Training Structured Neural Networks Through Manifold Identification and Variance Reduction

Anonymous authors
Paper under double-blind review

Abstract

This paper proposes an algorithm for training neural networks with a regularization term for promoting desired structures in the model. Our method does not incur computation additional to SGD with momentum and achieves variance reduction without requiring the objective function to be of the finite-sum form. Through the powerful tool of manifold identification from nonlinear optimization, we rigorously prove that the proposed method is able to correctly identify within finite steps the underlying structure, which can be represented as a manifold, of the neural network induced by the regularizer, even in the presence of engineering tricks like data augmentation and dropout that complicate the training process. Experiments on training neural networks with structured sparsity confirm that variance reduction is necessary for finding the desired structure and show that our method therefore significantly outperforms existing methods for this task in terms of both the structured sparsity level and the prediction accuracy.

1 Introduction

Training neural networks with regularization to obtain a certain desired structure such as structured sparsity or discrete-valued parameters is a problem of increasing interest. Existing approaches either use the subgradient of the regularized objective (Wen et al., 2016; 2018) or combine popular algorithms for training neural networks, such as SGD (with momentum), Adam (Kingma & Ba, 2015) with the proximal operator associated with the regularizer to conduct proximal stochastic gradient updates to obtain a model with preferred structures (Bai et al., 2018a; Yang et al., 2019; Yun et al., 2020; Deleu & Bengio, 2021). Such methods come with proven convergence for certain measures of first-order optimality and have shown some empirical success in applications. However, we notice that an essential theoretical support lacking in existing algorithms is the ability for identifying the desired structure. More specifically, those imposed regularization are only able to induce the desired structure at points that satisfy the first-order optimality condition of the underlying optimization problem, but training algorithms are only able to generate approximate solutions. Without further theoretical guarantees, it is unknown whether the approximation solutions generated by existing works can successfully identify the same structure as the points of stationarity. This work aims at filling this gap to propose an algorithm for training structured neural networks that can provably identify the optimal structure even if the output is only an approximate solution to the training problem.

It is well-known in the literature of nonlinear optimization that such an optimal structure can be identified if it can be represented as a manifold that contains a point of stationarity. This is often termed as the active manifold, and the task of finding this active manifold in finite steps is called manifold identification (Lewis, 2002; Hare & Lewis, 2004; Lewis & Zhang, 2013). An algorithm identifies such an active manifold if after a certain number of iterations, all following iterates all lie in this manifold, and in this case we can terminate the algorithm whenever the validation accuracy or other criteria has stabilized without worrying whether the structure found is correct. To identify the active manifold, we need the regularizer to be partly smooth (Lewis, 2002; Hare & Lewis, 2004; Lewis & Zhang, 2013), roughly meaning

that it is smooth along the active manifold around a stationary point, while the change in its value is drastic along directions leaving the manifold, so iterates generated by suitable algorithms will tend to jump into and stay within this manifold. A more technical definition will be given in Section 3. Fortunately, most regularizers widely used in machine learning are partly smooth, so active manifold, or optimal structure, identification is possible, and various deterministic algorithms are known to achieve so (Hare & Lewis, 2007; Hare, 2011; Wright, 2012; Liang et al., 2017a;b; 2018; Iutzeler & Malick, 2020; Li et al., 2020; Lee, 2020; Bareilles et al., 2020).

On the other hand, for stochastic algorithms, it is well-observed that stochastic gradient descent (SGD) is incapable of manifold identification due to the presence of the variance in the gradient estimation (Lee & Wright, 2012; Sun et al., 2019).[1] For stochastic methods to identify the active manifold, existing theory requires the variance in gradient estimation to vanish as the iterates approach a stationary point (Poon et al., 2018). Therefore, variance-reduction methods such as SVRG (Johnson & Zhang, 2013; Xiao & Zhang, 2014) or SAGA (Defazio et al., 2014) that utilize the finite-sum structure of empirical risk minimization (ERM) to drive the variance of their unbiased estimators for the gradient to zero are suitable for this task. Unfortunately, with the standard practice of data augmentation (see, for example, Shorten & Khoshgoftaar, 2019) and dropout (Srivastava et al., 2014) in deep learning, the training of deep learning models with a regularization should be treated as the following stochastic optimization problem that minimizes the expectation over a distribution, instead of the commonly seen ERM form:

$$\min_{W \in \mathcal{E}} \quad F(W) \coloneqq \mathbb{E}_{\xi \sim \mathcal{D}} \left[f_\xi(W) \right] + \psi(W), \tag{1}$$

where $\mathcal{E}$ is a Euclidean space with its inner product $\langle \cdot, \cdot \rangle$ and the associated norm $\|\cdot\|$, $\mathcal{D}$ is a distribution over a space Ω, f_ξ is differentiable almost everywhere for all $\xi \in \Omega$, and $\psi(W)$ is a regularizer that might be nondifferentiable. We will also use the notation $f(W) \coloneqq \mathbb{E}_{\xi \sim \mathcal{D}}[f_\xi(W)]$. Without a finite-sum structure in (1), Defazio & Bottou (2019) pointed out that classical variance reduction is known to be ineffective for deep learning, and one major reason is the necessity of periodically evaluating $\nabla f(W)$ (or at least using a large batch from $\mathcal{D}$ to get a precise approximation of it) in variance-reduction methods is intractable, hence manifold identification and therefore finding the optimal structure becomes an extremely tough task for deep learning. Although recently there are efforts in developing variance-reduction methods for (1) inspired by online/streaming problems (Allen-Zhu, 2018; Fang et al., 2018; Wang et al., 2019; Nguyen et al., 2021; Pham et al., 2020; Cutkosky & Orabona, 2019; Tran-Dinh et al., 2019), these methods all have multiple hyperparameters to tune and have computational cost at least twice or thrice to that of (proximal) SGD. As the training of deep learning models is time- and resource-consuming, these drawbacks make such methods less ideal for deep learning.

To tackle these difficulties, in this work, we extend the recently proposed modernized dual averaging framework (Jelassi & Defazio, 2020) to the regularized setting and obtain a new algorithm RMDA (Regularized Modernized Dual Averaging) for (1). The proposed algorithm provably achieves variance reduction beyond finite-sum problems without any computational cost or hard-to-tune hyperparameters additional to those of proximal momentum SGD, and we further provide theoretical guarantees for its convergence and ability in manifold identification. The key difference between RMDA and the original regularized dual averaging proposed by Xiao (2010) is that RMDA incorporates momentum and can achieve better performance in terms of the generalization ability for deep learning, as shown in Jelassi & Defazio (2020); Defazio & Jelassi (2021), and the new algorithm requires nontrivial proofs for its convergence and manifold identification. We further conduct experiments on training deep learning models with a regularizer for structured-sparsity to demonstrate the ability of RMDA for identifying the optimal structure without sacrificing the prediction accuracy.

The main contributions of this work are summarized as follows.

[1] An exception is the interpolation case, in which the variance of plain SGD vanishes asymptotically. But data augmentation and dropout often fails this interpolation assumption.

- ***Principled analysis***: We use the theory of partly smooth functions and manifold identification from nonlinear optimization to provide a principled way towards understanding algorithms for training structured neural networks.
- ***Variance reduction beyond finite-sum with low cost***: RMDA achieves variance reduction for problems that consist of an infinite-sum expectation term plus a regularization while incorporating momentum to improve the generalization performance. In comparison with existing variance reduction methods that require evaluating two to three gradient per iteration and storing previous iterates, the spatial and computational cost of RMDA is the same as that of SGD, making it suitable for large-scale problems like deep learning. Moreover, there are no more hyperparameters to tune in RMDA than SGD.
- ***Guaranteed structure identification***: With the help of variance reduction, our theory shows RMDA is guaranteed to identify the active manifold in a finite number of iterations.
- ***Superior empirical performance***: Experiments on neural networks with structured sparsity show that RMDA identifies the optimal structure without a drop in the validation accuracy, thus outperforming existing methods by achieving higher group sparsity levels.

