Coordinating Filters for Faster Deep Neural Networks

About Me   Code

Wei Wen[1], Cong Xu[2], Chunpeng Wu[1], Yandan Wang[3], Yiran Chen[1], Hai Li[1]

Duke University[1], HP Labs[2], University of Pittsburgh[3]

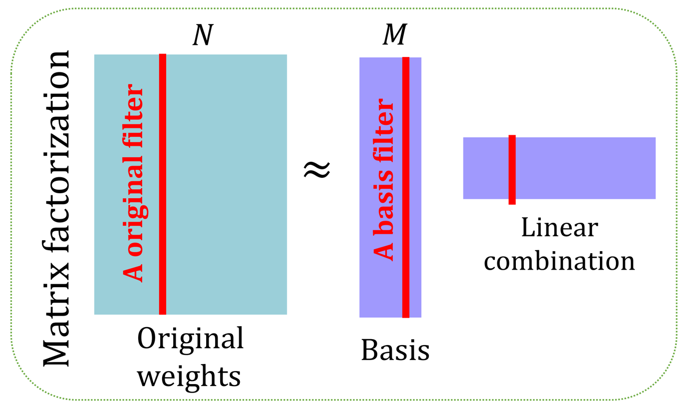
Background

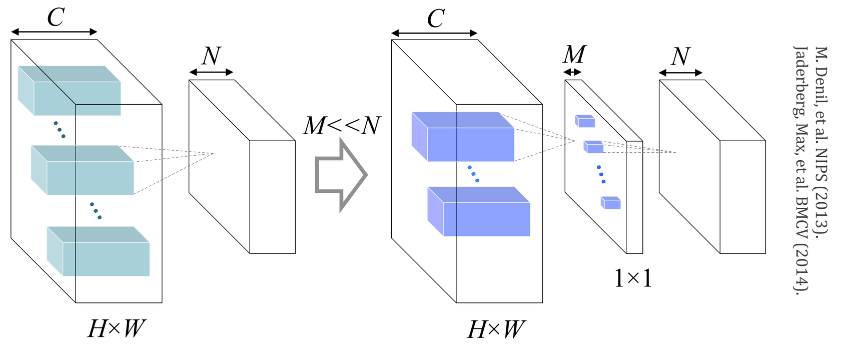
Goal

- ✓ Speedup the inference of Deep Neural Networks (DNNs)
- ✓ Focus on convolutional layers in deep neural networks

Low Rank Approximation (LRA) of Deep Neural Networks

- ✓ Filters are redundant and highly correlated with each other
- ✓ Decompose filters (weight matrices) to low-rank space

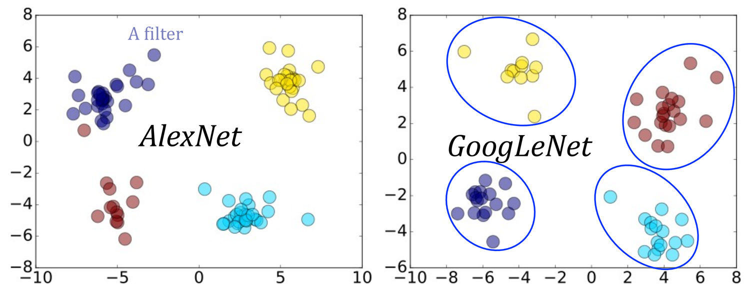




A big conv layer ≈ two small conv layers

This Work

- ✓ Coordinate filters to low**er**-rank space such that LRA has **more** compact DNNs



- ✓ Figure: Projected conv1 filters to 2D space by Linear Discriminant Analysis for visualization
- ✓ Goal: Coordinate a cluster of filters closer to each other (or even merge multiple clusters to one)
- ✓ An example: use each mean filter to approximate a cluster
  - Closer filters in a cluster -> more accurate LRA
  - Fewer clusters -> fewer mean filters (lower rank)

Method (*Force Regularization*)

Motivation

- ✓ Suppose the vector of a filter ($\mathbf{W}_i$) is a star in the universe
- ✓ There is pairwise gravity ($\mathbf{f}_{ji}$) between stars
- ✓ Gravity forces tend to pull stars closer
- ✓ Inertia resists stars to completely collapse

