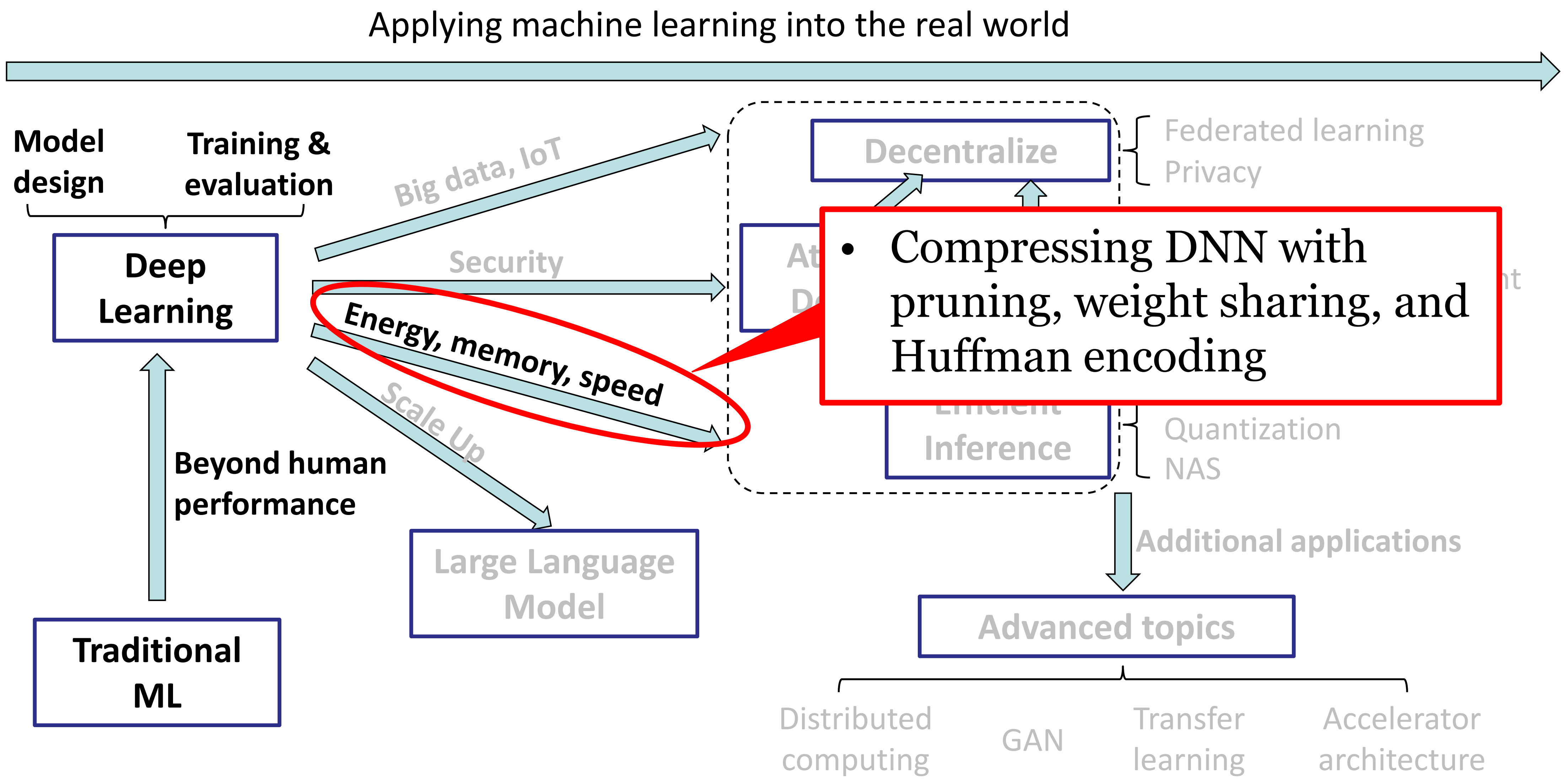
ECE 661 COMP ENG ML & DEEP NEURAL NETS

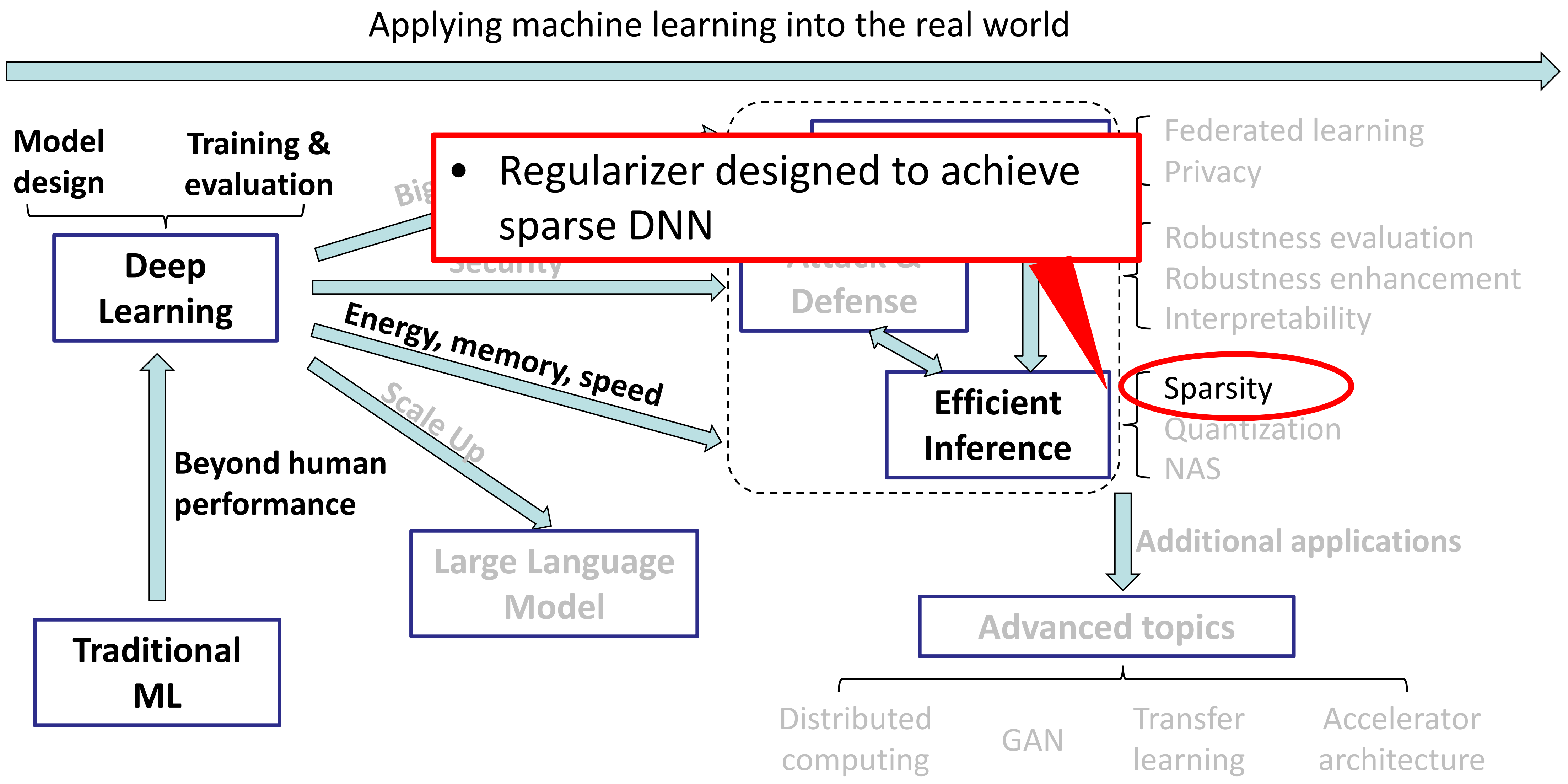
15. SPARSITY-INDUCING REGULARIZATION

HAI "HELEN" LI, SPRING 2025

Previously


Applying machine learning into the real world
Model design
Training & evaluation
Deep Learning
Big data, IoT
Security
Energy, memory, speed
Scale Up
Beyond human performance
Traditional ML
Large Language Model
Decentralize
Federated learning
Privacy
Compressing DNN with pruning, weight sharing, and Huffman encoding
Inference
Quantization
NAS
Additional applications
Advanced topics
Distributed computing
GAN
Transfer learning
Accelerator architecture

This lecture



The pursuit of sparse representations

- Efforts on searching and utilizing sparse representations started long before DNNs even existed, and is still valuable to current deep learning research (“Test of Time” award in ICML 2019)

Home » About » News »

Building Dictionaries for Machine Learning from Sparse Data

JUNE 12, 2019

Guillermo Sapiro's paper laying the foundations of modern machine learning earns a "Test of Time" award 10 years after its publication

https://pratt.duke.edu/about/news/sapiro-test-of-time
Mairal, Julien, et al. "Online dictionary learning for sparse coding." *Proceedings of the 26th annual international conference on machine learning*. ACM, 2009.

The pursuit of sparse representations

- 1921: Harold Jeffreys and Dorothy Wrinch’s simplicity principle
  - “It is found that general propositions with high probabilities must have the property of mathematical or logical simplicity”

https://icml.cc/media/Slides/icml/2019/halla(no-parent)-12-10-00-5269-online_dictiona.pdf

Jeffreys, Harold, and Dorothy Wrinch. "On certain fundamental principles of scientific enquiry." *Philosophical Magazine* 42 (1921): 269-298.

The pursuit of sparse representations

- 1921: Simplicity principle
- 1952: Harry Markowitz’s Portfolio selection
  - Finding the most efficient way of maximizing the expected return given certain level of risk

https://icml.cc/media/Slides/icml/2019/halla(no-parent)-12-10-00-5269-online_dictiona.pdf

Markowitz, Harry. "Portfolio selection." *The journal of finance*7.1 (1952): 77-91.

