_t, denoted $\mathbb{E}_{\hat{\xi}_t}$. Therefore $\mathbb{E}_{\hat{\xi}_t}[F(x_t + s^\xi)]$ is a relevant quantity, and the previous inequality becomes

$$F(x_t) - \mathbb{E}_{\hat{\xi}_t}[F(x_t + s^\xi)] \geq \eta_1 \Delta\psi_t \geq \tfrac{1}{2}\eta_1\sigma_{\min}\mathbb{E}_{\hat{\xi}_t}[||s_t||^2]. \tag{13}$$

Because $t < t(\epsilon)$, $\mathbb{E}_{\hat{\xi}_t}[||s_t||^2] \geq \epsilon^2$. Thus, since $t \in \mathcal{S}$, (13) becomes

$$F(x_t) - \mathbb{E}_{\hat{\xi}_t}[F(x_t + s^\xi)] = F(x_t) - \mathbb{E}_{\hat{\xi}_t}[F(x_{t+1})] \geq \tfrac{1}{2}\eta_1\sigma_{\min}\epsilon^2. \tag{14}$$

By analogy with Bottou et al. (2018), we introduce the total expectation $\mathbb{E}[.]$ with respect to the joint distribution of all previous realization of ξ that yield a success, thus $\mathbb{E}[F(x_t)] := \mathbb{E}_{\hat{\xi}_1}\mathbb{E}_{\hat{\xi}_2}\ldots\mathbb{E}_{\hat{\xi}_{t-1}}[F(x_t)]$. Taking the total expectation in (14) yields

$$\mathbb{E}[F(x_t)] - \mathbb{E}[F(x_{t+1})] \geq \tfrac{1}{2}\eta_1\sigma_{\min}\epsilon^2. \tag{15}$$

Because $x_{t+1} = x_t$ if ξ_t yields $t \in \mathcal{U}$, while $x_{t+1} = x_t + s_t$ if ξ_t yields $t \in \mathcal{S}$,

$$F(x_1) - F(x_{\text{low}}) \geq \mathbb{E}[F(x_1)] - \mathbb{E}[F(x_{t(\epsilon)})]$$

$$
\begin{aligned}
&= \sum_{t=1}^{t(\epsilon)} \Big[\mathbb{E}[F(x_t)] - \mathbb{E}[F(x_{t+1})]\Big] \\
&\geq \sum_{t\in\mathcal{S}(\epsilon)} \Big[\mathbb{E}[F(x_t)] - \mathbb{E}[F(x_{t+1})]\Big] \\
&\geq \tfrac{1}{2}\eta_1\sigma_{\min}\epsilon^2|\mathcal{S}(\epsilon)|.
\end{aligned}
$$

Therefore, $|\mathcal{S}(\epsilon)| = \mathcal{O}(\epsilon^{-2})$. □

Lemma 2. *Under the assumptions of Lemma 1,* $|\mathcal{U}(\epsilon)| = \mathcal{O}(\epsilon^{-2})$.

Proof. Let $t \in \mathcal{U}(\epsilon)$, so that $t < t(\epsilon)$. Algorithm 1 increases σ_t by a factor of at least $\gamma_1 > 1$ if the step is rejected, and decreases σ_t by a factor of at most $\gamma_3 \in (0, 1]$ if it is accepted.

Thus, at iteration $t(\epsilon) - 1$, we have successively

$$
\begin{aligned}
& \sigma_{\max} \geq \sigma_{t(\epsilon)-1} \geq \sigma_0\gamma_1^{|\mathcal{U}(\epsilon)|}\gamma_3^{|\mathcal{S}(\epsilon)|} \\
\Rightarrow \quad & \frac{\sigma_{\max}}{\sigma_0} \geq \gamma_1^{|\mathcal{U}(\epsilon)|}\gamma_3^{|\mathcal{S}(\epsilon)|} \\
\Leftrightarrow \quad & \log(\frac{\sigma_{\max}}{\sigma_0}) \geq |\mathcal{U}(\epsilon)|\log(\gamma_1) + |\mathcal{S}(\epsilon)|\log(\gamma_3) \\
\Leftrightarrow \quad & |\mathcal{U}(\epsilon)|\log(\gamma_1) \leq \log(\frac{\sigma_{\max}}{\sigma_0}) - |\mathcal{S}(\epsilon)|\log(\gamma_3).
\end{aligned}
$$

Because $\log(\gamma_3) < 0$ and $|\mathcal{S}(\epsilon)| = \mathcal{O}(\epsilon^{-2})$, we obtain $|\mathcal{U}(\epsilon)| = \mathcal{O}(\epsilon^{-2})$. □

From $t(\epsilon) = |\mathcal{S}(\epsilon)| + |\mathcal{U}(\epsilon)|$, we deduce $t(\epsilon) = \mathcal{O}(\epsilon^{-2})$, and obtain the two following results.