2 Algorithm

We provide details of the proposed RMDA in Algorithm 1. Starting with an initial point W^0, at the t-th iteration, we draw an independent sample $\xi_t \sim \mathcal{D}$ to compute the stochastic gradient $\nabla f_{\xi_t}(W^{t-1})$, decide a learning rate η_t, and update the weighted sum V_t of previous stochastic gradients using η_t and the scaling factor $\beta_t := \sqrt{t}$:

$$V_t := \sum_{k=1}^{t} \eta_k \beta_k \nabla f_{\xi_k}(W^{k-1}) = \begin{cases} V_{t-1} + \eta_t \beta_t \nabla f_{\xi_t}(W^{t-1}) & \text{if } t > 0, \\ 0 & \text{else.} \end{cases}$$

The tentative iterate $\tilde{W}^t$ is then obtained by the proximal operation associated with ψ:

$$\tilde{W}^t := \operatorname{prox}_{\frac{\alpha_t}{\beta_t}\psi}\left(W^0 - \frac{V^t}{\beta_t}\right), \quad \alpha_t := \sum_{k=1}^{t} \beta_k \eta_k, \tag{2}$$

where for any function g, the proximal operator prox_g is defined as

$$\operatorname{prox}_g(x) := \arg\min_y \quad \frac{1}{2}\|x - y\|^2 + g(y).$$

The iterate is then updated along the direction $\tilde{W}^t - W^{t-1}$ with a factor of $c_t \in [0, 1]$ by

$$W^t = (1 - c_t)\, W^{t-1} + c_t \tilde{W}^t = W^{t-1} + c_t \left(\tilde{W}^t - W^{t-1}\right). \tag{3}$$

When $\psi \equiv 0$, RMDA without regularization reduces to the modernized dual averaging algorithm of Jelassi & Defazio (2020), and in this case, it has been shown that the mixing of W^{t-1} and $\tilde{W}^t$ can be interpreted as introducing momentum (Jelassi & Defazio, 2020; Sebbouh et al., 2020; Tao et al., 2018). We found that this introduction of momentum greatly improves the performance of RMDA and is therefore essential for applying it on deep learning problems.

3 Analysis

We provide theoretical analysis of the proposed RMDA in this section. Our analysis shows variance reduction in RMDA and the stationarity of the limit point, but all of them revolves around our main purpose of identification of the desired structure within a finite number of iterations. The key tools for this end are partial smoothness and manifold identification (Hare & Lewis, 2004; Lewis, 2002). Our result is the currently missing cornerstone for those proximal algorithms applied to deep learning problems for identifying desirable structures. In stark contrast to our results, it is actually well-known in convex optimization that those algorithms based on proximal stochastic gradient are *unable to identify the manifold within finite iterations*. Our work is therefore the first one to provide justification for solving the regularized optimization problem in deep learning to really identify the desired solution structure.

Algorithm 1: RMDA $(W^0, T, \eta(\cdot), c(\cdot))$

input : Initial point W^0, learning rate schedule $\eta(\cdot)$, momentum schedule function $c(\cdot)$, number of epochs T

$V_0 \leftarrow 0, \quad \alpha_0 \leftarrow 0$

for $t = 1, \ldots, T$ **do**

  $\beta_t \leftarrow \sqrt{t}, \quad s_t \leftarrow \eta(t)\beta_t, \quad \alpha_t \leftarrow \alpha_{t-1} + s_t$

  Sample $\xi_t \sim \mathcal{D}$ and compute $G^t \leftarrow \nabla f_{\xi_t}(W^{t-1})$

  $V^t \leftarrow V^{t-1} + s_t G^t$

  $\tilde{W}^t \leftarrow \arg\min_W \langle V^t, W\rangle + \frac{\beta_t}{2}\left\|W - W^0\right\|^2 + \alpha_t \psi(W)$ // (2)

  $W^t \leftarrow (1 - c(t))W^{t-1} + c(t)\tilde{W}^t$

output: The final model W^T

3.1 Preliminaries

We start from introducing the notion of partial smoothness.

Definition 1 (Partly smooth)**.** *A function ψ is partly smooth at a point x^* relative to a set $\mathcal{M}$ containing W^* if*
1. Around W^*, $\mathcal{M}$ is a $\mathcal{C}^2$-manifold and $\psi|_{\mathcal{M}}$ is $\mathcal{C}^2$.
2. ψ is regular at all points $W \in \mathcal{M}$ around W^*, with $\partial\psi(W) \neq \emptyset$.
3. The affine span of $\partial\psi(W^*)$ is a translate of the normal space to $\mathcal{M}$ at W^*.
4. $\partial\psi$ is continuous at W^* relative to $\mathcal{M}$.

We often call $\mathcal{M}$ the active manifold. A function is regular at a point if it is finite and its Fréchet subdifferential coincides with the limiting Fréchet subdifferential there. Another concept required for manifold identification is prox-regularity (Poliquin & Rockafellar, 1996).

Definition 2 (Prox-regular)**.** *A function ψ is prox-regular at a point W^* for a subgradient $V^* \in \partial\psi(W^*)$ if ψ is finite at W^*, locally lower semi-continuous around W^*, and there exists $\rho > 0$ such that $\psi(W_1) \geq \psi(W_2) + \langle V, W_1 - W_2\rangle - \frac{\rho}{2}\|W_1 - W_2\|^2$ whenever W_1, W_2 are close to W^* with $\psi(W_2)$ near $\psi(W^*)$ and $V \in \partial\psi(W_2)$ near V^*.*

To broaden the applicable range, a prox-regular ψ is often also assumed to be subdifferentially continuous (Poliquin & Rockafellar, 1996), meaning that if $W^t \to W^\infty$ for some W^∞ and there are $U^\infty \in \partial\psi(W^\infty)$ and a sequence $\{U^t\}$ with $U^t \in \partial\psi(W^t)$ and $U^t \to U^\infty$, then $\psi(W^t) \to \psi(W^\infty)$. Notably, all convex and weakly-convex (Nurminskii, 1973) functions are regular, prox-regular, and subdifferentially continuous in their domain.

For our analysis of RMDA, we will make the following mild assumptions.

Assumption 1. *For any $\xi \sim \mathcal{D}$, f_ξ is L-Lipschitz-continuously-differentiable almost surely for some L, and consequently the expectation f is also L-Lipschitz-continuously-differentiable. The set of stationary points $\mathcal{Z} := \{W \mid 0 \in \partial F(W)\}$ is nonempty. Moreover, there is a constant $C \geq 0$ such that*

$$\mathbb{E}_{\xi_t \sim \mathcal{D}}\left\|\nabla f_{\xi_t}\left(W^{t-1}\right)\right\|^2 \leq C, \quad \forall t. \tag{4}$$

The bounded variance assumption is standard in analyses of stochastic algorithms.

3.2 Theoretical Results

We first show that RMDA achieves variance reduction using a classical result in stochastic optimization that can be found at, for example, (Gupal, 1979, Theorem 4.1, Chapter 2.4).

Lemma 1. *Consider Algorithm 1 under Assumption 1. If the sequence $\{\eta_t\}$ satisfies*

$$\sum \frac{\beta_t \eta_t}{\alpha_t} = \infty, \quad \sum \left(\frac{\beta_t \eta_t}{\alpha_t}\right)^2 < \infty, \quad \frac{\left\|W^{t+1} - W^t\right\|}{\frac{\beta_t \eta_t}{\alpha_t}} \xrightarrow{a.s.} 0, \tag{5}$$

then $\alpha_t^{-1} V^t \longrightarrow \nabla f(W^t)$ almost surely (or equivalently, with probability one).

In general, the last condition in (5) requires some regularity condition in F to control the speed of change in W^t. One possibility is when each stochastic gradient has a bounded norm and ψ is the indicator function of a certain convex set, then $\beta_t \eta_t \propto t^p$ for $t \in (1/2, 1)$ will satisfy this condition. However, in other settings for η_t, even when F and ψ are both convex, existing analyses for the classical regularized dual averaging, such that $c_t \equiv 1$ in Algorithm 1, such as Lee & Wright (2012); Duchi & Ruan (2021) still need an additional local error bound assumption to control the change of $W^{t+1} - W^t$. Hence, to stay focused on our main message, we take this assumption for granted, and leave suitable requirements for ensuring it such as the Kurdyka-Łojasiewicz condition as future work.

A requirement for manifold identification is surely that the iterates converge to a certain point, as partial smoothness, prox-regularity, and subdifferential continuity are all local properties and utilization of them without getting close enough to the region of interest is impossible. Therefore, we will assume that $\{\tilde{W}^t\}$ converges almost surely to a certain point W^*. Guaranteeing so also requires certain regularity conditions such as specific scheduling of the learning rate or restriction to a smaller problem class (like those satisfying the Kurdyka-Łojasiewicz condition), and usually such guarantees will further ensure certain convergence speed, but these are not the purpose of this work, so we avoid distracting the readers from such technical results. We also observe that often the iterates indeed converge to a certain point in practice. Under this assumption, we first show that $\{W^t\}$ also converges to the same point W^*.