The pursuit of sparse representations

- 1921: Simplicity principle
- 1952: Portfolio selection
- 1960’s & 70’s: Subset selection in regression and multivariate analysis
  - Finding the smallest set of features/variables that is suitable for analysis to avoid overfitting

https://icml.cc/media/Slides/icml/2019/halla(no-parent)-12-10-00-5269-online_dictiona.pdf

The pursuit of sparse representations

- 1921: Simplicity principle
- 1952: Portfolio selection
- 1960’s & 70’s: Subset selection
- 1990’s: Signal decomposition, wavelet
  – Reconstructing the signal as a sparse combination of a set of simple basis for denoising

https://icml.cc/media/Slides/icml/2019/halla(no-parent)-12-10-00-5269-online_dictiona.pdf

The pursuit of sparse representations

- 1921: Simplicity principle
- 1952: Portfolio selection
- 1960’s & 70’s: Subset selection
- 1990’s: Signal decomposition, wavelet
- Late 90’s: Emerging theoretical results on reconstructing sparse representations
  - 1994: Basis pursuit (Chen and Donoho)
  - 1996: LASSO (Tibshirani)
  - 2004: Compressed sensing

https://icml.cc/media/Slides/icml/2019/halla(no-parent)-12-10-00-5269-online_dictiona.pdf
Tibshirani, Robert. "Regression shrinkage and selection via the lasso." *Journal of the Royal Statistical Society: Series B (Methodological)* 58.1 (1996): 267-288.
Chen, Shaobing, and David Donoho. "Basis pursuit." *Proceedings of 1994 28th Asilomar Conference on Signals, Systems and Computers*. Vol. 1. IEEE, 1994.

The pursuit of sparse representations

- Sparsity related objectives have been widely applied in various applications
  - Feature selection
  - Signal reconstruction
  - Image denoising
  - Inpainting
  - Face recognition
  - …

https://icml.cc/media/Slides/icml/2019/halla(no-parent)-12-10-00-5269-online_dictiona.pdf
Mairal, Julien, et al. "Online dictionary learning for sparse coding." *Proceedings of the 26th annual international conference on machine learning*. ACM, 2009.

Pruning (Sparsity) is Not New

- The idea of pruning neural networks dates back to the last century!
- Up to 60% of the parameters can be removed without affecting the mean squared error (MSE).
- The method requires calculating second derivatives, which is still expensive today.

Optimal Brain Damage

Yann Le Cun, John S. Denker and Sara A. Solla
AT&T Bell Laboratories, Holmdel, N. J. 07733

2.2 THE RECIPE

The OBD procedure can be carried out as follows:

1. Choose a reasonable network architecture
2. Train the network until a reasonable solution is obtained
3. Compute the second derivatives h_{kk} for each parameter
4. Compute the saliencies for each parameter: $s_k = h_{kk}u_k^2/2$
5. Sort the parameters by saliency and delete some low-saliency parameters
6. Iterate to step 2

LeCun, Yann, et al. "Optimal brain damage." *Advances in neural information processing systems* 2 (1989).

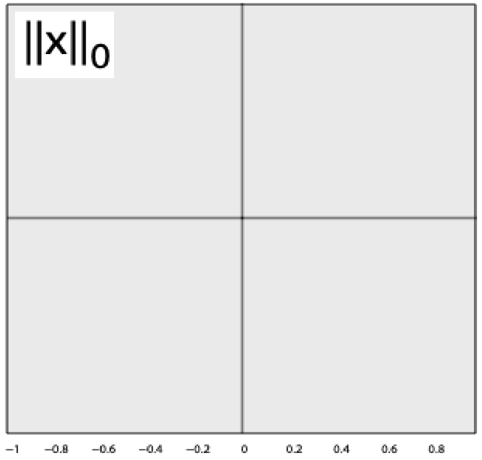
Sparsity objective

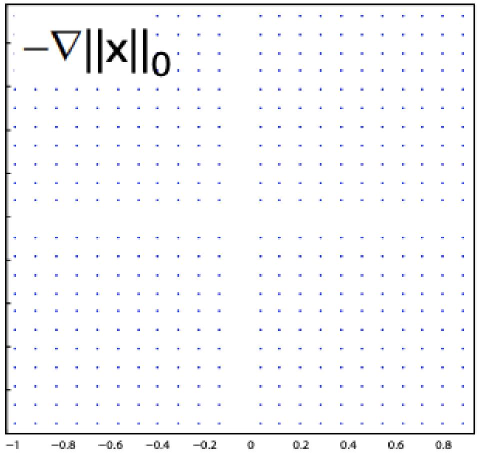
- We measure the sparsity of a set of parameters as the total number of nonzero elements
- This is the ℓ_0 norm $\left(||\cdot||_0\right)$ by definition
- For signal reconstruction
  - Given measurement A, find the sparsest signal x that can match the observation y
  - $\min_x ||x||_0 \ s.t. Ax = y$
- For deep learning
  - Finding the sparsest model without losing performance
  - $\min_\theta ||\theta||_0 \ s.t. \mathcal{L}(\theta) < \epsilon$