Theorem 3. *Under the assumptions of Lemma 1, either F is unbounded from below or* $\liminf_{t\to\infty} \mathbb{E}_{\hat{\xi}_t}[||s^\xi||^2] = 0$

Theorem 4. *Let $0 < \epsilon < 1$. Then,* $\mathbb{E}_{\hat{\xi}_{t(\epsilon)}}[\mathrm{dist}(0, \hat{\partial}F(x_{t(\epsilon)} + s^\xi))^2] \leq C\epsilon^2 + 3\mathbb{E}_{\hat{\xi}_{t(\epsilon)}}[||\nabla f(x_{t(\epsilon)}) - g^\xi||^2]$, *where* $C = 3(L^2 + \sigma_{\max}^2)$.

Proof. From the definition of s_t, we have

$$
\begin{aligned}
0 &\in g_t + \sigma_t s_t + \hat{\partial}\mathcal{R}(x_t + s_t) \\
\Leftrightarrow -(g_t + \sigma_t s_t) &\in \hat{\partial}\mathcal{R}(x_t + s_t) \\
\Leftrightarrow \nabla f(x_t + s_t) - (g_t + \sigma_t s_t) &\in \hat{\partial}F(x_t + s_t).
\end{aligned}
$$

Thus

$$
\begin{aligned}
\mathrm{dist}(0, \hat{\partial}F(x_t + s_t))^2 &\leq \|\nabla f(x_t + s_t) - g_t - \sigma_t s_t\|^2 \\
&\leq 3\|\nabla f(x_t + s_t) - \nabla f(x_t)\|^2 + 3\|\sigma_t s_t\|^2 + 3\|\nabla f(x_t) - g_t\|^2 \\
&\leq 3L^2\|s_t\|^2 + 3\|\sigma_t s_t\|^2 + 3\|\nabla f(x_t) - g_t\|^2 \\
&\leq 3(L^2 + \sigma_{\max}^2)\|s_t\|^2 + 3\|\nabla f(x_t) - g_t\|^2,
\end{aligned}
$$

which is true for every step s^ξ computed with a realization of ξ, i.e.,

$$
\mathrm{dist}(0, \hat{\partial}F(x_t + s^\xi))^2 \leq 3(L^2 + \sigma_{\max}^2)\|s^\xi\|^2 + 3\|\nabla f(x_t) - g^\xi\|^2.
$$

Therefore

$$
\mathbb{E}_{\xi_{t(\epsilon)}}[\mathrm{dist}(0, \hat{\partial}F(x_t + s^\xi))^2] \leq 3(L^2 + \sigma_{\max}^2)\mathbb{E}_{\xi_{t(\epsilon)}}[\|s^\xi\|^2] + 3\mathbb{E}_{\xi_{t(\epsilon)}}[\|\nabla f(x_t) - g^\xi\|^2].
$$

For $t = t(\epsilon)$, we have $\mathbb{E}_{\xi_{t(\epsilon)}}[\|s^\xi\|^2] \leq \epsilon^2$. Thus

$$\mathbb{E}_{\xi_{t(\epsilon)}}[\text{dist}(0, \hat{\partial} F(x_t + s^\xi))^2] \leq 3(L^2 + \sigma_{\max}^2)\epsilon^2 + 3\mathbb{E}_{\xi_{t(\epsilon)}}[\|\nabla f(x_t) - g^\xi\|^2]. \qquad \square$$

Finally, we are set to analyze the properties of $x_{t(\epsilon)}$ in terms of stationarity. Shamir (2020) and Zhang et al. (2020) discuss the impossibility of finding ϵ-stationary points for nonsmooth and nonconvex functions with first-order methods in finite time. Instead, Zhang et al. (2020) introduce a relaxation of the concept of ϵ-stationarity, namely, (δ, ϵ)-stationarity which is reported in Definition 2.

Definition 2. A point x is called (δ, ϵ)-stationary if $d(0, \partial F(x + \delta B)) \leq \epsilon$, where $\partial F(x + \delta B) := conv\big(\cup_{y \in x + \delta B} \partial F(y)\big)$, and ∂F is the generalized gradient of F (Clarke, 1990).

We propose a variant of Definition 2 that is better adapted to our method. Note that other adaptations of the (δ, ϵ)-stationarity notion are discussed in Shamir (2020).

Definition 3. A point x is called $\widehat{(\delta, \epsilon)}$-stationary if $\mathbb{E}_{\hat{\xi}}\Big[d(0, \hat{\partial} F(x + s^\xi))^2\Big] \leq \epsilon^2$, with $\delta = \max_{\hat{\xi}} ||s^\xi||$.

Definition 3 appears in the result of Theorem 4, if a variance reduction strategy is additionally implemented to ensure the second right term becomes lower than ϵ^2. This remark is expressed in Corollary 1.

