Learning to share: Simultaneous parameter tying and sparsification in deep learning

Dejiao Zhang†, Haozhu Wang†, Mário A.T. Figueiredo*, Laura Balzano†

†Department of Electrical Engineering and Computer Science, University of Michigan, USA
*Instituto de Telecomunicacõe and Instituto Superior Técnico, University of Lisbon, Portugal

Motivation and Objective

- **Motivation:** DNNs usually require expensive storage and computation.
- **Goal:** compress DNNs by **(i)** simultaneously eliminating unimportant neurons; **(ii)** tying together weights that correspond to strongly correlated neurons.
- **Outcome:** by automatically tying together weights corresponding to highly correlated features, we alleviate the negative effect of strong correlations that may be induced by noisy inputs or co-adaption.

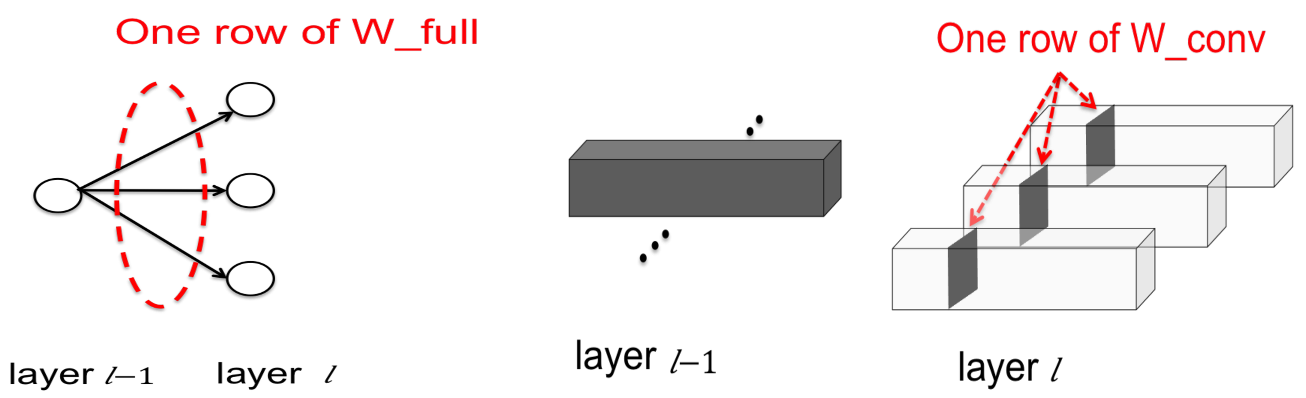
GrOWL Regularization for Deep Learning

GrOWL (group ordered weighted ℓ_1). Given a matrix $W \in \mathbb{R}^{n \times m}$, let $w_{[i]\cdot}$ denote the row of W with the i-th largest ℓ_2 norm. Let $\lambda \in \mathbb{R}^n_+$, with $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n \geq 0$, with $\lambda_1 > 0$ The GrOWL regularizer is defined as

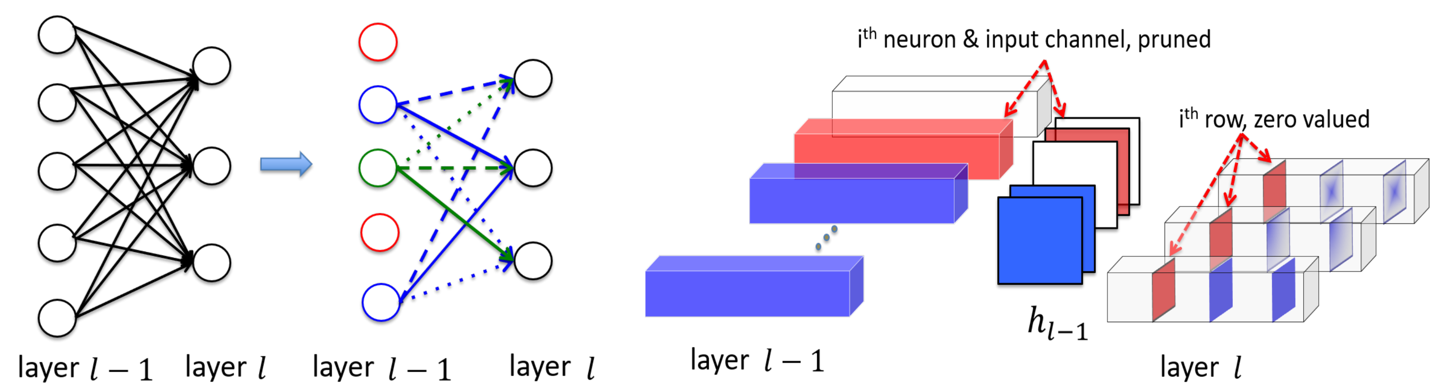
$$\Omega_\lambda(W) = \sum_{i=1}^{n} \lambda_i \left\| w_{[i]\cdot} \right\|$$