Lemma 2. *Consider Algorithm* 1 *with* $\{c_t\}$ *satisfying*

$$\sum_{t=0}^{\infty} c_t = \infty. \tag{6}$$

If $\{\tilde{W}^t\}$ *converges to a point* W^* *almost surely, then so does* $\{W^t\}$.

Note that Lemma 2 has only a minimum requirement and does not need Assumption 1.

With the help of Lemmas 1 and 2, we can show that the point of convergence is stationary.

Theorem 1. *Consider Algorithm* 1 *with Assumption* 1 *and* (5)–(6) *hold, and assume that* $\{\beta_t\}$, *and* $\{\eta_t\}$ *satisfy*

$$\beta_t \alpha_t^{-1} \to 0. \tag{7}$$

If $\{\tilde{W}^t\}$ *converges to a point* W^* *almost surely and* $\partial\psi$ *is outer semicontinuous at* W^*, *then* $W^* \in \mathcal{Z}$.

The assumption (7) is classical in analyses of dual averaging, as without such a condition, the term $\left\|W - W^0\right\|^2$ in the subproblem becomes dominant and no convergence can be expected.

Finally, with Lemmas 1 and 2 and Theorem 1, we prove the main result that the manifold is identified within finite iterations of Algorithm 1 under nondegeneracy.

Theorem 2. *Consider Algorithm* 1 *with Assumption* 1, (5)–(7) *satisfied. If* $\{\tilde{W}^t\}$ *converges to a point* W^* *almost surely;* ψ *is prox-regular and subdifferentially continuous at* W^*, *and partly smooth at* W^* *relative to the active* $\mathcal{C}^2$ *manifold* $\mathcal{M}$ *with* $\partial\psi$ *outer semicontinuous at* W^*; *and the nondegeneracy condition*

$$-\nabla f\left(W^*\right) \in \operatorname{relint} \partial\psi\left(W^*\right) \tag{8}$$

holds at W^*, *where* relint *is the relative interior, then almost surely there is a number* $T_0 \geq 0$ *such that*

$$W^t \in \mathcal{M}, \quad \forall t \geq T_0. \tag{9}$$

In other words, with probability one, the active manifold is identified by the iterates of Algorithm 1 *after a finite number of iterations.*

4 Applications in Deep Learning

In this section, we supply with details two popular schemes of structured deep learning models achieved through regularization to demonstrate the applications of our algorithm. They are structured sparsity and discrete neural networks.

4.1 Structured Sparsity

As modern deep neural network models are often tremendously large, it is sometimes desirable to trim the model to a smaller one when only limited resources are available. In this case, zeroing out redundant parameters during training at the group level is shown to be useful (Zhou et al., 2016), and one can utilize regularizers promoting structured sparsity for this purpose. The most famous regularizer for structured or group-wise sparsity is the group LASSO norm (Yuan & Lin, 2006; Friedman et al., 2010). Given $\lambda \geq 0$ and a collection $\mathcal{G}$ of index sets $\{\mathcal{I}_g\}$ of the variable W with $|\mathcal{I}_g|$ being the number of elements in $\mathcal{I}_g$, this regularizer is defined as

$$\psi(W) \coloneqq \lambda \sum_{g=1}^{|\mathcal{G}|} |\mathcal{I}_g| \|W_g\|. \tag{10}$$

The group LASSO norm is prox-regular and partly smooth around any W^* relative to the manifold $\mathcal{M} \coloneqq \{W \mid W_{\mathcal{I}_i} = 0, \forall i : W^*_{\mathcal{I}_i} = 0\}$ and is convex, so our theory applies.

In order to promote structured sparsity, we need to carefully design the grouping. Fortunately, in neural networks, the parameters can be grouped naturally (Wen et al., 2016). For any fully-connected layer, let $W \in \mathbb{R}^{\text{out}\times\text{in}}$ be the matrix representation of the associated parameters, where out is the number of output neurons and in is that of input neurons, we can consider the column-wise groups, defined as $W_{:,j}$ for all j, and the row-wise groups of the form $W_{i,:}$. For a convolutional layer with $W \in \mathbb{R}^{\text{filter}\times\text{channel}\times\text{height}\times\text{width}}$ being the tensor form of the corresponding parameters, we can consider channel-wise, filter-wise, and kernel-wise groups, defined respectively as $W_{:,j,:,:}$, $W_{i,:,:,:}$ and $W_{i,j,:,:}$.

4.2 Binary Neural Networks

Making the parameters of a neural network binary integers is another way to obtain a more compact model during training and deployment (Hubara et al., 2016), but discrete optimization is hard to scale-up. Using a vector representation $w \in \mathbb{R}^m$ of the variables, Hou et al. (2017) thus proposed to use the indicator function of the following set to induce the entries of w to be binary without resorting to discrete optimization tools.

$$\left\{ w \mid w_{\mathcal{I}_i} = \alpha_i b_{\mathcal{I}_i}, \alpha_i > 0, b_{\mathcal{I}_i} \in \{\pm 1\}^{|\mathcal{I}_i|} \right\},$$

where each $\mathcal{I}_i$ enumerates all parameters in the i-th layer. They also showed that projection to this set can be done efficiently through a closed-form solution. Yang et al. (2019) later proposed to use the following regularizer

$$\min_{w\in\mathbb{R}^m, \alpha\in[0,1]^m} \sum_{i=1}^{m} (\alpha_i (w_i + 1)^2 + (1 - \alpha_i)(w_i - 1)^2).$$

Extension to discrete neural networks beyond the binary ones is also possible, and Bai et al. (2018b) have proposed regularizers with closed-form proximal operators for such a case.

5 Experiments

In this section, we use the structured sparsity application introduced in Section 4.1 to exemplify the ability of the proposed algorithm in finding the desired structure in the trained neural network models. In particular, we use PyTorch to compare RMDA with the following algorithms proposed for identifying structured sparsity in deep learning using the same type of regularization.

- ProxSGD (Yang et al., 2019): A simple proximal SGD algorithm without variance reduction. The ProxSGD used in our experiments is slightly different from the original version by Yang et al. (2019). In their original ProxSGD, they conduct proximal operations and then update the parameters by interpolating. However, in this way the parameters would be very close to zero but will not become zero due to interpolating. Our modifications skips the interpolating part in order to obtain group sparsity.

- ProxSSI (Deleu & Bengio, 2021): This is a special case of the adaptive proximal SGD framework of Yun et al. (2020) that uses the Newton-Raphson algorithm to approximately solve the subproblem. We directly use the package released by the authors.[2]

The algorithm of Wen et al. (2016) is not included in our comparison because their method is shown to be worse than ProxSSI by Deleu & Bengio (2021). We also exclude Bungert et al. (2021) because we cannot reproduce the reported performance, and their package gives inferior results. To compare these algorithms, we examine both the validation accuracy and the group sparsity of their trained models. We compute the group sparsity as the percentage of groups whose elements are all zero. Thus, the reported group sparsity is zero when there is no group with a zero norm, while it is one when the whole model is zero. In all experiments, we use column-wise and channel-wise groupings in the regularization for model training, but use the kernel-wise grouping in our group sparsity evaluation, as a channel-wise group being zero implies that the corresponding filter-wise group in the previous layer should also be zero.

Throughout the experiments, we always use multi-step learning rate scheduling that decays the learning rate by a constant factor every time the epoch count reaches a threshold, as this setting leads to better performance for all algorithms in our trials. For all algorithms, we conduct grid search to find the best hyperparameters for the two criteria mentioned above. All results shown in tables are the mean and standard deviation of three independent runs with the same hyperparameters, while the figures use one representative run for better visualization.

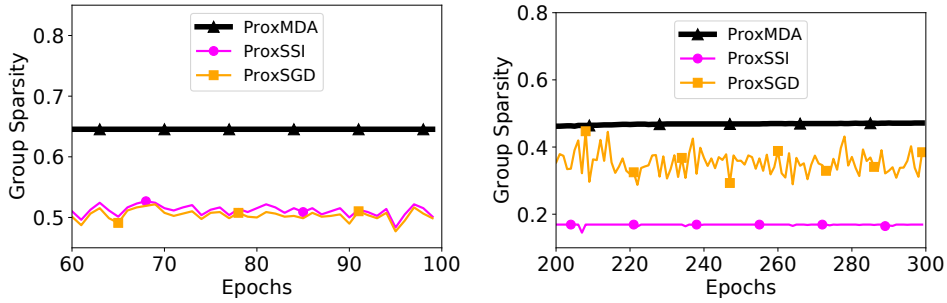
In convex optimization, for momentum-based methods like accelerated proximal gradient (Nesterov, 1983; Beck & Teboulle, 2009; Nesterov, 2013), a known way to improve the practical convergence behavior is restarting that periodically reset the momentum to zero (O'donoghue & Candes, 2015). Following this idea, we introduce a restart heuristic to RMDA as well. At each round, we use the output of Algorithm 1 from the previous round as the new input for the same algorithm, and continue using the scheduling η and c without resetting them. For the case of $\psi \equiv 0$, Jelassi & Defazio (2020) suggested to increase c_t proportional to the decrease of η_t in the multi-step scheduling until reaching $c_t = 1$. We also adopt such a setting for c_t and η_t in our experiments and restart whenever η_t changes. Moreover, we have shown Section 3 that $\tilde{W}^t$ finds the active manifold, so this increment of c_t to 1 also accords with our interest in identifying the optimal structure.