Corollary 1. *Let $0 < \epsilon < 1$. If a variance reduction strategy ensures $\mathbb{E}_{\hat{\xi}_{t(\epsilon)}}[||\nabla f(x_{t(\epsilon)}) - g^\xi||^2] \leq \epsilon^2$, then $x_{t(\epsilon)}$ is a $(\delta, \widehat{(C+3)}\epsilon)$ stationary point, with $C = 3(L^2 + \sigma_{\max}^2)$ and $\delta = \max_{\hat{\xi}} ||s^\xi||$.*

5 Experiments

We compare SR2 against ProxSGD and ProxGEN to train three DNNs on the CIFAR-10 and CIFAR-100 datasets. The networks considered are DenseNet-121, ResNet-34 and DenseNet-201, with 7.98M, 21.79M and 20M parameters respectively. Each set of tests uses $\mathcal{R} = \lambda\|\cdot\|_1, \lambda \in \{10^{-4}, 10^{-3}, 10^{-2}\}$, while $\mathcal{R} = \lambda\|\cdot\|_0$ is not tested with ProxSGD as it is not designed for nonconvex regularization.

We use the proximal SGD variant of ProxGEN. The implementation of ProxGEN was provided to us by its authors, and we also use their implementation of ProxSGD. Both methods use the hyperparameters mentioned in their respective papers and implementations. The implementation of SR2 is available at `https://github.com/DouniaLakhmiri/SR2` and its configurationis reported in Table 1. In our implementation, we compute ρ_t based on the sampled value of $F(., \xi_t)$ instead of the full objective.

Table 1: SR2 hyperparameters.

η_1	η_2	γ_1	γ_2	γ_3
$7.5 \cdot 10^{-4}$	0.99	5.56	2.95	0.8

For the sake of a fair comparison, we have disabled the momentum directions and preconditioners from ProxSGD and ProxGEN as well as the scheduled updates of the learning rate at epochs 150 and 250. These common accelerating strategies are not yet incorporated to SR2 and would give ProxSGD and ProxGEN an unfair advantage as shown in Figure 1.

Each test trains for 300 epochs after which we proceed to pruning each solution based on the criterion $|w_i| \leq \alpha$ with $\alpha = 10^{-k}$, $k = 1, \ldots, 8$, where w_i is the i-th weight in the network. We then compare the sparsity level and retained accuracy of the sparse networks without re-training.

5.1 Results on CIFAR-10

Table 2a reports the results with $\mathcal{R} = \lambda\|\cdot\|_1$. For both networks, we observe that SR2 combined with $\lambda = 10^{-4}$ achieves the highest accuracy overall, while ProxSGD gets the highest accuracies with

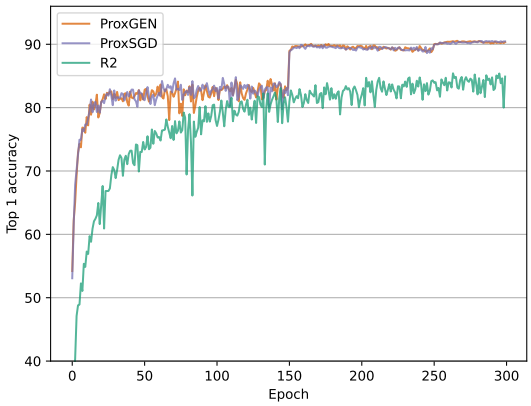

Figure 1: Accuracy of ProxGEN and ProxSGD with momentum, preconditioner, and learning rate updates at epochs 150 **and** 250**.**

$\lambda = 10^{-3}, 10^{-2}$. Table 2a also reports information on the magnitudes of the weights in each solution. Interestingly, SR2 has a consistent tendency to set a large portion of the network's weights to exactly 0 while ProxSGD does the opposite and ProxGEN falls in between in this regard. This observation highlights the clear difference between ProxSGD and ProxGEN in the solutions each method finds. In addition, SR2 identifies a larger proportion of small weights than ProxSGD and ProxGEN.

Table 2: Results of training DenseNet-121 and ResNet-34 on CIFAR-10.

(a) with $\mathcal{R} = \lambda\|\cdot\|_1$**.**

Net.	λ	Optim.	Acc.	$\%\|w\|=0$	$\%\|w\|\leq 10^{-3}$
D-121	10^{-4}	ProxSGD	72.20%	0.00%	20.22%
		ProxGEN	72.26%	6.50%	22.71%
		SR2	**84.69%**	**79.47%**	**92.15%**
	10^{-3}	ProxSGD	**77.43%**	0.00%	82.18%
		ProxGEN	76.81%	44.21%	94.40%
		SR2	68.26%	**95.43%**	**98.17%**
	10^{-2}	ProxSGD	**78.36%**	0.00%	94.16%
		ProxGEN	59.69%	**98.03%**	**99.13%**
		SR2	76.49%	78.11%	98.59%
R-34	10^{-4}	ProxSGD	85.12%	0.00%	64.70%
		ProxGEN	85.98%	1.02%	72.94%
		SR2	**93.94%**	**56.18%**	**98.12%**
	10^{-3}	ProxSGD	**89.67%**	0.00%	94.85%
		ProxGEN	83.27%	**73.34%**	99.25%
		SR2	88.46%	67.97%	**99.50%**
	10^{-2}	ProxSGD	**88.12%**	0.00%	98.28%
		ProxGEN	35.56%	**99.34%**	**99.92%**
		SR2	29.33%	62.17%	91.84%