Having the smallest amount of non-zero weights

Dealing with ℓ_0 minimization

- The ℓ_0 norm is combinatorial, not suitable for gradient-based optimization
  - Not continuous
  - No informative gradients

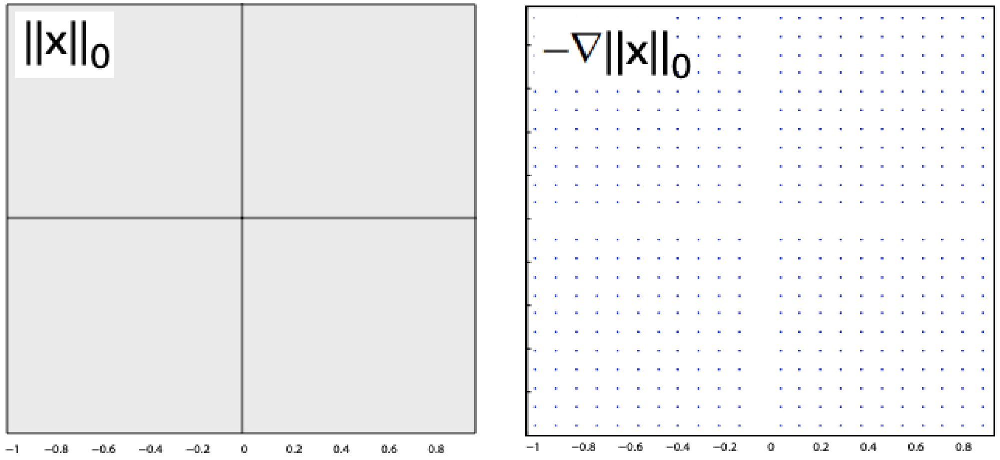




- Alternative optimization methods
  - Continuous sparsity-inducing regularizer
    - ℓ_1 norm (Lasso), nuclear norm, Hoyer, etc.
  - Proximal optimization method
    - PGD (previously introduced), proximal operator, ADMM, etc.

Dealing with ℓ_0 minimization

- The ℓ_0 norm is combinatorial, not suitable for gradient-based optimization
  - Not continuous
  - No informative gradients



- Alternative optimization methods
  - Continuous sparsity-inducing regularizer
    - ℓ_1 norm (Lasso), nuclear norm, Hoyer, etc.
  - Proximal optimization method
    - PGD (previously introduced), proximal operator, ADMM, etc.

This lecture focuses on sparsity-inducing regularizer, we will see proximal optimization-based methods later.

Simple yet effective: Lasso

- Previously, we have learnt the ℓ_2 regularization (a.k.a. ridge regression, weight decay)

$$R_2(W) = \frac{1}{2}\sum_i w_i^2, \qquad \frac{\partial R_2(W)}{\partial w_i} = w_i$$

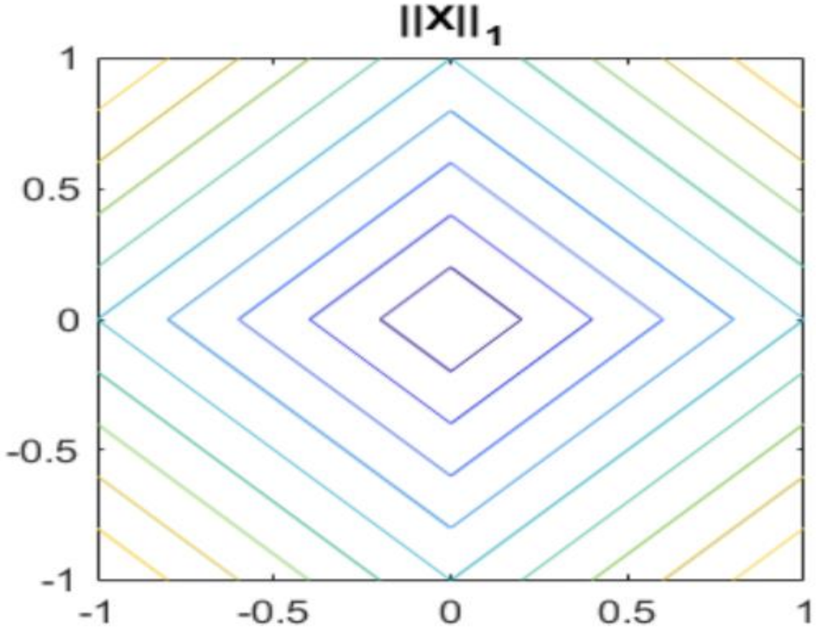
- ℓ_2 regularization can shrink all weight elements, but the shrinkage speed gets slower as the weight is close to zero, cannot induce a sparse solution
- What if we always shrink all the nonzero weight elements at a constant speed, until they reach zero?