We use multiple models to thoroughly examine the ability in simultaneously identifying a stable structured sparsity pattern and achieving satisfactory validation accuracy of RMDA. From the simplest to the most complicated ones, we train a linear model, a multi-layer fully-connected network, a small (LeNet5) and a modern (VGG16) convolutional neural networks (CNN). Due to the page limit, models details and hyperparameters are put in the appendices.

5.1 Linear Model and Fully-Connected Neural Network

We start with two relatively simple problems to illustrate the difference between RMDA and other algorithms. The first one is a simple logistic regression model with the MNIST dataset (LeCun et al., 1998) and the second one is training a 7-layer fully-connected neural network on the FashionMNIST dataset (Xiao et al., 2017). The logistic regression model is linear and convex, and thus stronger convergence guarantees for all algorithms exist, so we can focus on the structured sparsity part. For the second problem, although state-of-the-art CNNs such as that of He et al. (2016); Huang et al. (2017) only use one fully-connected layer as the final layer, language transformers (Devlin et al., 2019) still include a large proportion of parameters, up to 75% of the encoder parameters, in fully-connected layers, so inducing structured sparsity in fully-connected layers can still have a significant effect in the overall model size. We therefore start from this simplified setting. Surprisingly, on such simple tasks, we already observe that existing packages cannot steadily identify the structured pattern, so their structured sparsity obtained is quite by chance and can vary a lot even if the cut-off epoch is just slightly altered.

[2] `https://github.com/tristandeleu/pytorch-structured-sparsity`.



(a) Logistic regression

(b) 7-layer fully-connected neural network

Figure 1: Group Sparsity of different algorithms on simple models of a single run.

Table 1: Group sparsity and validation accuracy of different algorithms on simple models. We report mean and standard deviation of three independent runs.

	Logistic regression		Fully-connected neural network	
Algorithm	Validation accuracy	Group sparsity	Validation accuracy	Group sparsity
ProxSGD	89.95 ± 0.00%	49.87 ± 0.00%	86.12 ± 0.36%	37.41 ± 1.47%
ProxSSI	89.80 ± 0.01%	50.13 ± 0.00%	86.37 ± 0.45%	15.74 ± 1.15%
RMDA	**89.96 ± 0.02%**	**64.58 ± 0.06%**	**88.09 ± 0.21%**	**47.53 ± 0.32%**

From Fig. 1, we can see that the group sparsity of RMDA is stable in the last epochs, while that of ProxSGD and ProxSSI is still oscillating, even though their step sizes have been decreased to close to zero. This means that, as expected, RMDA is the only algorithm that steadily identifies the correct structured sparsity, while other methods with no variance reduction fail. In addition, Table 1 shows that manifold identification of RMDA is achieved with no sacrifice in the validation accuracy.

5.2 Convolutional Neural Network

We now move to CNNs and start with training a version of LeNet5 on MNIST. We then consider VGG16 (Simonyan & Zisserman, 2015) and follow Deleu & Bengio (2021) to eliminate all fully-connected layers except the output layer. We further add one batch-norm layer (Ioffe & Szegedy, 2015) after each convolutional layer to simulate modern CNNs like those proposed in He et al. (2016); Huang et al. (2017), which also include batch-norm layers and take the output layer as the only fully-connected layer. The modified VGG16 is experimented with the CIFAR10 dataset Krizhevsky (2009). Following the standard practice in modern vision model training, we apply data augmentation through random cropping and horizontal flipping in training VGG16 so that the training problem is no longer a finite-sum one.[3] The group sparsity results are shown in Figs. 2 and 3. Notably, in the experiment of VGG16, RMDA succeeds in finding the structured sparsity pattern despite the presence of data argumentation, showing that RMDA indeed can overcome the infinite-sum setting frequently encountered in deep learning tasks, while other methods still have their group sparsity levels fluctuating way below that of RMDA, even though the step sizes are already nearly zero. For this more complicated task, we also include a baseline of momentum SGD (Bottou, 1991; LeCun et al., 1998; Bottou & Bousquet, 2008) without any sparsity-inducing regularization in our comparison in Table 2 to see how structured sparsity affects the performance. Remarkably, RMDA beats other proximal methods in not only the structured

[3]Rigorously speaking, it is still a finite-sum problem as there are only finitely many choices for cropping and flipping, but the data size becomes so large that traversing through all instances is intractable.

Table 2: Group sparsity and validation accuracy of different algorithms on CNNs. We report mean and standard deviation of three independent runs. SGD is the baseline without a group-sparsity-inducing regularizer.

	LeNet5		VGG16	
Algorithm	Validation accuracy	Group sparsity	Validation accuracy	Group sparsity
SGD	**99.28 ± 0.05%**	-	**93.05 ± 0.22%**	-
ProxSGD	99.10 ± 0.03%	33.66 ± 2.43%	90.64 ± 0.10%	73.33 ± 1.29%
ProxSSI	99.14 ± 0.04%	30.83 ± 4.59%	90.94 ± 0.05%	83.88 ± 0.10%
RMDA	99.11 ± 0.06%	**45.33 ± 0.99%**	92.52 ± 0.13%	**88.10 ± 0.09%**

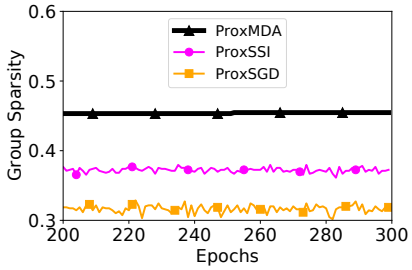


Figure 2: Group Sparsity of different algorithms on LeNet5 of a single run.

sparsity but also the validation accuracy, and the accuracy is very close to the baseline that has no sparsity. We also notice that in terms of the running time, for training VGG16 with CIFAR10 on our single NVIDIA Quadro RTX 8000 GPU, momentum SGD, ProxSGD, and RMDA all have an average per-epoch cost of around 39 seconds, while the state-of-the-art ProxSSI needs 189 seconds per epoch. Therefore, RMDA is also more suitable for large-scale deep learning problems in terms of the practical efficiency.

6 Conclusions

In this work, we proposed and analyzed a new algorithm, RMDA, for efficiently training structured neural networks with state-of-the-art performance. Even in the presence of data augmentation and dropout, our algorithm can still achieve variance reduction and provably identify the desired structure at a solution using the tools of manifold identification. Our experiments show that existing algorithms for the same purpose fail to find a stable optimal structure, while RMDA achieves so with no accuracy drop nor additional time cost.

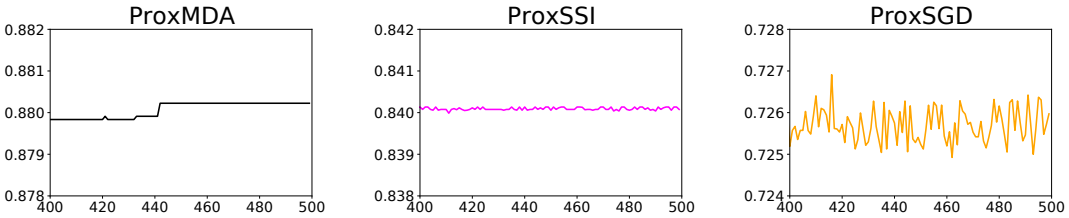


Figure 3: Group Sparsity of different algorithms on VGG16 of a single run.

Reproducibility Statement

The assumptions and the theoretical result are stated in Section 3, with the proofs in Appendix B. For the experiments, our source code will be made available in the supplementary materials and the detailed experiment settings are described in Appendix C.

References

Zeyuan Allen-Zhu. Natasha 2: Faster non-convex optimization than SGD. In *Advances in Neural Information Processing Systems*, volume 31, pp. 2675–2686, 2018.

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

Yu Bai, Yu-Xiang Wang, and Edo Liberty. Proxquant: Quantized neural networks via proximal operators. Technical report, 2018b.