(b) with $\mathcal{R} = \lambda\|\cdot\|_0$**.**

Net.	λ	Optim.	Acc.	$\%\|w\|=0$	$\%\|w\|\leq 10^{-3}$
D-121	10^{-4}	ProxGEN	71.09%	2.39%	3.27%
		SR2	**80.29%**	**14.67%**	**14.74%**
	10^{-3}	ProxGEN	70.44%	4.13%	4.13%
		SR2	**79.11%**	**23.82%**	**25.85%**
	10^{-2}	ProxGEN	71.03%	9.63%	10.05%
		SR2	**79.79%**	**94.99%**	**95.06%**
R-34	10^{-4}	ProxGEN	86.87%	5.48%	7.13%
		SR2	**90.59%**	**93.43%**	**93.43%**
	10^{-3}	ProxGEN	85.81%	11.04%	11.07%
		SR2	**92.20%**	**94.41%**	**94.42%**
	10^{-2}	ProxGEN	86.40%	28.85%	28.86%
		SR2	**87.82%**	**99.04%**	**99.07%**

Figure 2 (top) reports accuracy and sparsity results on pruned DenseNet-121 with $\mathcal{R} = \lambda\|\cdot\|_1$. The top plot shows that most configurations retain full accuracy until $\alpha = 10^{-3}$ or 10^{-2}, except for the one trained with ProxSGD, which shows a small drop at $\alpha = 10^{-3}$. The accuracy of all networks drops to 10% for $\alpha = 10^{-1}$. The plot at the top right shows the sparsity ratio with each pruning criteria. Overall, the combination of SR2 with $\lambda = 10^{-4}$ and $\alpha = 10^{-2}$ has the highest accuracy with a high sparsity level of 97.5%, followed by ProxGEN with $\lambda = 10^{-3}$ and $\alpha = 10^{-3}$ and a sparsity of 94.4%.

Figure 3 (top) shows that ResNet-34 retains full accuracy with $\alpha = 10^{-3}$ in most cases. The best combination is obtained with SR2, $\lambda = 10^{-4}$ and $\alpha = 10^{-3}$ that results in an accuracy of 93.32% and a sparsity ratio of 98.12%, followed by ProxGEN with $\lambda = 10^{-3}$ and $\alpha = 10^{-3}$ with an accuracy of 86.93% and a sparsity of 99.50%.

