Training Structured Neural Networks Through Manifold Identification and Variance Reduction

LEE Ching-pei

Joint work with Zih-Syuan Huang (AS)

Outline

Structured Models

- In many scenarios, it is desirable to train a machine learning model with certain structures
- Usually achieved by adding a regularizer Ψ to the training/optimization objective function
- Examples (regularizer in the bracket):
  - Prevent overfitting/Achieve low model complexity (ℓ_2-norm regularization)
  - Satisfy certain constraints (indicator function of the feasible set)
  - Achieve structured or unstructured sparsity (ℓ_1-norm or group-LASSO norm)

Motivating Example: Structured Sparsity I

- In many cases, many model parameters can be trimmed out without affecting its generalization ability
- Trimming out such parameters achieves sparsity and can reduce computational burden for prediction
- For neural networks, we want to trim out neurons (in fully-connected layers) or a whole convolutional kernel, but not just individual weights, to really reduce model size and accelerate prediction
- But surely we still want to start from an overparameterized one before starting training
- GPUs are unable to do sparse matrix operations efficiently, mainly because of the memory access pattern
  - Dense computation: sequential access, cache miss minimized

Motivating Example: Structured Sparsity II

- Sparse computation: non-continuous access, high miss rate

- Group-LASSO norm: parameters that should either exist together or be removed together are bound together
- Given $\lambda > 0$ and a collection $\mathcal{G}$ of index sets $\{\mathcal{I}_g\}$ of the model variable W, together with weights $\{w_g\}$, this regularizer is defined as

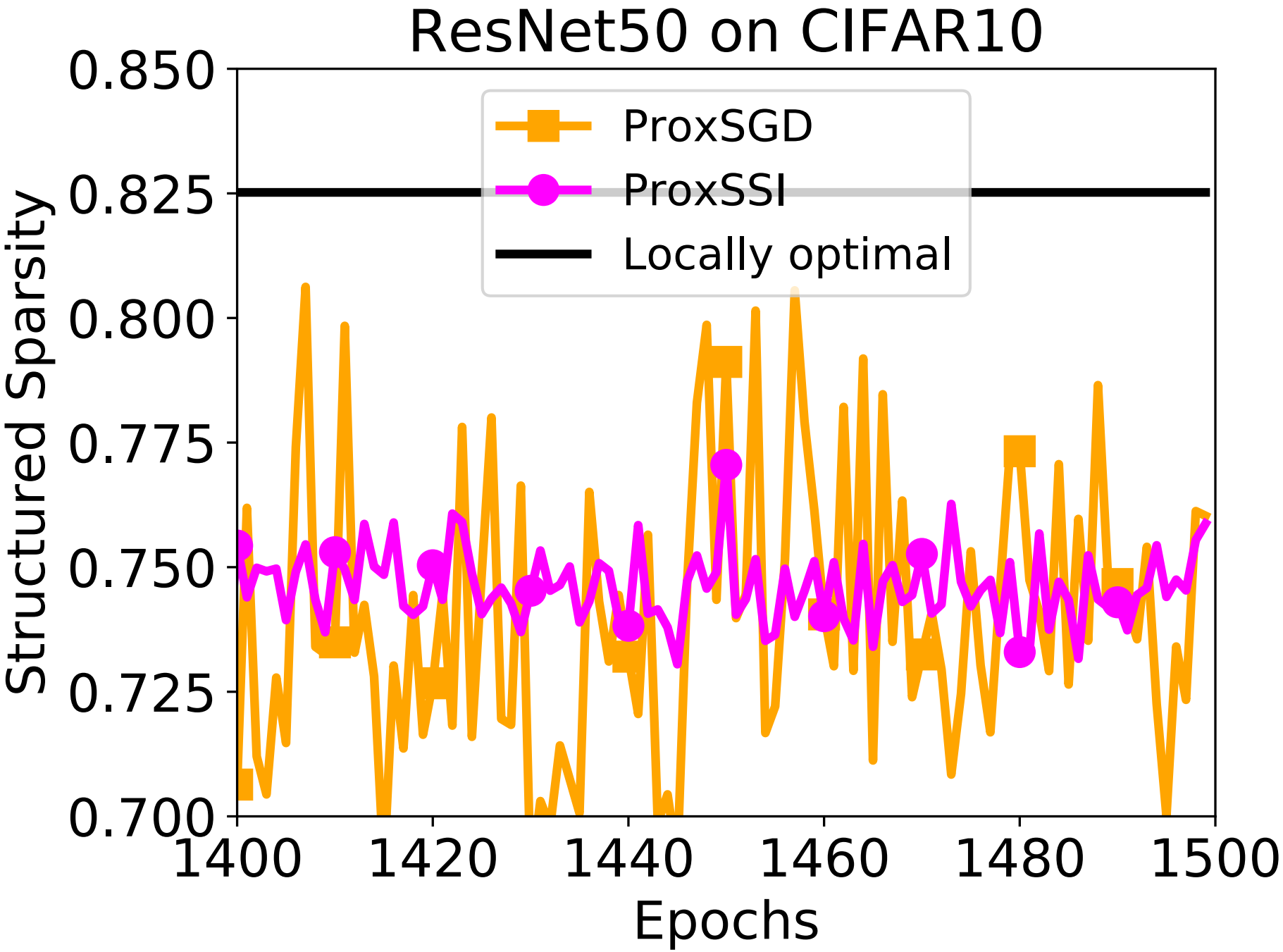
$$\psi(W) := \lambda \sum_{g=1}^{|\mathcal{G}|} w_g \left\| W_{\mathcal{I}_g} \right\|$$