Gilles Bareilles, Franck Iutzeler, and Jérôme Malick. Newton acceleration on manifolds identified by proximal-gradient methods. Technical report, 2020. arXiv:2012.12936.

Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

Léon Bottou. Stochastic gradient learning in neural networks. *Proceedings of Neuro-Nımes*, 91(8):12, 1991.

Léon Bottou and Olivier Bousquet. The tradeoffs of large scale learning. In J. Platt, D. Koller, Y. Singer, and S. Roweis (eds.), *Advances in Neural Information Processing Systems*, volume 20. Curran Associates, Inc., 2008.

Patrick Breheny and Jian Huang. Penalized methods for bi-level variable selection. *Statistics and its interface*, 2(3):369, 2009.

Leon Bungert, Tim Roith, Daniel Tenbrinck, and Martin Burger. A Bregman learning framework for sparse neural networks. Technical report, 2021. arXiv:2105.04319.

Ashok Cutkosky and Francesco Orabona. Momentum-based variance reduction in non-convex SGD. In *Advances in Neural Information Processing Systems*, volume 32, pp. 15236–15245, 2019.

Aaron Defazio and Leon Bottou. On the ineffectiveness of variance reduced optimization for deep learning. In *Advances in Neural Information Processing Systems*, volume 32, pp. 1755–1765, 2019.

Aaron Defazio and Samy Jelassi. Adaptivity without compromise: a momentumized, adaptive, dual averaged gradient method for stochastic optimization. Technical report, 2021. arXiv:2101.11075.

Aaron Defazio, Francis Bach, and Simon Lacoste-Julien. SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives. In *Advances in neural information processing systems*, pp. 1646–1654, 2014.

Tristan Deleu and Yoshua Bengio. Structured sparsity inducing adaptive optimizers for deep learning. Technical report, 2021. arXiv:2102.03869.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2019.

John C. Duchi and Feng Ruan. Asymptotic optimality in stochastic optimization. *The Annals of Statistics*, 49(1):21–48, 2021.

Cong Fang, Chris Junchi Li, Zhouchen Lin, and Tong Zhang. SPIDER: near-optimal non-convex optimization via stochastic path integrated differential estimator. In *Advances in Neural Information Processing Systems*, pp. 687–697, 2018.

Jerome Friedman, Trevor Hastie, and Robert Tibshirani. A note on the group lasso and a sparse group lasso. Technical report, 2010.

A.M. Gupal. *Stochastic methods for solving nonsmooth extremal problems*. Naukova Dumka, 1979.

Warren L. Hare. Identifying active manifolds in regularization problems. In *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, pp. 261–271. Springer, 2011.

Warren L. Hare and Adrian S. Lewis. Identifying active constraints via partial smoothness and prox-regularity. *Journal of Convex Analysis*, 11(2):251–266, 2004.

Warren L. Hare and Adrian S. Lewis. Identifying active manifolds. *Algorithmic Operations Research*, 2(2):75–75, 2007.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Lu Hou, Quanming Yao, and James T Kwok. Loss-aware binarization of deep networks. In *International Conference on Learning Representations*, 2017.

Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. In *IEEE conference on computer vision and pattern recognition*, pp. 4700–4708, 2017.

Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized neural networks. *Advances in neural information processing systems*, 29, 2016.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pp. 448–456, 2015.

Franck Iutzeler and Jérôme Malick. Nonsmoothness in machine learning: specific structure, proximal identification, and applications. *Set-Valued and Variational Analysis*, 28(4): 661–678, 2020.

Samy Jelassi and Aaron Defazio. Dual averaging is surprisingly effective for deep learning optimization. Technical report, 2020. arXiv:2010.10502.

Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. *Advances in neural information processing systems*, 26:315–323, 2013.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Ching-pei Lee. Accelerating inexact successive quadratic approximation for regularized optimization through manifold identification. Technical report, 2020. arXiv:2012.02522.

Sangkyun Lee and Stephen J. Wright. Manifold identification in dual averaging for regularized stochastic online learning. *Journal of Machine Learning Research*, 13:1705–1744, 2012.

Adrian S. Lewis. Active sets, nonsmoothness, and sensitivity. *SIAM Journal on Optimization*, 13(3):702–725, 2002.

Adrian S. Lewis and Shanshan Zhang. Partial smoothness, tilt stability, and generalized hessians. 23(1):74–94, 2013.

Yu-Sheng Li, Wei-Lin Chiang, and Ching-pei Lee. Manifold identification for ultimately communication-efficient distributed optimization. In *International Conference on Machine Learning*, 2020.

Jingwei Liang, Jalal Fadili, and Gabriel Peyré. Activity identification and local linear convergence of forward–backward-type methods. 27(1):408–437, 2017a.

Jingwei Liang, Jalal Fadili, and Gabriel Peyré. Local convergence properties of douglas–rachford and alternating direction method of multipliers. *Journal of Optimization Theory and Applications*, 172(3):874–913, 2017b.

Jingwei Liang, Jalal Fadili, and Gabriel Peyré. Local linear convergence analysis of primal–dual splitting methods. *Optimization*, 67(6):821–853, 2018.

Yurii Nesterov. Gradient methods for minimizing composite functions. *Mathematical Programming*, 140(1):125–161, 2013.

Yurii E. Nesterov. A method of solving a convex programming problem with convergence rate $O(1/k^2)$. *Soviet Mathematics Doklady*, 27:372–376, 1983.

Lam M. Nguyen, Katya Scheinberg, and Martin Takáč. Inexact SARAH algorithm for stochastic optimization. *Optimization Methods and Software*, 36(1):237–258, 2021.

Evgeni Alekseevich Nurminskii. The quasigradient method for the solving of the nonlinear programming problems. *Cybernetics*, 9(1):145–150, 1973.

Brendan O'donoghue and Emmanuel Candes. Adaptive restart for accelerated gradient schemes. *Foundations of computational mathematics*, 15(3):715–732, 2015.

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*, 21:1–48, 2020.

René Poliquin and R Rockafellar. Prox-regular functions in variational analysis. *Transactions of the American Mathematical Society*, 348(5):1805–1838, 1996.

Clarice Poon, Jingwei Liang, and Carola-Bibiane Schönlieb. Local convergence properties of SAGA/prox-SVRG and acceleration. In *International Conference on Machine Learning*, 2018.

Othmane Sebbouh, Robert M Gower, and Aaron Defazio. On the convergence of the stochastic heavy ball method. Technical report, 2020.

Connor Shorten and Taghi M. Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1):1–48, 2019.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. 2015.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

Yifan Sun, Halyun Jeong, Julie Nutini, and Mark Schmidt. Are we there yet? manifold identification of gradient-related proximal methods. In *International Conference on Artificial Intelligence and Statistics*, pp. 1110–1119, 2019.

Wei Tao, Zhisong Pan, Gaowei Wu, and Qing Tao. Primal averaging: A new gradient evaluation step to attain the optimal individual convergence. *IEEE transactions on cybernetics*, 50(2):835–845, 2018.

Quoc Tran-Dinh, Nhan H. Pham, Dzung T. Phan, and Lam M. Nguyen. Hybrid stochastic gradient descent algorithms for stochastic nonconvex optimization. Technical report, 2019.

Zhe Wang, Kaiyi Ji, Yi Zhou, Yingbin Liang, and Vahid Tarokh. Spiderboost and momentum: Faster variance reduction algorithms. In *Advances in Neural Information Processing Systems*, volume 32, pp. 2406–2416, 2019.

Wei Wen, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. Learning structured sparsity in deep neural networks. *Advances in neural information processing systems*, 29: 2074–2082, 2016.

Wei Wen, Yuxiong He, Samyam Rajbhandari, Minjia Zhang, Wenhan Wang, Fang Liu, Bin Hu, Yiran Chen, and Hai Li. Learning intrinsic sparse structures within long short-term memory. In *International Conference on Learning Representations*, 2018.

Stephen J. Wright. Accelerated block-coordinate relaxation for regularized optimization. *SIAM Journal on Optimization*, 22(1):159–186, 2012.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. Technical report, 2017. arXiv:1708.07747.

Lin Xiao. Dual averaging methods for regularized stochastic learning and online optimization. *Journal of Machine Learning Research*, 11(88):2543–2596, 2010.

Lin Xiao and Tong Zhang. A proximal stochastic gradient method with progressive variance reduction. *SIAM Journal on Optimization*, 24(4):2057–2075, 2014.

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*, 2019.

Ming Yuan and Yi Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(1):49–67, 2006.

Jihun Yun, Aurelie C. Lozano, and Eunho Yang. A general family of stochastic proximal gradient methods for deep learning. Technical report, 2020. arXiv:2007.07484.