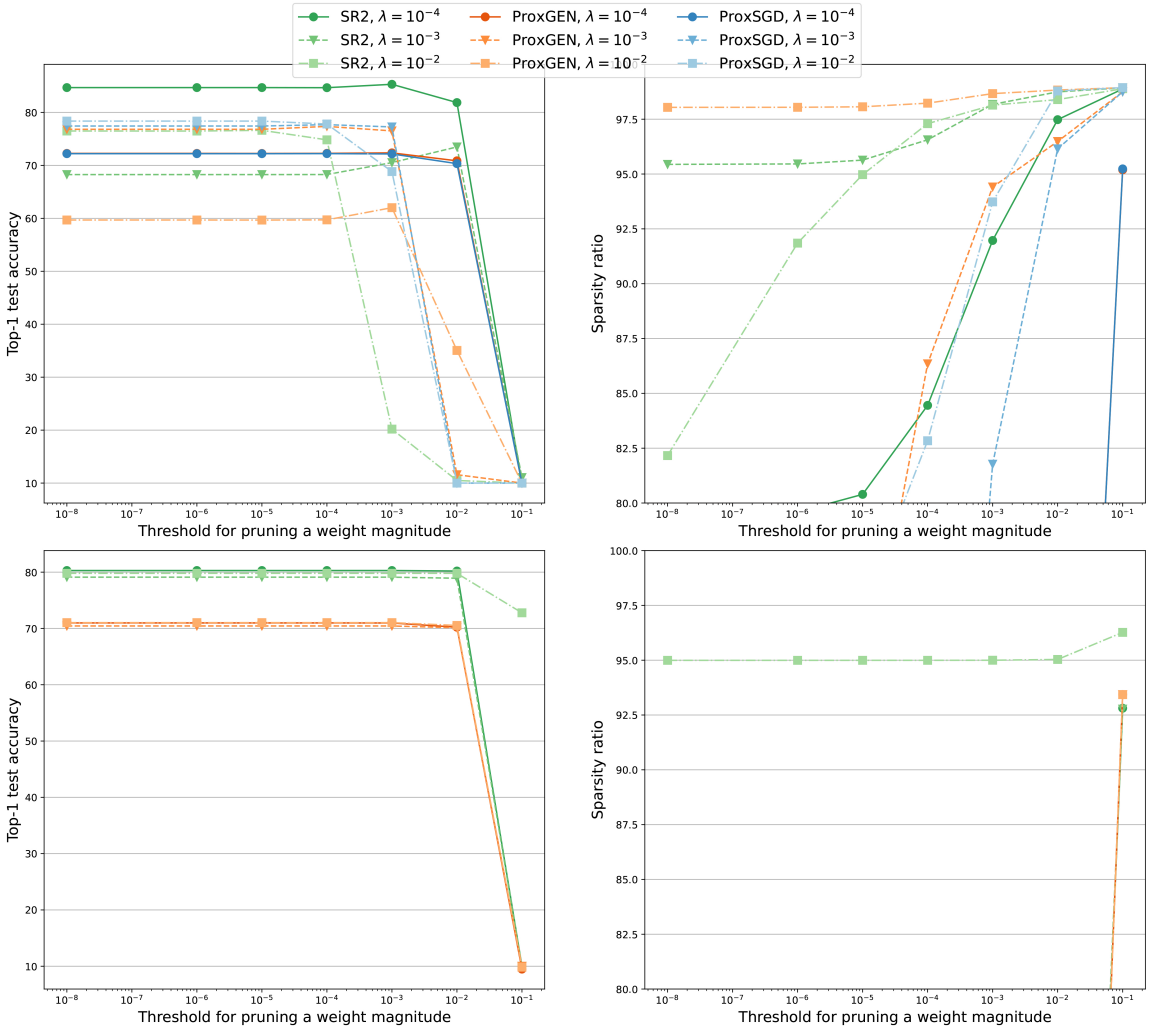

Figure 2: Accuracy and sparsity ratio of pruned DenseNet-121 on CIFAR-10 with $\mathcal{R} = \lambda\|\cdot\|_1$ (top) and $\mathcal{R} = \lambda\|\cdot\|_0$ (bottom).

Table 2b and Figure 2 (bottom) report results on the same networks with $\mathcal{R} = \|\cdot\|_0$ and compares SR2 against ProxGEN only, since ProxSGD does not handle nonconvex regularizations. The results show a clear advantage of SR2 both in terms of final accuracy and sparsity ratios. Compared to $\mathcal{R} = \|\cdot\|_1$, using $\mathcal{R} = \|\cdot\|_0$ allows SR2 to reach higher accuracies overall at the expense of higher weight magnitudes. The results seem to suggest that the value of λ needs a special adjustment for each regularizer. Figure 3 (bottom) summarizes the retained accuracy after pruning and the equivalent sparsities for ResNet-34. It is clear that SR2 generates the better solutions with the highest sparsity levels while retaining most of the full accuracies. A similar figure for DenseNet-121 is reported in the appendix.

5.2 Results on CIFAR-100

In this section, SR2 is compared against ProxSGD and ProxGEN on a more challenging dataset. We train DenseNet-201 on CIFAR-100 with $\mathcal{R} = \|\cdot\|_1$ and $\|\cdot\|_0$ and compare each solution's resulting accuracy and sparsity. Once again, our goal is to extract sparse substructures, and we do not focus our resources on tuning each method to reach high test accuracies.