Regularizers and Structures

- Model-side analyses use (first-order necessary) optimality conditions to show that at stationary points, certain structures will occur
- But we don't have such guarantees for approximate solutions
- Unfortunately, training/optimization algorithms can only provide approximate solutions: they generate a sequence $\{W^t\}$ of models such that $W^t \to W^*$ for some W^* that is a stationary point/solution, and output W^T for some T when the algorithm terminates
- What do we know about structures at W^T?

State of the Art in Deep Learning

- State of the art for training structured neural networks: only have convergence guarantees to stationary points, but no guarantee for the structure of their output model even if a regularizer is incorporated
  - Stochastic subgradient methods (Wen et al., 2016, 2018): no structure at all at the output; reported result requires a post-processing step and another round of training
  - Proximal stochastic gradient methods: (Yang et al., 2019; Bai et al., 2019; Deleu and Bengio, 2021; Yun et al., 2021): identify artificial structure from the proximal operator from the regularizer
    - The output structure can be very far away from that at W^* due to the variance of the stochastic gradients
    - Known to output unstable and highly suboptimal structure even in the convex setting (Sun et al., 2019; Poon et al., 2018)


ResNet50 on CIFAR10
ProxSGD
ProxSSI
Locally optimal
Structured Sparsity
0.850
0.825
0.800
0.775
0.750
0.725
0.700
1400
1420
1440
1460
1480
1500
Epochs

Manifold Identification I

- For regularizer that are partly smooth at a point W^*, the structure at W^* can be represented as a low-dimensional manifold
- A function is partly smooth at W^* if it is smooth around W^* when restricted to a certain smooth manifold $\mathcal{M}$, and its value changes drastically along directions leaving $\mathcal{M}$