Layerwise GrOWL Regularization: Let $\mathcal{L}$ denote the loss incurred by a DNN, N_l the # neurons in the l-th layer, and W_l be the weight matrix.
DNN learning can be formalized as an optimization problem:

$$\min_\theta \mathcal{L}(\theta) + \mathcal{R}(\theta), \quad \text{where } \mathcal{R}(\theta) = \sum_{l=1}^{L} \Omega_{\lambda^{(l)}}(W_l), \quad \lambda^{(l)} \in \mathbb{R}_+^{N_{l-1}}. \qquad (1)$$



Predefined row of fully connected layer and convolutional layer



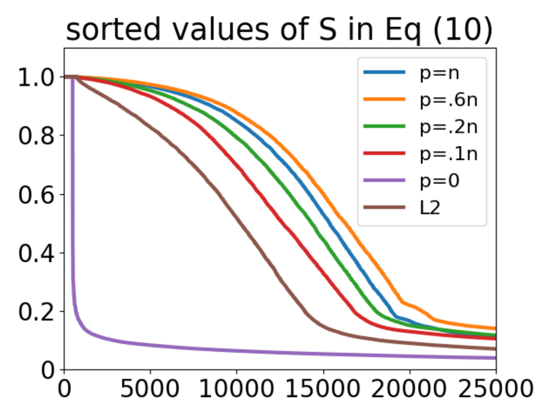
GrOWL regularization effect: simultaneously eliminating unimportant (red) neurons by setting the associated weights to zero, and explicitly identifying strongly correlated (blue) neurons by tying the corresponding weights to a common/close value(s).

Stairwise Regularization

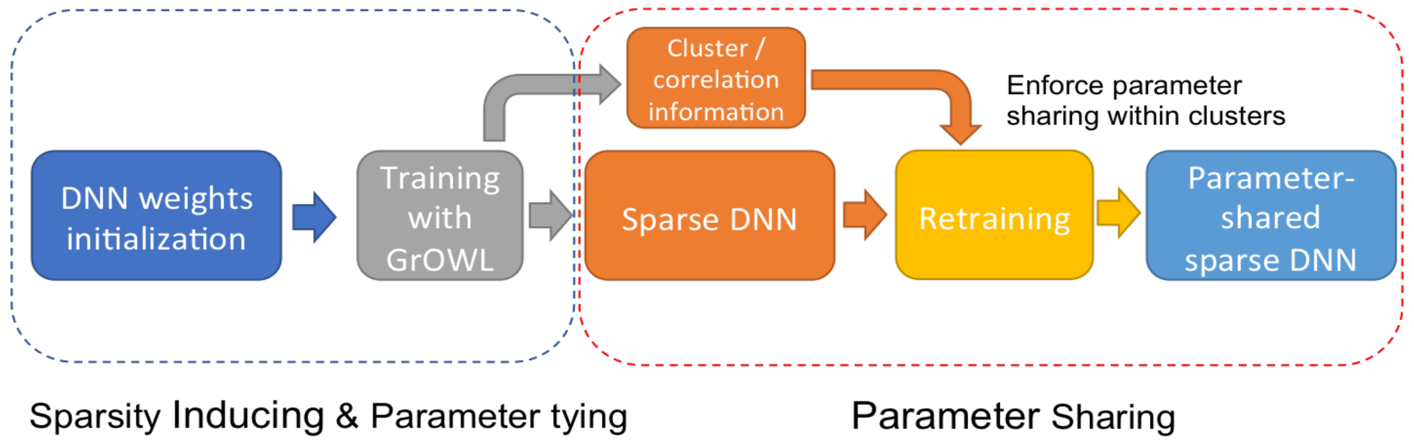
Intuition: identify the correlations only among top-p important neurons from the previous layer