Cun-Hui Zhang. Nearly unbiased variable selection under minimax concave penalty. *The Annals of statistics*, 38(2):894–942, 2010.

Hao Zhou, Jose M. Alvarez, and Fatih Porikli. Less is more: Towards compact cnns. In *European Conference on Computer Vision*, pp. 662–677. Springer, 2016.

A Organization of the Appendices

The appendices are organized as follows. The detailed proofs are given in Appendix B. Appendix C provides experiment details for Section 5. We then discuss the possibility of different regularizers for structured sparsity and conduct additional experiments in Appendix D.

B Proofs

B.1 Proof of Lemma 1

Proof. We observe that

$$\begin{aligned}\alpha_t^{-1} V^t &= \sum_{k=1}^{t} \frac{\eta_k \beta_k}{\alpha_t} \nabla f_{\xi_k}\left(W^{k-1}\right) \\ &= \frac{\alpha_{t-1}}{\alpha_t} \alpha_{t-1}^{-1} V^{t-1} + \frac{\alpha_t - \alpha_{t-1}}{\alpha_t} \nabla f_{\xi_t}\left(W^{t-1}\right) \\ &= \left(1 - \frac{\beta_t \eta_t}{\alpha_t}\right) \alpha_{t-1}^{-1} V^{t-1} + \frac{\beta_t \eta_t}{\alpha_t} \nabla f_{\xi_t}\left(W^{t-1}\right).\end{aligned}$$

From that f is L-Lipschitz-continuously differentiable, we have that

$$\begin{aligned}\left\|\mathbb{E}_{\xi_{t+1}\sim\mathcal{D}}\left[\nabla f_{\xi_{t+1}}\left(W^t\right)\right] - \mathbb{E}_{\xi_t\sim\mathcal{D}}\left[\nabla f_{\xi_t}\left(W^{t-1}\right)\right]\right\| &= \left\|\nabla f\left(W^t\right) - f\left(W^{t-1}\right)\right\| \\ &\leq L\left\|W^t - W^{t-1}\right\|. \end{aligned} \tag{11}$$

Therefore, (5) and (11) imply that

$$0 \leq \frac{\left\|\mathbb{E}_{\xi_{t+1}\sim\mathcal{D}}\left[\nabla f_{\xi_{t+1}}\left(W^t\right)\right] - \mathbb{E}_{\xi_t\sim\mathcal{D}}\left[\nabla f_{\xi_t}\left(W^{t-1}\right)\right]\right\|}{\beta_t \eta_t \alpha_t^{-1}} \leq L \frac{\left\|W^t - W^{t-1}\right\|}{\beta_t \eta_t \alpha_t^{-1}} \xrightarrow{\text{a.s.}} 0,$$

which together with the sandwich lemma shows that

$$\frac{\left\|\mathbb{E}_{\xi_{t+1}\sim\mathcal{D}}\left[\nabla f_{\xi_{t+1}}\left(W^t\right)\right] - \mathbb{E}_{\xi_t\sim\mathcal{D}}\left[\nabla f_{\xi_t}\left(W^{t-1}\right)\right]\right\|}{\beta_t \eta_t \alpha_t^{-1}} \xrightarrow{\text{a.s.}} 0. \tag{12}$$

Therefore, the first two conditions of (5) together with (12) and (4) satisfy the requirements of (Gupal, 1979, Chapter 2.4, Theorem 4.1), so the conclusion follows. □

B.2 Proof of Lemma 2

Proof. Using (3), the distance between W^t and W^* can be upper bounded through the triangle inequality:

$$\begin{aligned}\left\|W^t - W^*\right\| &= \left\|(1 - c_t)\left(W^{t-1} - W^*\right) + c_t\left(\tilde{W}^t - W^*\right)\right\| \\ &\leq c_t\left\|\tilde{W}^t - W^*\right\| + (1 - c_t)\left\|W^{t-1} - W^*\right\|. \end{aligned} \tag{13}$$

For any event such that $\tilde{W}^t \to W^*$, for any $\epsilon > 0$, we can find $T_\epsilon \geq 0$ such that

$$\left\|\tilde{W}^t - W^*\right\| \leq \epsilon, \quad \forall t \geq T_\epsilon.$$

Let $\delta_t \coloneqq \|W^t - W^*\|$, we see from the above inequality and (13) that

$$\delta_t \leq (1 - c_t)\,\delta_{t-1} + c_t \epsilon, \quad \forall t \geq T_\epsilon.$$

By deducting ϵ from both sides, we get that

$$(\delta_t - \epsilon) \leq (1 - c_t)\,(\delta_{t-1} - \epsilon), \quad \forall t \geq T_\epsilon.$$

Since $\sum c_t = \infty$, we further deduce that

$$\begin{aligned}\lim_{t\to\infty}(\delta_t - \epsilon) &\leq \prod_{t=T_\epsilon}^{\infty}(1-c_t)(\delta_{T_\epsilon-1}-\epsilon)\\ &\leq \prod_{t=T_\epsilon}^{\infty}\exp(-c_t)(\delta_{T_\epsilon-1}-\epsilon)\\ &= \exp\left(-\sum_{t=T_\epsilon}^{\infty} c_t\right)(\delta_{T_\epsilon-1}-\epsilon) = 0,\end{aligned}$$

where in the first inequality we used the fact that $\exp(x) \geq 1 + x$ for all real number x. Thie result above then implies that

$$\lim_{t\to\infty}\delta_t \leq \epsilon.$$

As ϵ is arbitrary and $\delta_t \geq 0$ from the definition, we conclude that $\lim_{t\to\infty}\delta_t = 0$, which is equivalent to that $W^t \to W^*$. As the event that $\{\tilde{W}^t\}$ converges to W^* is with probability one and $\{W^t\}$ converges to W^* whenever $\{\tilde{W}^t\}$ does, we conclude that $W^t \xrightarrow{\text{a.s.}} W^*$. □

B.3 Proof of Theorem 1

Proof. Using Lemma 1, we can we can view $\alpha_t^{-1}V^t$ as $\nabla f(W^t)$ plus some noise that asymptotically decreases to zero with probability one:

$$\alpha_t^{-1}V_t = \nabla f(W^t) + \epsilon_t, \quad \|\epsilon_t\| \xrightarrow{\text{a.s.}} 0. \tag{14}$$

We use (14) to rewrite the optimality condition of (2) as (also see Line 6 of Algorithm 1)

$$-\left(\nabla f\left(W^t\right) + \epsilon_t + \beta_t\alpha_t^{-1}\left(\tilde{W}^t - W^0\right)\right) \in \partial\psi\left(\tilde{W}^t\right). \tag{15}$$

Now we consider $\partial F(\tilde{W}^t)$. Clearly from (15), we have that

$$\nabla f\left(\tilde{W}^t\right) - \nabla f\left(W^t\right) - \epsilon_t - \beta_t\alpha_t^{-1}\left(\tilde{W}^t - W^0\right) \in \partial\nabla f\left(\tilde{W}^t\right) + \psi\left(\tilde{W}^t\right) = \partial F\left(\tilde{W}^t\right). \tag{16}$$

From Lemma 2, we know that $W^t \xrightarrow{\text{a.s.}} 0$ as well. For the three events that $\epsilon_t \to 0$, $W^t \to W^*$, and $\tilde{W}^t \to W^*$, since each individual one happens with probability one, the union of the three events also happens with probability one. We thus consider any such realization. For the right-hand side of (16), as $\tilde{W}^t$ is convergent and $\beta_t\alpha_t^{-1}$ decreases to zero, by letting t approaches infinity, we have that

$$\lim_{t\to\infty}\epsilon_t + \beta_t\alpha_t^{-1}\left(\tilde{W}^t - W^0\right) = 0 + 0\left(W^* - W^0\right) = 0.$$

By the continuity of ∇f from Assumption 1, we have from (3) and (5) that

$$0 \leq \left\|\nabla f\left(\tilde{W}^t\right) - \nabla f\left(W^t\right)\right\| \leq L\left\|W^t - \tilde{W}^t\right\|.$$

As $\{W^t\}$ and $\{\tilde{W}^t\}$ converge to the same point, we also see that $\left\|W^t - \tilde{W}^t\right\| \to 0$, so $\nabla f\left(\tilde{W}^t\right) - \nabla f(W^t)$ also approaches zero. Hence, the the limit of the right-hand side of (16) is

$$\lim_{t\to\infty}\nabla f\left(\tilde{W}^t\right) - \left(\nabla f\left(W^t\right) + \epsilon_t + \beta_t\alpha_t^{-1}\left(\tilde{W}^t - W^0\right)\right) = 0. \tag{17}$$