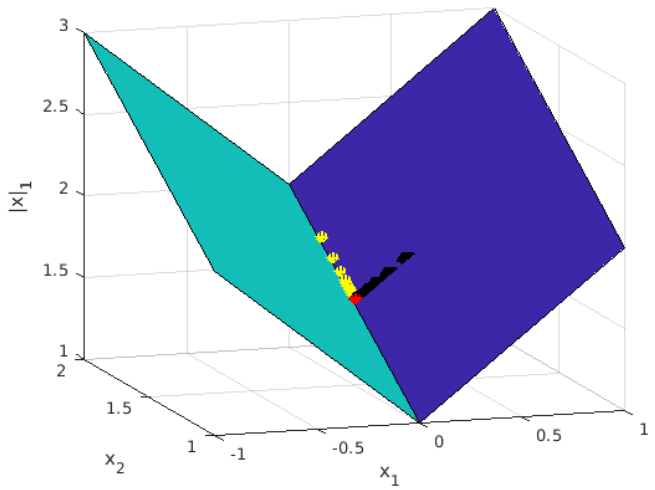


Figure: Example:ℓ_1-norm with the associated manifold (the valley) for the red dot W^*

Manifold Identification II

- Most regularizers popular in machine learning are partly smooth: ℓ_1 norm, group-LASSO norm, nuclear norm, etc
- The structure at the solution W^* can be identified if for $\{W^t\}$ converging to W^*, W^t is in $\mathcal{M}$ for all t large enough
- Called manifold identification in nonlinear optimization: our goal! (locally optimal structure for sequences converging to W^*)
- Deterministic first-order methods like proximal-gradient-type methods are known to achieve so

Variance Reduction

- For stochastic methods to achieve manifold idenification, variance reduction is needed (Poon et al., 2018)
  - The missing element in existing methods for training structured NNs
- But variance reduction that utilizes the finite-sum structure of ERM does not work for deep learning (Defazio and Bottou, 2019) because of data augmentation
- Need an algorithm achieving variance reduction in the infinite-sum setting, while
  - being practically feasible: not more expensive than SGD + momentum
  - Incorporate momentum for good prediction performance
- Our proposal: regularized modernized dual averaging (RMDA), inspired by RDA (Xiao, 2010) and MDA (Jelassi and Defazio, 2020)

Our Contributions

- Variance reduction beyond finite-sum with low cost
- Guaranteed optimal structure identification in finite steps
- Superior empirical performance over state of the art for both structured sparsity and pruning

Outline

Problem Formulation

Consider the following regularized optimization problem:

$$\min_{W \in \mathcal{E}} \quad F(W) := \mathbb{E}_{\xi \sim \mathcal{D}} \left[f_\xi (W) \right] + \psi (W) \qquad \text{(Regularized Loss)}$$

- $\mathcal{E}$ 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 $\xi \in \Omega$
- $\psi(W)$ is a regularizer that might be nondifferentiable

Algorithmic Framework

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 $G^t \leftarrow \nabla f_{\xi_t}(W^{t-1})$

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

$\tilde{W}^t \leftarrow \operatorname{argmin}_W \langle V^t, W \rangle + \frac{\beta_t}{2}\|W - W^0\|^2 + \alpha_t \psi(W)$

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

output: The final model W^T

Algorithm Details

Starting with an initial point W^0, at the $t > 0$ iteration,

- Draw an independent sample $\xi_t \sim \mathcal{D}$ to compute the stochastic gradient $\nabla f_{\xi_t}(W^{t-1})$
- Deciding a learning rate η_t and the scaling factor $\beta_t := \sqrt{t}$
- $V_t := \sum_{k=1}^{t} \eta_k \beta_k \nabla f_{\xi_k}(W^{k-1}) = V_{t-1} + \eta_t \beta_t \nabla f_{\xi_t}(W^{t-1})$
- $\tilde{W}^t := \mathrm{prox}_{\frac{\alpha_t}{\beta_t}\psi}\left(W^0 - \frac{V^t}{\beta_t}\right), \alpha_t := \sum_{k=1}^{t} \beta_k \eta_k$: dual weighted averaging
- $\mathrm{prox}_g(x) := \mathrm{argmin}_y \quad \frac{1}{2}\|x - y\|^2 + g(y)$: proximal operator
- $W^t = (1 - c_t)\, W^{t-1} + c_t \tilde{W}^t = W^{t-1} + c_t \left(\tilde{W}^t - W^{t-1}\right)$: momentum
- Multi-stage learning rates; restart V_t and α_t (set to 0) whenever the learning rate changes