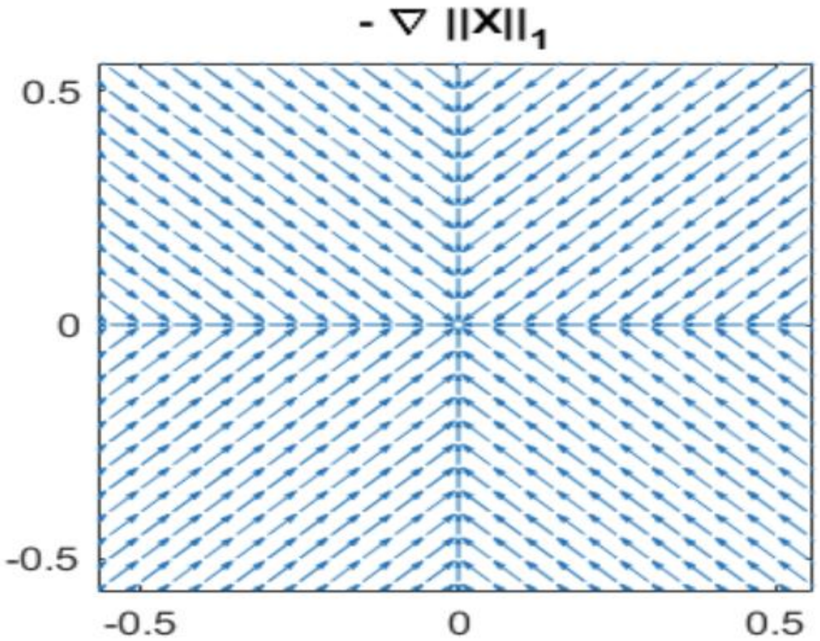
$$\frac{\partial R(W)}{\partial w_i} = \begin{cases} sign(w_i), & w_i \neq 0 \\ 0, & w_i = 0 \end{cases}$$

Simple yet effective: Lasso

$$R(W) = \sum_i |w_i|, \frac{\partial R(W)}{\partial w_i} = \begin{cases} sign(w_i), & w_i \neq 0 \\ 0, & w_i = 0 \end{cases}$$

- Easily seen that the ℓ_1 norm satisfies our requirements to the gradient
- The sparsity-inducing ability of the ℓ_1 regularization is firstly analyzed in 1996, where it's named "Lasso": Least Absolute Shrinkage and Selection Operator





Tibshirani, Robert. "Regression shrinkage and selection via the lasso." *Journal of the Royal Statistical Society: Series B (Methodological)* 58.1 (1996): 267-288.

Why Lasso works

- Objective with the Lasso regularizer

$$\min_{\theta} \mathcal{L}(\theta) + \alpha \sum_i |\theta_i|$$

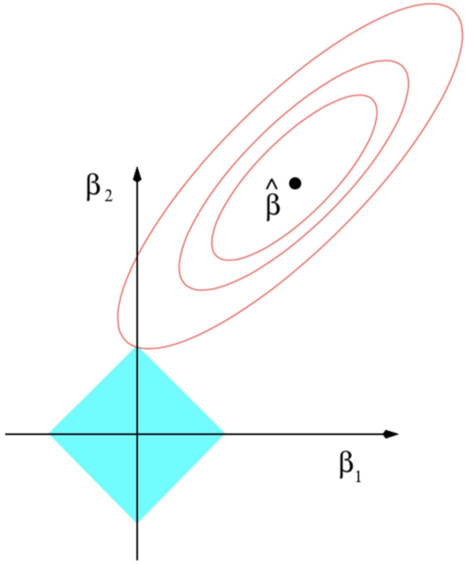
- If $\mathcal{L}(\theta)$ is convex, the objective can be equivalently written as a form of constrained optimization

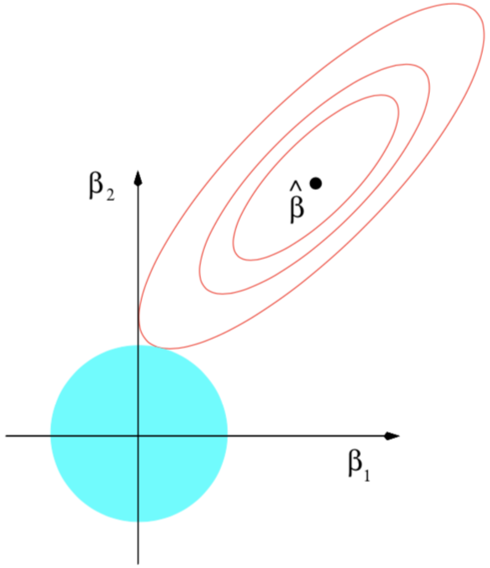
$$\min_{\theta} \mathcal{L}(\theta)\ s.t. \sum_i |\theta_i| < t$$

- The constraint enforces a “feasible area” for the possible solution, whose shape effectively affects the property of the optimization result

Why Lasso works

$$\min_{\theta} \mathcal{L}(\theta)\ s.t. \sum_i |\theta_i| < t \qquad \min_{\theta} \mathcal{L}(\theta)\ s.t. \sum_i \theta_i^2 < t$$