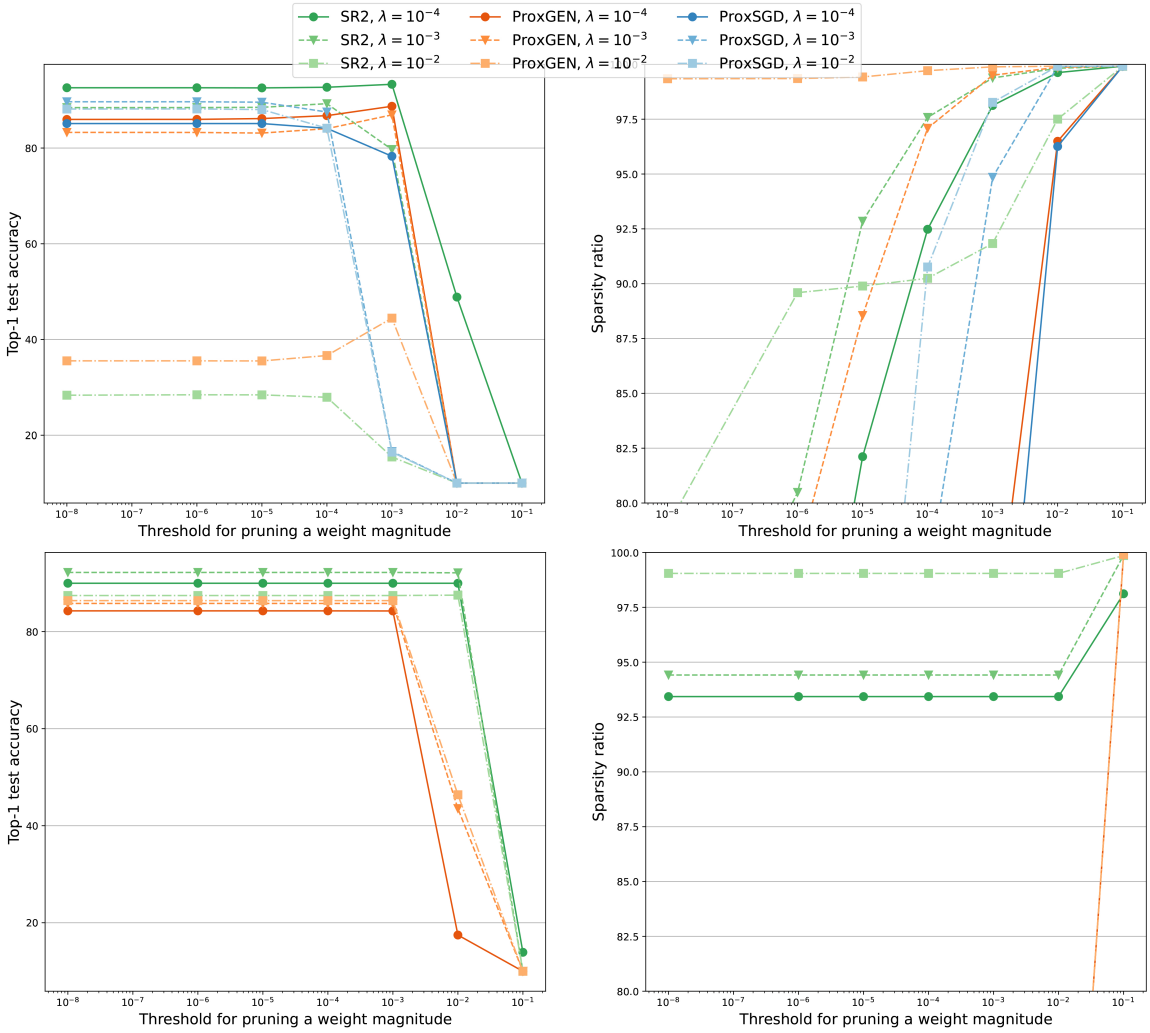

Figure 3: Accuracy and sparsity ratio of ResNet-34 trained on CIFAR-10 with $\mathcal{R} = \lambda\|\cdot\|_1$ (top) and $\mathcal{R} = \lambda\|\cdot\|_0$ (bottom).

Table 3a summarizes the relevant scores of each solution with $\mathcal{R} = \lambda\|\cdot\|_1$, and Figure 4 (top) illustrates the retained accuracy and equivalent sparsity ratio after each pruning. SR2 with $\lambda = 10^{-4}, \alpha = 10^{-2}$ obtains the highest accuracy of 58.50% after removing 97.57% of the weights from the original network. Other solvers that obtain a higher sparsity after pruning do so at the expense of the final accuracy of the network.

Table 3: Results of DenseNet-201 on CIFAR-100.

(a) with $\mathcal{R} = \lambda\|\cdot\|_1$.

λ	Optim.	Acc.	$\%\|w\| = 0$	$\%\|w\| \leq 10^{-3}$
10^{-4}	ProxSGD	42.38%	0.00%	25.23%
	ProxGEN	41.11%	2.95%	4.01%
	SR2	**57.63%**	**59.74%**	**92.65%**
10^{-3}	ProxSGD	42.38%	0.00%	72.47%
	ProxGEN	**46.70%**	48.94%	96.99%
	SR2	33.04%	**97.21%**	**98.92%**
10^{-2}	ProxSGD	**42.86%**	0.00%	25.21%
	ProxGEN	6.96%	98.83%	99.48%
	SR2	7.31%	98.30%	99.60%

(b) with $\mathcal{R} = \lambda\|\cdot\|_0$.

λ	Optim.	Acc.	$\%\|w\| = 0$	$\%\|w\| \leq 10^{-3}$
10^{-4}	ProxGEN	40.77%	2.97%	4.02%
	SR2	**48.91%**	**22.69%**	**22.78%**
10^{-3}	ProxGEN	40.44%	5.11%	5.60%
	SR2	**49.28%**	**39.82%**	**39.92%**
10^{-2}	ProxGEN	39.91%	11.84%	12.33%
	SR2	1.00%	98.91%	99.50%

Similarly Table 3b and Figure 4 (bottom) show that SR2 obtains the best accuracy when the network is trained with $\mathcal{R} = \lambda = 10^{-3}\|\cdot\|_0$ allows to consistently reach higher sparsity ratios while maintaining at least the accuracy of the full network . The best solution is found with $\lambda = 10^{-3}$ and $\alpha = 10^{-2}$ as shown in Figure 4 (bottom).