Outline

Lemma 1

Consider Algorithm 1. Assume for any $\xi \sim \mathcal{D}$, f_ξ is L-Lipschitz-continuously-differentiable almost surely for some L, and there is $C \geq 0$ such that $\mathbb{E}_{\xi_t \sim \mathcal{D}} \left\| \nabla f_{\xi_t}\left(W^{t-1}\right) \right\|^2 \leq C$ for all t. If $\{\eta_t\}$ satisfies

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

then $\alpha_t^{-1} V^t \longrightarrow \nabla f(W^{t-1})$ with probability one. Moreover, if $\{W^t\}$ lies in a bounded set, we get $\mathbb{E}\left\| \alpha_t^{-1} V^t - \nabla f\left(W^{t-1}\right) \right\|^2 \to 0$ even if the second condition in (1) is replaced by a weaker condition of $\beta_t \eta_t \alpha_t^{-1} \to 0$.

Convergence to Stationary Points

Theorem 2

Consider Algorithm 1 with the conditions in Lemma 1 hold, and $\{c_t\}$ *satisfying* $\sum c_t = \infty$. *Assume the set of stationary points* $\mathcal{Z} := \{W \mid 0 \in \partial F(W)\}$ *is nonempty and* $\beta_t \alpha_t^{-1} \to 0$. *For any given* W^0, *consider the event that* $\{\tilde{W}^t\}$ *converges to a point* W^* *(each event corresponds to a different* W^**), then if* $\partial\psi$ *is outer semicontinuous at* W^*, *and this event has a nonzero probability,* $W^* \in \mathcal{Z}$, *or equivalently,* W^* *is a stationary point, with probability one conditional on this event.*

Manifold Identification of RMDA

Theorem 3

Consider Algorithm 1 with the conditions in Theorem 2 satisfied. Consider the event of $\{\tilde{W}^t\}$ *converging to a certain point* W^* *as in Theorem 2, if the probability of this event is nonzero;* ψ *is prox-regular and subdifferentially continuous at* W^* *and partly smooth at* W^* *relative to the active* $\mathcal{C}^2$ *manifold* $\mathcal{M}$*;* $\partial\psi$ *is outer semicontinuous at* W^**; and the nondegeneracy condition*

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

holds at W^**, then conditional on this event, almost surely there is* $T_0 \geq 0$ *such that*

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

In other words, the active manifold at W^* *is identified by the iterates of Algorithm 1 after a finite number of iterations almost surely.*

Outline

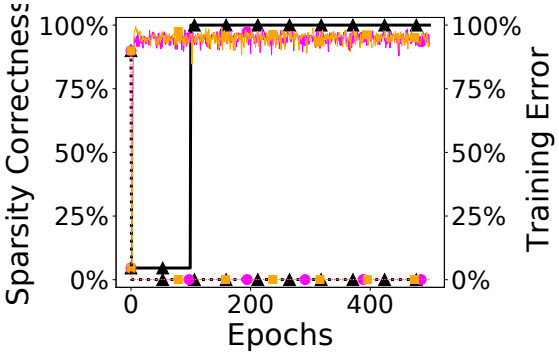
Setting

- Task: structured sparsity by using the group-LASSO norm
  - Each channel in convolutional layers as one group
  - All outputs from one neuron as one group

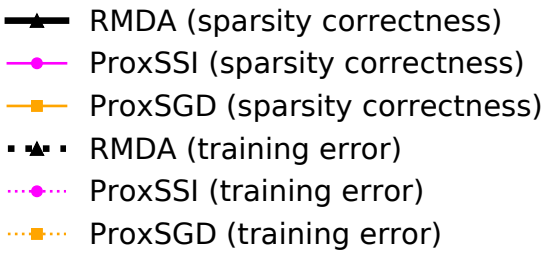
  ¡++¿
- We compare with the following state of the art methods for this task
  - RMDA: Our method
  - ProxSGD (Yang et al., 2019): A simple proxMSGD algorithm.
  - ProxSSI (Deleu and Bengio, 2021): This is a special case of the adaptive proximal SGD framework of Yun et al. (2021)
  - MSGD: SGD with momentum, this is a dense baseline