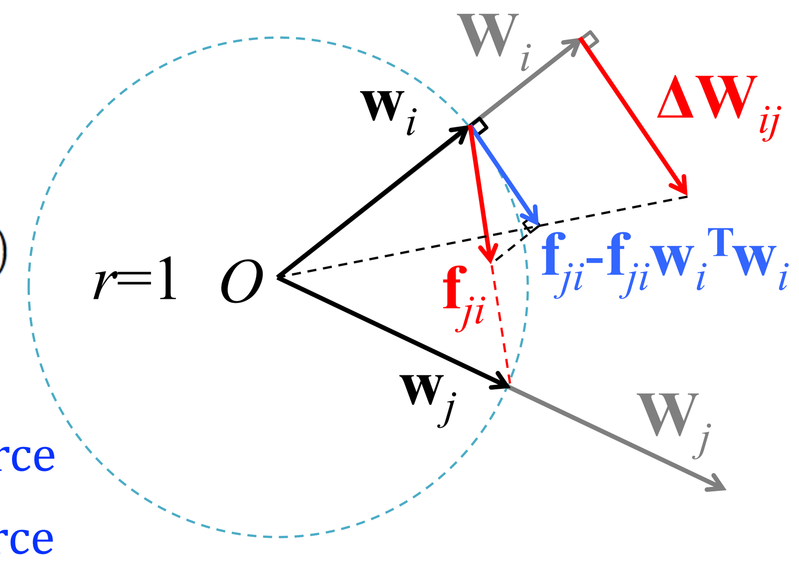
Force Regularization

- ✓ Introduce additional gradients in Stochastic Gradient Descent (SGD):

$$\Delta\mathbf{W}_i = \underbrace{\sum_{j=1}^{N}\Delta\mathbf{W}_{ij}}_{\text{Forces from all other stars/filters}} = \|\mathbf{W}_i\|\sum_{j=1}^{N}\left(\mathbf{f}_{ji} - \mathbf{f}_{ji}\mathbf{w}_i^T\mathbf{w}_i\right)$$

$$\mathbf{f}_{ji} = f(\mathbf{w}_j - \mathbf{w}_i) = \mathbf{w}_j - \mathbf{w}_i \quad L_2\text{-norm force}$$

$$\text{or} \quad = \frac{\mathbf{w}_j - \mathbf{w}_i}{\|\mathbf{w}_j - \mathbf{w}_i\|} \quad L_1\text{-norm force}$$



SGD Training with *Force Regularization*

- ✓ Filters are updated by both loss function gradients and force gradients:

$$\mathbf{W}_i \leftarrow \mathbf{W}_i - \eta\cdot\left(\boxed{\frac{\partial E(\mathcal{W})}{\partial\mathbf{W}_i}} - \lambda_s\cdot\boxed{\Delta\mathbf{W}_i}\right) \qquad \Delta\mathbf{W}_i = \sum_{j=1}^{N}\Delta\mathbf{W}_{ij} = \|\mathbf{W}_i\|\sum_{j=1}^{N}\left(\mathbf{f}_{ji} - \mathbf{f}_{ji}\mathbf{w}_i^T\mathbf{w}_i\right)$$

Minimize error (Inertia)   Reduce ranks (Gravity)