On the other hand, for the left-hand side of (16), the outer semicontinuity of $\partial\psi$ at W^* and the continuity of ∇f show that

$$\lim_{t\to\infty}\nabla f(\tilde{W}^t) + \partial\psi(\tilde{W}^t) \subseteq \partial\nabla f(W^*) + \psi\left(W^*\right) = \partial F(W^*). \tag{18}$$

Substituting (17) and (18) back into (16) then proves that $0 \in \partial F(W^*)$ and thus $W^* \in \mathcal{Z}$. □

B.4 Proof of Theorem 2

Proof. From Lemma 2, we know that $W^t \xrightarrow{\text{a.s.}} W^*$. From (15), there is $U^t \in \partial\psi\left(\tilde{W}^t\right)$ such that

$$U^t = -\alpha_t^{-1} V^t + \alpha_t^{-1}\beta_t\left(\tilde{W}^t - W^0\right). \tag{19}$$

From Lemma 1, we can view $\alpha_t^{-1}V^t$ as $\nabla f(W^t)$ plus some noise that asymptotically decreases to zero with probability one as shown in (14). Moreover, we define

$$\gamma_t \coloneqq W^t - \tilde{W}^t. \tag{20}$$

By combining (19)–(20), we obtain

$$\begin{aligned}
&\min_{Y\in\partial F(\tilde{W}^t)} \|Y\| \\
\leq& \left\|\nabla f\left(\tilde{W}^t\right) + U^t\right\| \\
=& \left\|\nabla f\left(\tilde{W}^t\right) - \nabla f\left(W^t\right) - \epsilon_t - \alpha_t^{-1}\beta_t\left(\tilde{W}^t - W^0\right)\right\| \\
\leq& \left\|\nabla f\left(\tilde{W}^t\right) - \nabla f\left(W^t\right)\right\| + \|\epsilon_t\| + \alpha_t^{-1}\beta_t\left\|\tilde{W}^t - W^0\right\| \\
\leq& L\|\gamma_t\| + \|\epsilon_t\| + \alpha_t^{-1}\beta_t\left(\left\|W^* - \tilde{W}^t\right\| + \left\|W^0 - W^*\right\|\right),
\end{aligned} \tag{21}$$

where we used the Lipschitz continuity of ∇f and the triangle inequality in the last inequality.

We now separately bound the terms in (21). From that $W^t \xrightarrow{\text{a.s.}} W^*$ and $\tilde{W}^t \xrightarrow{\text{a.s.}} W^*$, it is straightforward that $\|\gamma_t\| \xrightarrow{\text{a.s.}} 0$. The second term also decreases to zero almost surely according to (14). For the last term, since $\alpha_t^{-1}\beta_t \to 0$, and $\left\|\tilde{W}^t - W^*\right\| \xrightarrow{\text{a.s.}} 0$, we know that

$$\alpha_t^{-1}\beta_t\left\|W^0 - W^*\right\| \to 0, \quad \alpha_t^{-1}\beta_t\left\|\tilde{W}^t - W^*\right\| \xrightarrow{\text{a.s.}} 0.$$

Therefore, we conclude from the above argument and (21) that

$$\min_{Y\in\partial F(\tilde{W}^t)} \|Y\| \xrightarrow{\text{a.s.}} 0.$$

As f is smooth with probability one, we know that if ψ is partly smooth at W^* relative to $\mathcal{M}$, then so is $F = f + \psi$ with probability one. Moreover, Lipschitz-continuously differentiable functions are always prox-regular, and the sum of two prox-regular functions is still prox-regular, so F is also prox-regular at W^* with probability one. As $\tilde{W}^t \xrightarrow{\text{a.s.}} W^*$ and $\nabla f(\tilde{W}^t) + U^t \xrightarrow{\text{a.s.}} 0 \in \partial F(W^*)$ (the inclusion is from (8)), we have from the subdifferrential continuity of ψ and the smoothness of f that $F(\tilde{W}^t) \xrightarrow{\text{a.s.}} F(W^*)$. Since we also have $\tilde{W}^t \xrightarrow{\text{a.s.}} W^*$ and $\min_{Y\in\partial F(\tilde{W}^t)} \|Y\| \xrightarrow{\text{a.s.}} 0$, clearly

$$\left(\tilde{W}^t, F\left(W^t\right), \min_{Y\in\partial F\left(\tilde{W}^t\right)} \|Y\|\right) \xrightarrow{\text{a.s.}} \left(W^*, F(W^*), 0\right). \tag{22}$$

Therefore, (22) and (8) together with the assumptions on ψ at W^* imply that with probability one, all conditions of Theorem 4.10 of Lewis & Zhang (2013) are satisfied, so from it, (9) holds almost surely. □

C Experiment Setting Details

All ProxSSI parameter settings expect the regularization weight and the learning rate schedule follow the default values in their package. Tables 3 and 4 provide the detailed setting of Section 5.1 while Tables 5 and 6 are for Section 5.2.

D Regularizers for Better Group Sparsity and Generalization

A downside of (10) is that it pushes all groups toward zeros and thus introduces bias in the final model. For its remedy, minimax concave penalty (MCP, Zhang, 2010) is then proposed

Table 3: Details of the experimental settings of logistic regression on MNIST in Section 5.1.

Parameter	Value
Data set	MNIST
Model	Logistic regression
Loss function	Cross entropy
Regularization function	Group LASSO
Total epochs	100
ProxSGD	
Regularization weight	10^{-3}
Learning rate schedule	$\eta(\text{epoch}) = 10^{-1-\lfloor \text{epoch}/10 \rfloor}$
Momentum schedule	0.9
ProxSSI	
Regularization weight	3×10^{-3}
Learning rate schedule	$\eta(\text{epoch}) = 10^{-3-\lfloor \text{epoch}/10 \rfloor}$
RMDA	
Regularization weight	10^{-4}
Learning rate schedule	$\eta(\text{epoch}) = \max(10^{-3}, 10^{-1-\lfloor \text{epoch}/10 \rfloor})$
Momentum schedule	$c(\text{epoch}) = \min(10^{-2+\lfloor \text{epoch}/10 \rfloor}, 1)$
Restart epochs	$10, 20$

Table 4: Details of the experimental settings of the multi-layer fully-connected neural network on FashionMNIST in Section 5.1

Parameter	Value
Data set	FashionMNIST
Model	Fully-connected neural network (see Table 7)
Loss function	Cross entropy
Regularization function	Group LASSO
Total epochs	300
ProxSGD	
Regularization weight	10^{-4}
Learning rate schedule	$\eta(\text{epoch}) = 10^{-2-\lfloor \text{epoch}/30 \rfloor}$
Momentum schedule	0.9
ProxSSI	
Regularization weight	10^{-5}
Learning rate schedule	$\eta(\text{epoch}) = 10^{-4-\lfloor \text{epoch}/30 \rfloor}$
RMDA	
Regularization weight	7×10^{-6}
Learning rate schedule	$\eta(\text{epoch}) = \max(10^{-6}, 10^{-1-\lfloor \text{epoch}/30 \rfloor})$
Momentum schedule	$c(\text{epoch}) = \min(1, 10^{-2+\lfloor \text{epoch}/30 \rfloor})$
Restart epochs	$30, 60, 90, 120, 150$

to penalize only the groups whose norm is smaller than a user-specified threshold. More precisely, given hyperparameters $\lambda \geq 0, c \geq 1$, the one-dimensional MCP is defined by

$$\text{MCP}(w) := \begin{cases} \lambda|w| - \frac{w^2}{2c} & \text{if} |w| < c\lambda, \\ \frac{c\lambda^2}{2} & \text{if} |w| \geq c\lambda. \end{cases}$$

Given the grouping $\mathcal{G}$, MCP is then defined as (Breheny & Huang, 2009)

$$\text{MCP}\left(W_{\mathcal{I}_g}\right) := \begin{cases} \lambda|\mathcal{I}_g|\left\|W_{\mathcal{I}_g}\right\|^2 - \frac{\left\|W_{\mathcal{I}_g}\right\|^2}{2c} & \text{if } \left\|W_{\mathcal{I}_g}\right\| < c\lambda, \\ \frac{c\lambda^2|\mathcal{I}_g|^2}{2} & \text{if} \left\|W_{\mathcal{I}_g}\right\| \geq c\lambda, \end{cases} \quad \psi(W) := \sum_{g=1}^{|\mathcal{G}|} MCP\left(W_{\mathcal{I}_g}\right).$$

It is shown in Deleu & Bengio (2021) that group MCP regularization can simultaneously provide higher group sparsity and better validation accuracy than the group LASSO norm