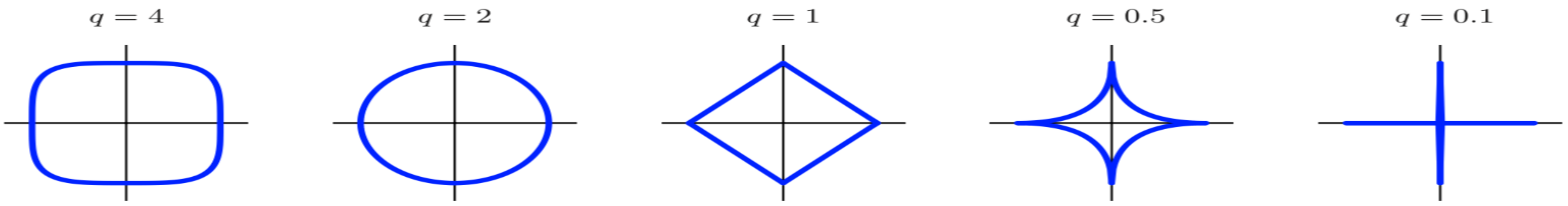


- The feasible area enforced by Lasso has “pointed” corners on the axes, leading to sparse solutions

Tibshirani, Robert, Martin Wainwright, and Trevor Hastie. *Statistical learning with sparsity: the lasso and generalizations*. Chapman and Hall/CRC, 2015.

Why Lasso works

- Among general ℓ_q norms $|w|_q = \sum_i |w_i|^q$, the feasible area induced by the ℓ_1 norm has the best property
  - There are no pointed corner when $q > 1$
  - The region is nonconvex when $q < 1$
  - Lasso is the "**closest convex relaxation**" of the ℓ_0 minimization



Tibshirani, Robert, Martin Wainwright, and Trevor Hastie. *Statistical learning with sparsity: the lasso and generalizations*. Chapman and Hall/CRC, 2015.

Lasso for DNN pruning

- As a differentiable regularization term, the Lasso regularizer can be directly added to DNN training objective, and optimized with SGD

$$\min_{W} \mathcal{L}(W) + \alpha \sum |W|_1$$

- During the training, the Lasso will automatically guide the parameters in the DNN to move towards zero
- Only one final pruning step with a small constant threshold (e.g., 1e-4) is needed to reach a sparse model, can reach similar sparse level as the iterative pruning
- Sparsity-performance tradeoff can be made by altering the regularization strength α

Lasso on structure: group Lasso

- LASSO leads to non-structured sparsity
- Recap: non-structured sparsity may not bring much speedup on traditional platforms like GPUs

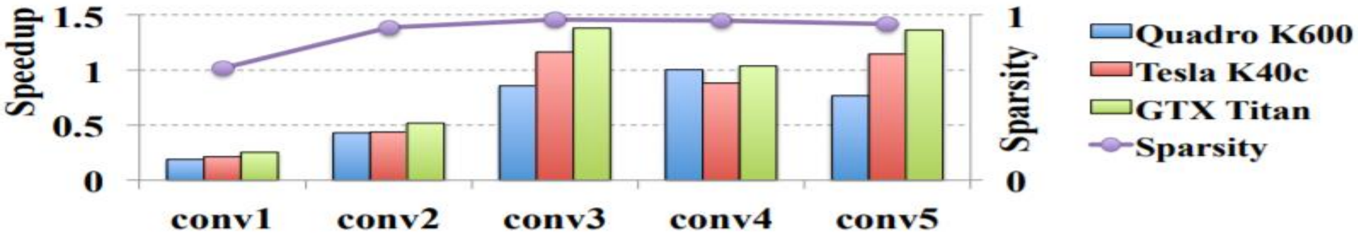


Figure 1: Evaluation speedups of AlexNet on GPU platforms and the sparsity. conv1 refers to convolutional layer 1, and so forth. Baseline is profiled by GEMM of cuBLAS. The sparse matrixes are stored in the format of Compressed Sparse Row (CSR) and accelerated by cuSPARSE.

- Structured sparsity can be achieved by having all the parameters within a structured group (i.e., channel or filter of the conv kernel) to become zero simultaneously

Wen, Wei, et al. "Learning structured sparsity in deep neural networks." *Advances in neural information processing systems*. 2016.

Lasso on structure: group Lasso

- How to have all the parameters within a group become zero simultaneously?
  - Apply ℓ_2 regularization to each group
- How to induce all-zero groups?
  - Apply ℓ_1 regularization to the ℓ_2 norms of all the groups
- The resulted regularizer is called "Group Lasso"
  - Suppose the model parameters can be divided into J groups $\theta_1, \theta_2, \ldots, \theta_J$, the Group Lasso is defined as

$$\sum_{j=1}^{J} \left|\left|\theta_j\right|\right|_2$$

Yuan, Ming, and Yi Lin. "Model selection and estimation in regression with grouped variables." *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 68.1 (2006): 49-67.