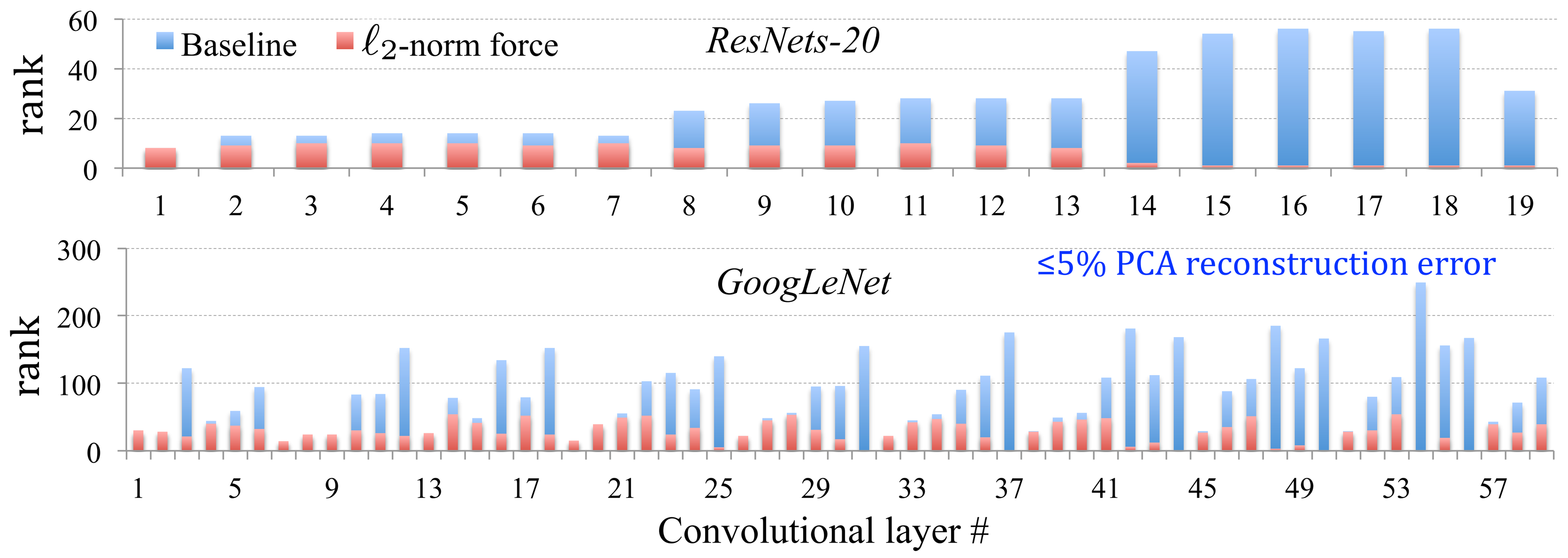
Intuitive *Force Regularization* has strong mathematical implications

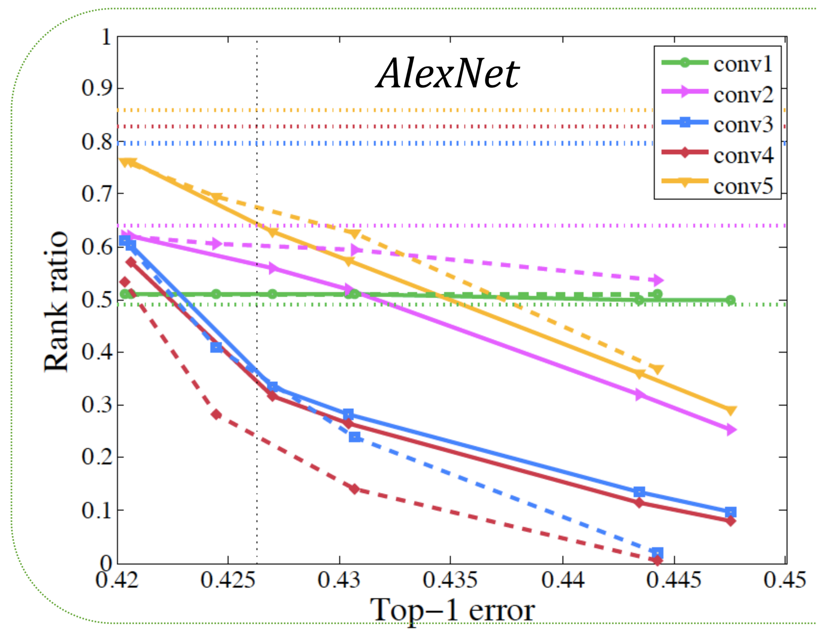
- ✓ Two types of regularization have
  - the same gradient direction, but
  - different step sizes