Overall, the results on CIFAR-100 are more contrasted than on CIFAR-10 with examples of ProxGEN and SR2 converging in some settings towards solutions with low accuracy. This suggests the need for a better tuning of the methods.

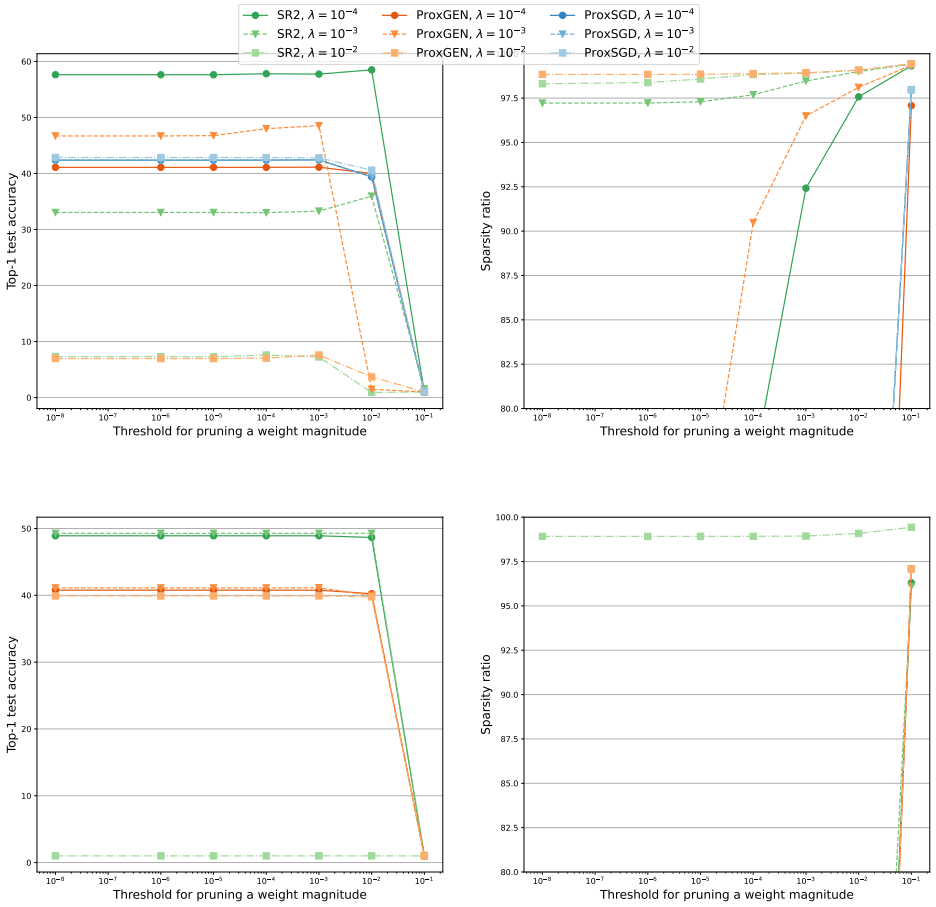

Figure 4: Accuracy and sparsity ratio of pruned DenseNet-201 on CIFAR-100 with $\mathcal{R} = \lambda\|\cdot\|_1$ (top) and $\mathcal{R} = \lambda\|\cdot\|_0$ (bottom).

6 Conclusion

SR2 is a new stochastic proximal method for training DNNs with nonsmooth, potentially nonconvex regularizers. SR2 relies on an adaptive quadratic regularization framework that does not automatically accept every step during the training to ensure a decrease in the objective. We establish the convergence of a first-order stationarity measure to zero with a $\mathcal{O}(\epsilon^{-2})$ worst-case iteration complexity. Our numerical experiments show that SR2 consistently produces solutions that achieve high accuracy and sparsity levels after an unstructured pruning. Ongoing research is focusing on incorporating a momentum term, a preconditioner, and second-order information to accelerate the convergence and attain higher accuracy.

References

Aleksandr Y Aravkin, Robert Baraldi, and Dominique Orban. A proximal quasi-newton trust-region method for nonsmooth regularized optimization. arXiv preprint arXiv:2103.15993, page to appear in SIAM Optimization, 2021.

Yu Bai, Yu-Xiang Wang, and Edo Liberty. ProxQuant: Quantized neural networks via proximal operators. In International Conference on Learning Representations, 2018.

Amir Beck. First-order methods in optimization. MOS-SIAM. SIAM, 2017.

Raghu Bollapragada, Richard Byrd, and Jorge Nocedal. Adaptive sampling strategies for stochastic optimization. SIAM Journal on Optimization, 28(4):3312–3343, 2018.

J. Bolte, S. Sabach, and M. Teboulle. Proximal alternating linearized minimization for nonconvex and nonsmooth problems. Mathematical Programming, 146:459–494, 2014. doi: 10.1007/s10107-013-0701-9.