Stability and Correctness

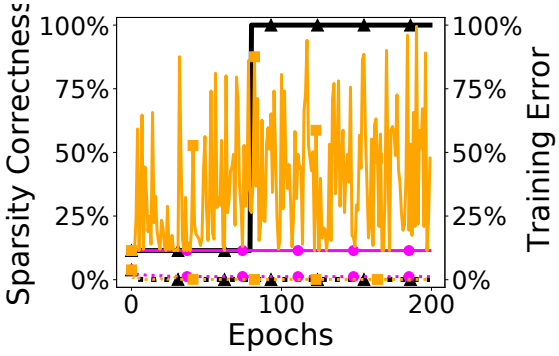
- Group sparsity pattern correctness and training error rates on synthetic data
- Generate a sparse model first and decide data labels using it
- Solid lines: sparsity pattern correctness
- dotted lines: prediction accuracy on *training* data



Logistic regression

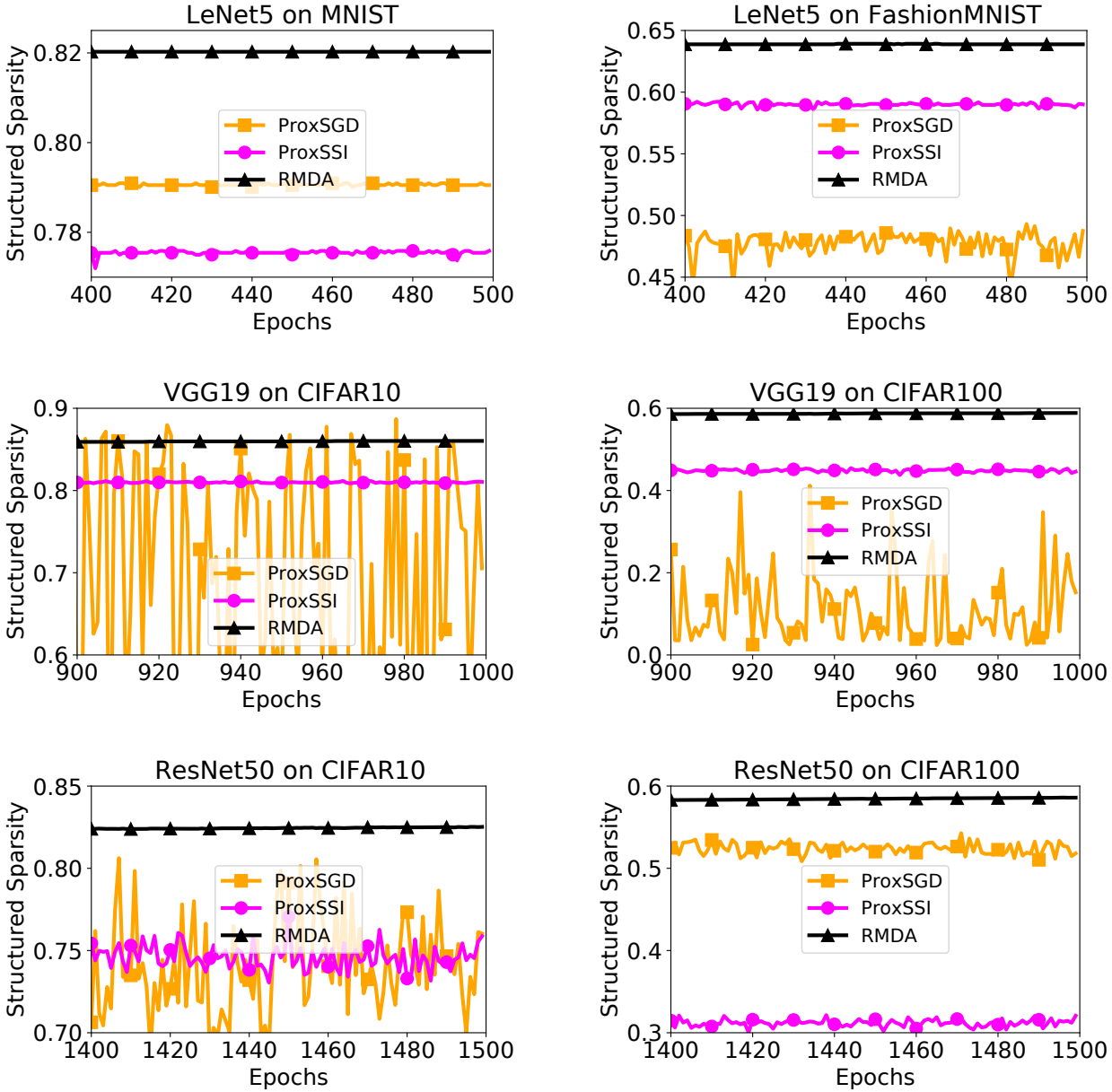


Legend



Convolutional network

Structured Sparsity v.s. Epochs



Final Structured Sparsity and Validation Accuracy

Algorithm	Validation acc.	Group sparsity	Validation acc.	Group sparsity
	LeNet5/MNIST		LeNet5/FashionMNIST	
Dense	99.4 ± 0.1%	-	92.0 ± 0.0%	-
ProxSGD	99.1 ± 0.0%	76.6 ± 2.3%	91.0 ± 0.2%	50.5 ± 2.7%
ProxSSI	99.1 ± 0.0%	77.8 ± 1.6%	90.9 ± 0.0%	60.5 ± 1.1%
RMDA	99.1 ± 0.1%	79.8 ± 1.6%	91.4 ± 0.1%	66.2 ± 1.7%
	VGG19/CIFAR10		VGG19/CIFAR100	
Dense	94.0 ± 0.1%	-	74.6 ± 0.2%	-
ProxSGD	92.4 ± 0.3%	72.6 ± 6.0%	71.9 ± 0.1%	08.6 ± 4.9%
ProxSSI	92.5 ± 0.0%	81.1 ± 0.2%	66.2 ± 0.4%	46.4 ± 1.4%
RMDA	93.6 ± 0.2%	86.4 ± 0.3%	72.2 ± 0.2%	58.9 ± 0.4%