Intuition of the group Lasso

- The outer Lasso applied on the ℓ_2 norms of each group will encourage some groups' ℓ_2 norms to be 0
- For ℓ_2 norm to be 0, all elements within the group have to be 0 simultaneously, leading to structured sparsity

$$\sum_{j=1}^{J} ||\theta_j||_2$$

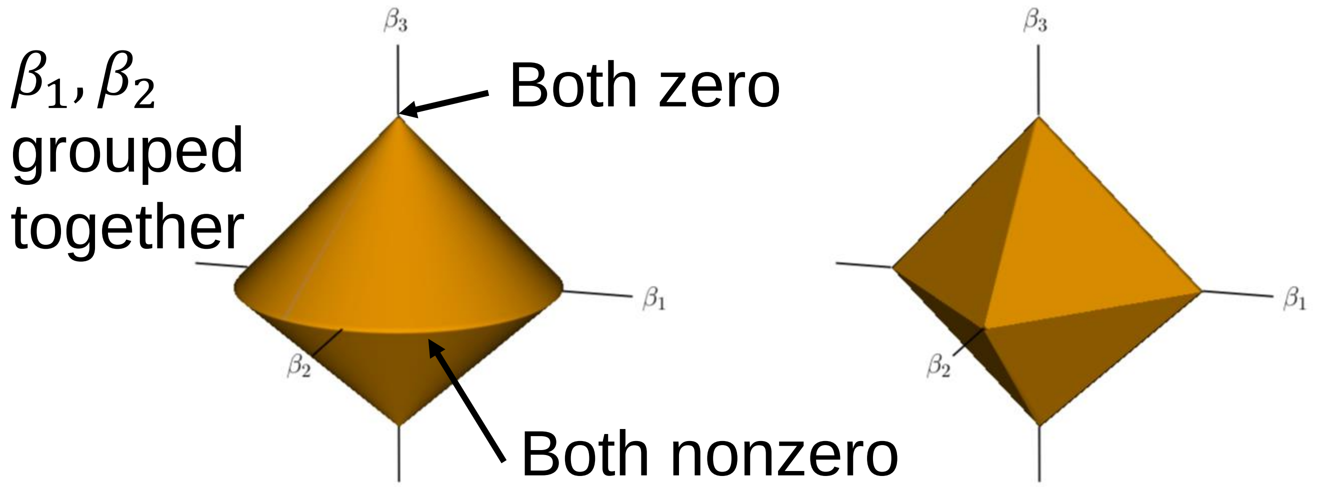
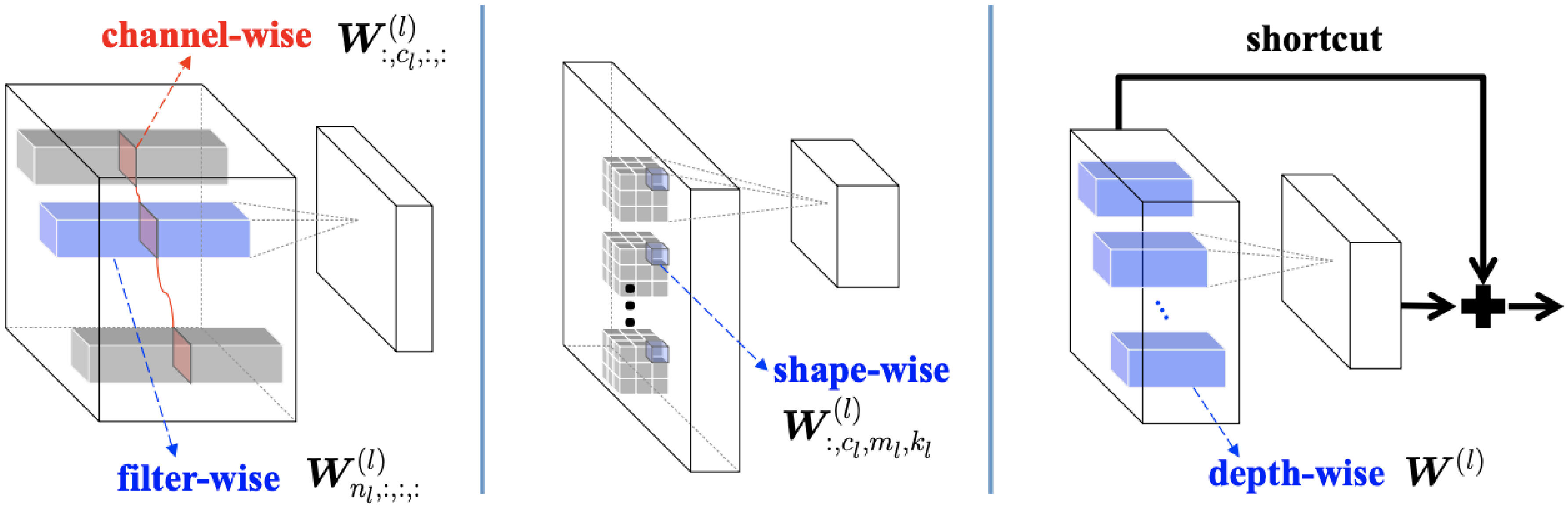


Figure 4.3 *The group lasso ball (left panel) in* $\mathbb{R}^3$*, compared to the* ℓ_1 *ball (right panel). In this case, there are two groups with coefficients* $\theta_1 = (\beta_1, \beta_2) \in \mathbb{R}^2$ *and* $\theta_2 = \beta_3 \in \mathbb{R}^1$.