Force Regularization	Regularization by sum of pairwise distances
L_2-norm force ⟷	$R(\mathcal{W}) = \frac{1}{2}\sum_{j=1}^{N}\sum_{i=1}^{N}\left\|\frac{\mathbf{W}_j}{\|\mathbf{W}_j\|} - \frac{\mathbf{W}_i}{\|\mathbf{W}_i\|}\right\|^2.$
L_1-norm force ⟷	$R(\mathcal{W}) = \sum_{j=1}^{N}\sum_{i=1}^{N}\left\|\frac{\mathbf{W}_j}{\|\mathbf{W}_j\|} - \frac{\mathbf{W}_i}{\|\mathbf{W}_i\|}\right\|.$

Experiments

Coordinating DNNs to lower-rank space by *Force Regularization*





- ✓ ≤ 5% PCA reconstruction error
- ✓ Horizontal dotted lines: baseline ranks
- ✓ Vertical dotted line: baseline error
- ✓ Solid curves: L_2-norm force
- ✓ Dashed curves: L_1-norm force
- ✓ Control λ_s to make trade-off
- ✓ Reduce ranks without accuracy loss

Speedup by *Force Regularization*

Table 4. The higher speedups of *AlexNet* by *Force Regularization*.

Force	Top-1 error		conv3	conv4	conv5
None	43.21%	rank	184	201	146
ℓ_2-norm	43.25%	rank	124	106	129
None	43.21%	GPU	1.58×	1.21×	1.15×
ℓ_2-norm	43.25%	GPU	2.16×	2.03×	1.33×
None	43.21%	CPU	1.78×	1.60×	1.47×
ℓ_2-norm	43.25%	CPU	2.45×	2.76×	1.64×
None	43.21%	theoretical	1.79×	1.72×	1.63×
ℓ_2-norm	43.25%	theoretical	2.65×	3.26×	1.85×

Originally very low ranks in conv1&2 are maintained

Step 1: LRA to the same ranks

Step 2: Fine-tuning DNNs after LRA

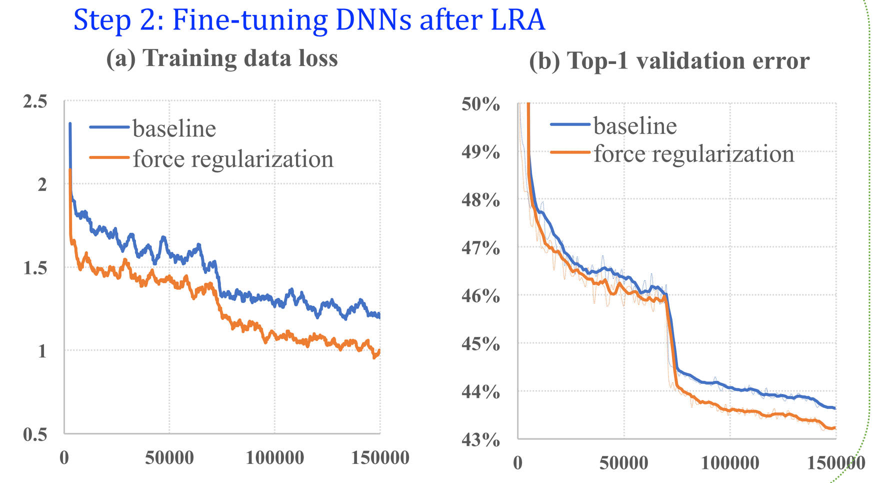
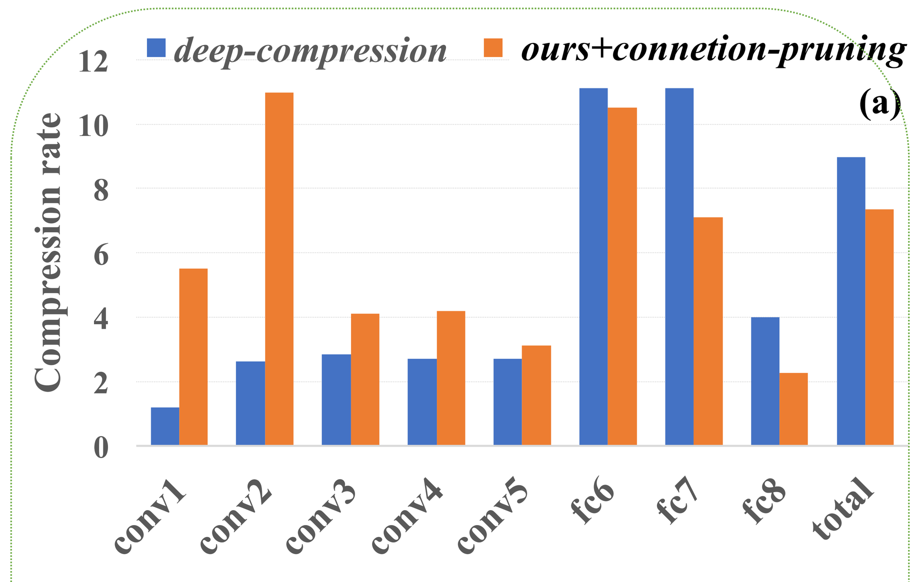