Léon Bottou, Frank E. Curtis, and Jorge Nocedal. Optimization methods for large-scale machine learning. SIAM Review, 60:223–311, 2018.

Frank H Clarke. Optimization and nonsmooth analysis. SIAM, 1990.

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

Masao Fukushima and Hisashi Mine. A generalized proximal point algorithm for certain non-convex minimization problems. International Journal of Systems Science, 12(8):989–1000, 1981. doi: 10.1080/00207728108963798.

Torsten Hoefler, Dan Alistarh, Tal Ben-Nun, Nikoli Dryden, and Alexandra Peste. Sparsity in deep learning: Pruning and growth for efficient inference and training in neural networks, 2021.

Hamed Karimi, Julie Nutini, and Mark Schmidt. Linear convergence of gradient and proximal-gradient methods under the Polyak-Lojasiewicz condition. In Joint European Conference on Machine Learning and Knowledge Discovery in Databases, pages 795–811. Springer, 2016.

Jack Kiefer and Jacob Wolfowitz. Stochastic estimation of the maximum of a regression function. The Annals of Mathematical Statistics, pages 462–466, 1952.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7–9, 2015, Conference Track Proceedings, 2015.

Anders Krogh and John A Hertz. A simple weight decay can improve generalization. In Advances in neural information processing systems, pages 950–957, 1992.

S. Lotfi, T. Bonniot de Ruisselet, D. Orban, and A. Lodi. Stochastic damped L-BFGS with controlled norm of the Hessian approximation. In OPT2020 Conference on Optimization for Machine Learning, 2020. doi: 10.13140/RG.2.2.27851.41765/1.

Sanae Lotfi, Tiphaine Bonniot de Ruisselet, Dominique Orban, and Andrea Lodi. Adaptive first-and second-order algorithms for large-scale machine learning. arXiv preprint arXiv:2111.14761, 2021.

Lam M Nguyen, Jie Liu, Katya Scheinberg, and Martin Takáč. Sarah: A novel method for machine learning problems using stochastic recursive gradient. In International Conference on Machine Learning, pages 2613–2621. PMLR, 2017.

Nhan H Pham, Lam M Nguyen, Dzung T Phan, and Quoc Tran-Dinh. ProxSARAH: An efficient algorithmic framework for stochastic composite nonconvex optimization. Journal of Machine Learning Research, 21: 110–1, 2020.

Herbert Robbins and Sutton Monro. A stochastic approximation method. The annals of mathematical statistics, pages 400–407, 1951.

R. Tyrrell Rockafellar and Roger J. B. Wets. Variational Analysis, volume 317. Springer Berlin Heidelberg, 1998.

Sebastian Ruder. An overview of gradient descent optimization algorithms. arXiv preprint arXiv:1609.04747, 2016.

Ohad Shamir. Can we find near-approximately-stationary points of nonsmooth nonconvex functions? arXiv preprint arXiv:2002.11962, 2020.

Marc Teboulle. Convergence of proximal-like algorithms. SIAM Journal on Optimization, 7(4):1069–1083, 1997.

Huan Wang, Xinyi Hu, Qiming Zhang, Yuehai Wang, Lu Yu, and Haoji Hu. Structured pruning for efficient convolutional neural networks via incremental regularization. IEEE Journal of Selected Topics in Signal Processing, 14(4):775–788, 2019.

Matthias Wess, Sai Manoj Pudukotai Dinakarrao, and Axel Jantsch. Weighted quantization-regularization in DNNs for weight memory minimization toward hw implementation. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 37(11):2929–2939, 2018.

Yi Xu, Rong Jin, and Tianbao Yang. Non-asymptotic analysis of stochastic methods for non-smooth non-convex regularized problems. arXiv preprint arxiv:1902.07672, 2019.

Chen Yang, Zhenghong Yang, Abdul Mateen Khattak, Liu Yang, Wenxin Zhang, Wanlin Gao, and Minjuan Wang. Structured pruning of convolutional neural networks via l1 regularization. IEEE Access, 7:106385–106394, 2019a.

Yang Yang, Yaxiong Yuan, Avraam Chatzimichailidis, Ruud JG van Sloun, Lei Lei, and Symeon Chatzinotas. ProxSGD: Training structured neural networks under regularization and constraints. In International Conference on Learning Representations, 2019b.

Jihun Yun, Aurelie Lozano, and Eunho Yang. Adaptive proximal gradient methods for structured neural networks. In Thirty-Fifth Conference on Neural Information Processing Systems, 2021.

J. Zhang, Hongzhou Lin, Suvrit Sra, and Ali Jadbabaie. On complexity of finding stationary points of nonsmooth nonconvex functions. ArXiv, abs/2002.04130, 2020.

Yucong Zhou, Yunxiao Sun, and Zhao Zhong. FixNorm: Dissecting Weight Decay for Training Deep Neural Networks. arXiv preprint arXiv:2103.15345, 2021.