$$\lambda_i = \begin{cases} \Lambda_1 + (p - i + 1)\Lambda_2, & i = 1, ..., p, \\ \Lambda_1, & i = p+1, ..., n. \end{cases}$$

- Λ_1 controls the sparsifying strength
- Λ_2 controls the correlation identification ability



Two-stage procedure



Training: simultaneously removing redundant neurons and identifying correlations among the remaining ones by tying the associated weights together; **Retraining:** keeping only the significant neurons and enforcing the learned tying structure.

Numerical Results

Single fully connected layer on MNIST

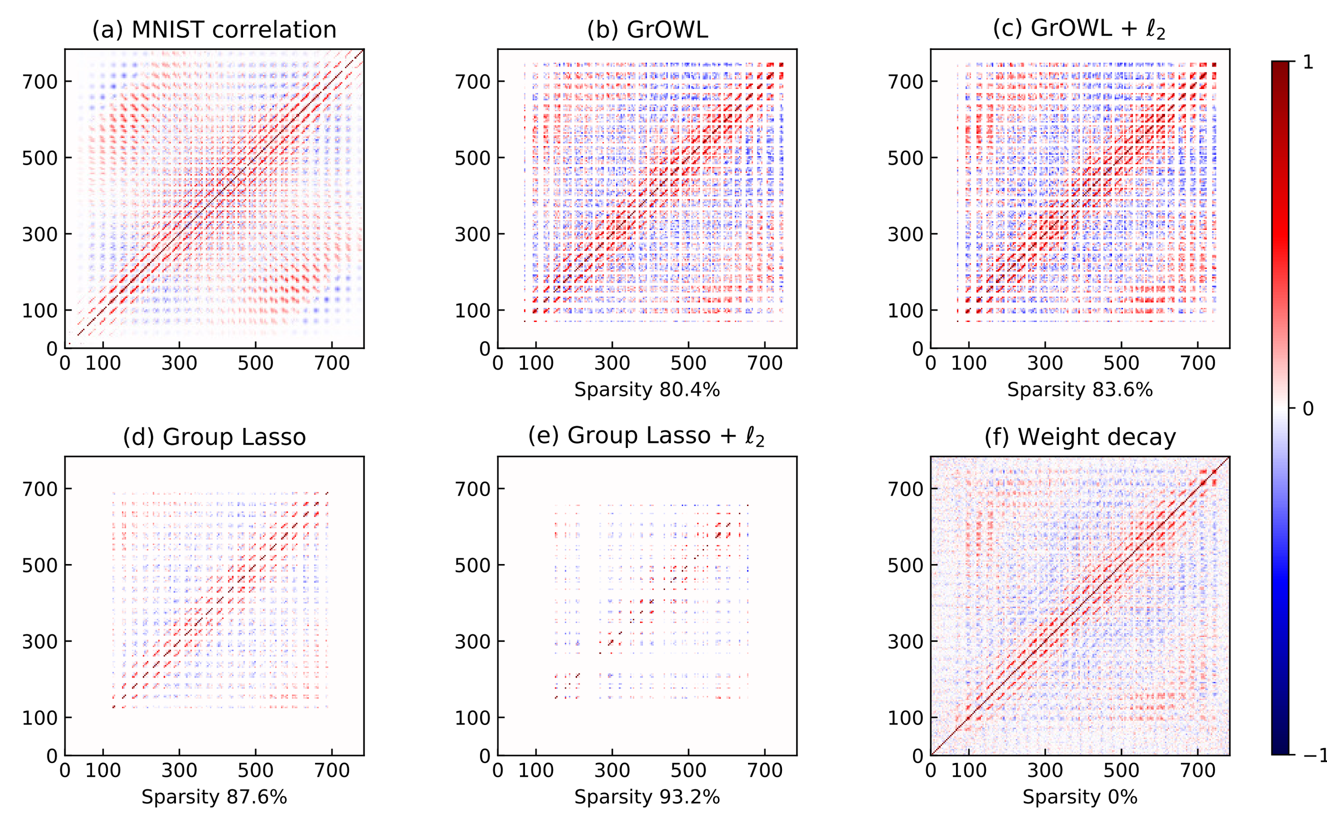
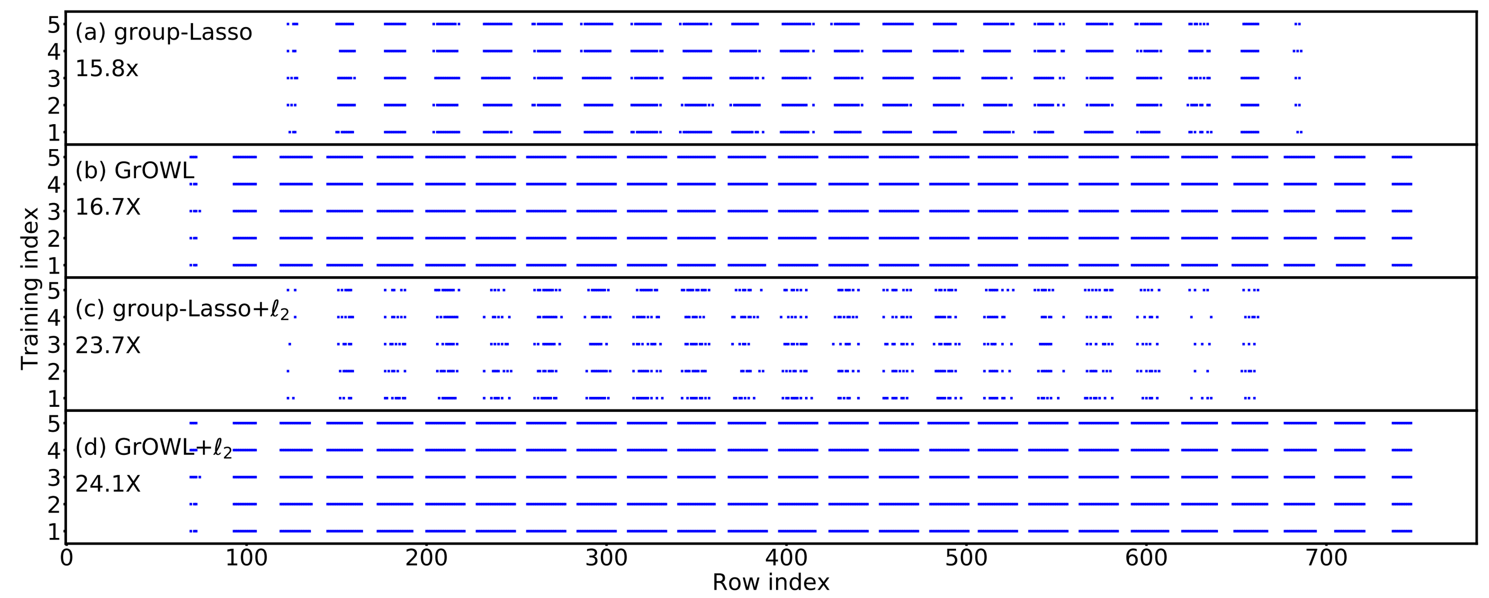


Figure: Data correlation and the correlation patterns identified by different regularizers.