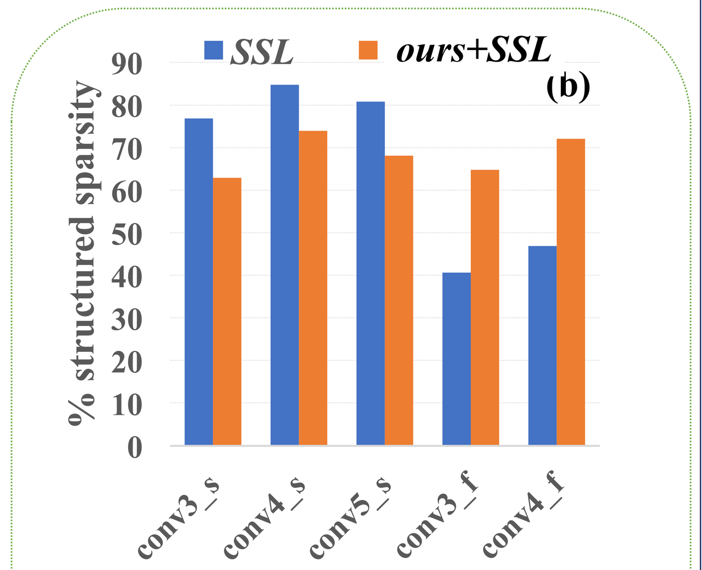
Table 6. Comparison of speedup factor on *AlexNet* by state-of-the-art DNN acceleration methods.

Method	Top-5 err.	conv1	conv2	conv3	conv4	conv5	total
AlexNet in Caffe	19.97%	1.00×	1.00×	1.00×	1.00×	1.00×	**1.00**×
cp-decomposition [1]	20.97% (+1.00%)	–	4.00×	–	–	–	**1.27**×
one-shot [2]	21.67% (+1.70%)	1.48×	2.30×	3.84×	3.53×	3.13×	**2.52**×
SSL [3]	19.58% (-0.39%)	1.00×	1.27×	1.64×	1.68×	1.32×	**1.35**×
	21.63% (+1.66%)	1.05×	3.37×	6.27×	9.73×	4.93×	**3.13**×
ours	20.14% (+0.17%)	2.61×	6.06×	2.48×	2.20×	1.58×	**2.69**×
	21.68% (+1.71%)	2.65×	6.22×	4.81×	4.00×	2.92×	**4.05**×

[1] V. Lebedev, et al. ICLR 2015; [2] Y.-D. Kim, et al. ICLR 2016; [3] W. Wen, et al. NIPS 2016

Lower rank + sparse DNNs





- ✓ *deep-compression*: S. Han, et al., NIPS 2015 (only counting compression from connection pruning)
- ✓ Non-structurally sparse DNNs
- ✓ Higher compression in conv layers for computation saving
- ✓ Comparable total compression rate
- ✓ Higher speedup (~2.7x)

- ✓ *SSL*: W. Wen, et al., NIPS 2016
- ✓ Structurally sparse DNNs
- ✓ conv3_s: 1st small conv3 after LRA
- ✓ conv3_f: 2nd small conv3 after LRA
- ✓ Ours can work with SSL for potentially higher speedup

Acknowledgments

This work was supported in part by NSF CCF-1744082. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF or their contractors.