	ResNet50/CIFAR10		ResNet50/CIFAR100	
Dense	95.7 ± 0.0%	-	79.1 ± 0.2%	-
ProxSGD	92.4 ± 0.1%	76.8 ± 4.1%	75.5 ± 0.5%	51.8 ± 0.3%
ProxSSI	94.1 ± 0.1%	74.8 ± 1.3%	74.5 ± 0.3%	32.8 ± 2.5%
RMDA	94.3 ± 0.0%	83.0 ± 0.5%	76.1 ± 0.5%	57.7 ± 3.8%

Comparison with Pruning

- ℓ_1 norm for pruning/unstructured sparsity
- Compare RMDA with a state-of-the-art pruning method: RigL (Evci et al., 2020, ICML'20) by Google Brain/DeepMind
- $1,000$ epochs for both RMDA and RigL

	ResNet50 with CIFAR10		ResNet50 with CIFAR100	
Algorithm	Sparsity	Accuracy	Sparsity	Accuracy
Dense baseline		94.81%		74.61%
RMDA	98.36%	93.78%	98.32%	74.32%
RigL	98.00%	93.41%	98.00%	70.88%

Conclusions

- Proposed an algorithm RMDA for training structured neural networks, utilizing variance reduction and manifold identification
- Experiments on structured and unstructured sparsity outperformed state of the art
- Code available at https://www.github.com/zihsyuan1214/rmda

  Full paper at https://openreview.net/pdf?id=mdUYT5QV0O
- Future work: an adaptive version, and extend to other tasks