Table 5: Details of the experimental settings of LeNet5 on MNIST in Section 5.2

Parameter	Value
Data set	MNIST
Model	LeNet5 Table 8
Loss function	Cross entropy
Regularization function	Group LASSO
Total epochs	300
ProxSGD	
Regularization weight	1×10^{-4}
Learning rate schedule	$\eta(\text{epoch}) = 10^{-2} \times 0.1^{\text{epoch}//30}$
Momentum schedule	$c(\text{epoch}) = 0.9$
ProxSSI	
Regularization weight	5×10^{-5}
Learning rate schedule	$\eta(\text{epoch}) = 10^{-3} \times 0.1^{\text{epoch}//30}$
RMDA	
Regularization weight	1×10^{-5}
Learning rate schedule	$\eta(\text{epoch}) = 10^{-1} \times 0.1^{\text{epoch}//30}$, if epoch ≤ 120; 10^{-5}, otherwise
Momentum schedule	$c(\text{epoch}) = 10^{-2} \times 10^{\text{epoch}//30}$, if epoch ≤ 60; 1, otherwise
Restart epochs	$30, 60, 90, 120$

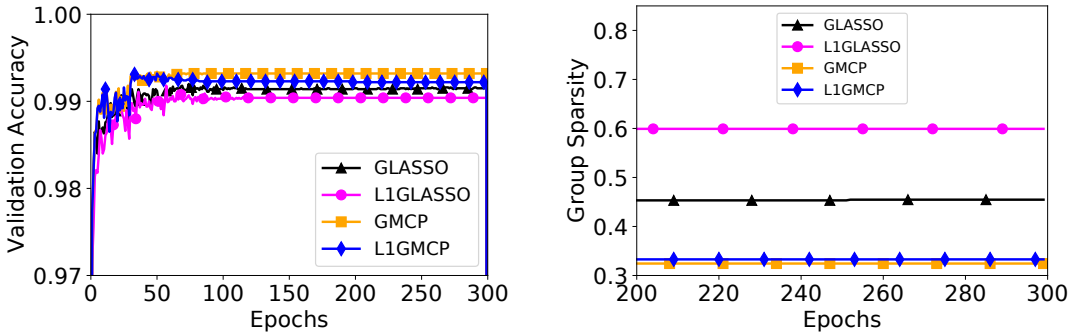


Figure 4: Comparisons between Gruop LASSO, L1+Gruop LASSO, Group MCP, L1+Group MCP

in vision and language tasks. Another possibility to enhance sparsity is to add another ℓ_1-norm or entry-wise MCP regularization to the group-level regularizer. The major drawback of these approaches is the requirement of additional hyperparameters, and we prefer simpler approaches over those with more hyperparameters as their tuning can be troublesome for users with limited computational resources. The experiment in this subsection is therefore only for illustrating that these more complicated regularizers can be combined with RMDA if the user wishes and might lead to better results. Therefore, we train a simple CNN (LeNet5) on the MNIST dataset with such regularizers using RMDA and display the respective performance of various regularization schemes in Fig. 4. In this figure, Gruop LASSO is abbreviated as GLASSO, ℓ_1-norm plus a Gruop LASSO norm is abbreviated as L1GLASSO, Group MCP is abbreviated as GMCP, and element-wise MCP plus group MCP is abbreviated as L1+GMCP. Our results exemplify that different regularization schemes might have different benefits on one of the criterion with proper hyperparameters tuning. The detailed numbers are reported in Table 10 and the experiment settings can be found in Table 11.

Table 6: Details of the experimental settings of modified VGG16 on CIFAR10 in Section 5.2

Parameter	Value
Data set	CIFAR10
Model	VGG16 Table 9
Loss function	Cross entropy
Regularization function	Group LASSO
Total epochs	500
ProxSGD	
Regularization weight	1×10^{-4}
Learning rate schedule	$\eta(\text{epoch}) = 10^{-2} \times 0.1^{\text{epoch}//50}$
Momentum schedule	$c(\text{epoch}) = 0.9$
ProxSSI	
Regularization weight	1×10^{-4}
Learning rate schedule	$\eta(\text{epoch}) = 10^{-4} \times 0.1^{\text{epoch}//50}$
RMDA	
Regularization weight	2×10^{-5}
Learning rate schedule	$\eta(\text{epoch}) = 10^{-1} \times 0.1^{\text{epoch}//50}, \text{if epoch} \leq 250; 10^{-6}, \text{otherwise}$
Momentum schedule	$c(\text{epoch}) = 10^{-2} \times 10^{\text{epoch}//50}, \text{if epoch} \leq 100; 1, \text{otherwise}$
Restart epochs	$50, 100, 150, 200, 250$

Table 7: Details of the multi-layer fully-connected neural network.

Parameter	Value
Type of layers	fully-connected layer
Number of layers	7
Number of output neurons each layer: 1,2,3,4,5,6,7	512,256,128,64,32,16,10
Activation function for convolution/output layer	relu/softmax

Table 8: Details of the modified LeNet5.

Parameter	Value
Number of layers	5
Number of convolutional layers	13
Number of fully-connected layers	1
Size of convolutional kernels	3×3
Number of output filters 1,2	6,16
Number of output neurons 3,4,5	120,84,10
Kernel size, stride, padding of maxing pooling	2×2, none, invalid
Operations after convolutional layers	max pooling
Activation function for convolution/output layer	relu/softmax

Table 9: Details of the modified VGG16.

Parameter	Value
Number of layers	14
Number of convolutional layers	13
Number of fully-connected layers	1
Size of convolutional kernels	3×3
Number of output filters 1-2, 3-4, 5-7, 8-13	64, 128, 256, 512
Kernel size, stride, padding of maxing pooling	2×2, 2, invalid
Operations after convolutional layers	max pooling, batchnorm
Activation function for convolution/output layer	relu/softmax

Table 10: Results of training LeNet5 on MNIST with different regularizers. We report mean and standard deviation of three independent runs.

Regularizers	Validation accuracy	Group sparsity
Group LASSO	99.11 ± 0.06%	45.33 ± 0.99%
Group LASSO+L1	99.02 ± 0.01%	58.92 ± 1.30%
Gruop MCP	99.25 ± 0.08%	32.81 ± 0.96%
Gruop MCP + L1	99.21 ± 0.03%	32.91 ± 0.35%

Table 11: Details of the experimental settings of comparing different regularizers in Appendix D

Parameter	Value
Data set	MNIST
Model	LeNet5 Table 8
Loss function	Cross entropy
Algorithms	RMDA
Total epochs	300
Group LASSO only	
Group LASSO weight	1×10^{-5}
Learning rate schedule	$\eta(\text{epoch}) = 10^{-1} \times 0.1^{\text{epoch}//30}, \text{if epoch} \leq 120; 10^{-5}, \text{otherwise}$
Momentum schedule	$c(\text{epoch}) = 10^{-2} \times 10^{\text{epoch}//30}, \text{if epoch} \leq 60; 1, \text{otherwise}$
Restart epochs	$30, 60, 90, 120$
L1 + Group LASSO	
L1 weight	1×10^{-4}
Group LASSO weight	1×10^{-5}
Learning rate schedule	$\eta(\text{epoch}) = 10^{-1} \times 0.1^{\text{epoch}//30}, \text{if epoch} \leq 120; 10^{-5}, \text{otherwise}$
Momentum schedule	$c(\text{epoch}) = 10^{-2} \times 10^{\text{epoch}//30}, \text{if epoch} \leq 60; 1, \text{otherwise}$
Restart epochs	$30, 60, 90, 120$
Group MCP only	
Group MCP weight	1×10^{-5}
γ	64
Learning rate schedule	$\eta(\text{epoch}) = 10^{-1} \times 0.1^{\text{epoch}//30}, \text{if epoch} \leq 120; 10^{-5}, \text{otherwise}$
Momentum schedule	$c(\text{epoch}) = 10^{-2} \times 10^{\text{epoch}//30}, \text{if epoch} \leq 60; 1, \text{otherwise}$
Restart epochs	$30, 60, 90, 120$
L1 + Group MCP	
L1 weight	1×10^{-4}
Group MCP weight	1×10^{-5}
γ	64
Learning rate schedule	$\eta(\text{epoch}) = 10^{-1} \times 0.1^{\text{epoch}//30}, \text{if epoch} \leq 120; 10^{-5}, \text{otherwise}$
Momentum schedule	$c(\text{epoch}) = 10^{-2} \times 10^{\text{epoch}//30}, \text{if epoch} \leq 60; 1, \text{otherwise}$
Restart epochs	$30, 60, 90, 120$