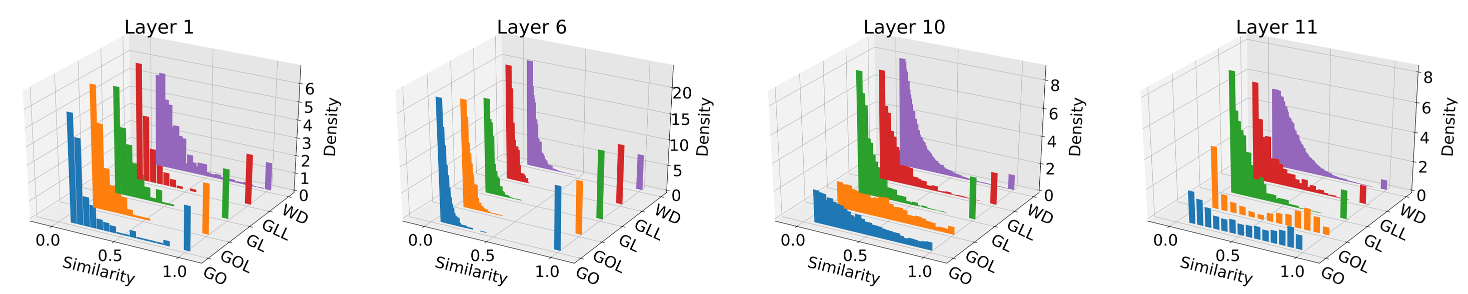
Table: Sparsity, parameter sharing, and compression rate on MNIST over 5 runs.

Regularizers	Sparsity	Parameter Sharing	Compression	Accuracy
baseline	$0.0 \pm 0\%$	1.0 ± 0	$1.0 \pm 0X$	$98.3 \pm 0.1\%$
weight decay	$0.0 \pm 0\%$	1.6 ± 0	$1.6 \pm 0X$	$98.4 \pm 0.0\%$
group-Lasso	$87.6 \pm 0.1\%$	1.9 ± 0.1	$15.8 \pm 1.0X$	$98.1 \pm 0.1\%$
group-Lasso+ℓ_2	$93.2 \pm 0.4\%$	1.6 ± 0.1	$23.7 \pm 2.1X$	$98.0 \pm 0.1\%$
GrOWL	$80.4 \pm 1.0\%$	3.2 ± 0.1	$16.7 \pm 1.3X$	$98.1 \pm 0.1\%$
GrOWL+ℓ_2	$83.6 \pm 0.5\%$	3.9 ± 0.1	$24.1 \pm 0.8X$	$98.1 \pm 0.1\%$



Sparsity pattern of the learned neural network over five runs. The mean ratio of changed indices are 11.09%, 0.59%, 32.07%, and 0.62% respectively.

VGG-16 on CIFAR-10



Output channel cosine similarity histogram. Labels: **GO**:GrOWL, **GOL**:GrOWL+ℓ_2, **GL**:group-Lasso, **GLL**:group-Lasso+ℓ_2, **WD**:weight decay.

Table: Accuracy and memory trade-off of VGG-16 on CIFAR-10 over 5 runs.

	Weight Decay	group-Lasso	group-Lasso + ℓ_2	GrOWL	GrOWL + ℓ_2
Compression	1.3 ± 0.1X	11.1 ± 0.5X	14.5 ± 0.5X	11.4 ± 0.5X	14.5 ± 0.5X
Accuracy	$93.1 \pm 0.0\%$	$92.1 \pm 0.2\%$	$92.7 \pm 0.1\%$	$92.2 \pm 0.1\%$	$92.7 \pm 0.1\%$
Baseline	Accuracy: $93.4 \pm 0.2\%$, Compression: 1.0X				

Forthcoming Research

- Applying GrOWL on more complex datasets and larger neural networks.
- Improving diversity vs sharing trade-off by encouraging sharing among smaller units: instead of predefining all 2D convolutional filters corresponding to the same input features as a group/row, apply GrOWL within each neuron by predefining each 2D convolutional filter as a group/row.