Tibshirani, Robert, Martin Wainwright, and Trevor Hastie. *Statistical learning with sparsity: the lasso and generalizations*. Chapman and Hall/CRC, 2015.

Structured sparsity in CNN

- For a convolution layer, the weight should be a 4-D tensor: $W^{(l)} \in \mathbb{R}^{N_l \times C_l \times M_l \times K_l}$, where N_l is the number of filters, C_l is the number of input channels, and $M_l \times K_l$ is the convolution kernel size
- Grouping along different dimensions will lead to different pruned structures



Wen, Wei, et al. "Learning structured sparsity in deep neural networks." *Advances in neural information processing systems*. 2016.

Structured sparsity in CNN

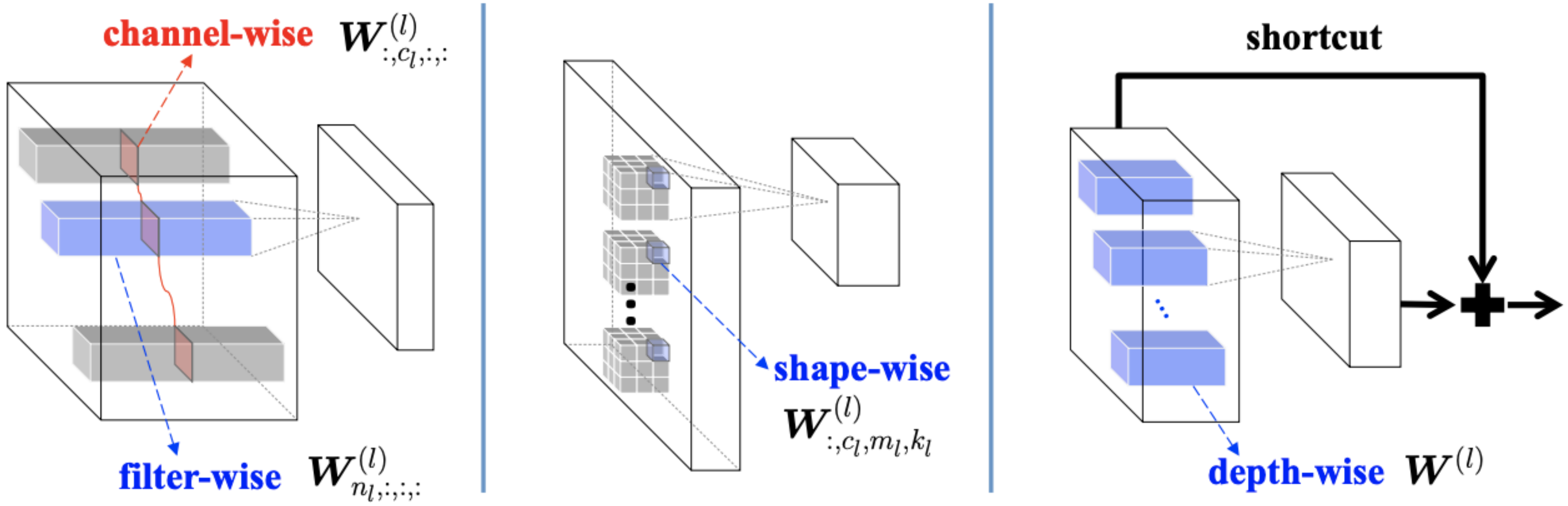
- Removing filters and channels:

$$E(\boldsymbol{W}) = E_D(\boldsymbol{W}) + \lambda_n \cdot \sum_{l=1}^{L} \left(\sum_{n_l=1}^{N_l} ||\boldsymbol{W}^{(l)}_{n_l,:,:,:}||_g \right) + \lambda_c \cdot \sum_{l=1}^{L} \left(\sum_{c_l=1}^{C_l} ||\boldsymbol{W}^{(l)}_{:,c_l,:,:}||_g \right)$$

- Modifying filter shape:

$$E(\boldsymbol{W}) = E_D(\boldsymbol{W}) + \lambda_s \cdot \sum_{l=1}^{L} \left(\sum_{c_l=1}^{C_l} \sum_{m_l=1}^{M_l} \sum_{k_l=1}^{K_l} ||\boldsymbol{W}^{(l)}_{:,c_l,m_l,k_l}||_g \right)$$

- Shortcut is added to enable whole layer removal



Wen, Wei, et al. "Learning structured sparsity in deep neural networks." *Advances in neural information processing systems*. 2016.

Experiment results with group Lasso

- LeNet on MNIST
  - Conv1: $20 \times 1 \times 5 \times 5$ – Conv2: $50 \times 20 \times 5 \times 5$

Table 1: Results after penalizing unimportant filters and channels in *LeNet*

LeNet #	Error	Filter # §	Channel # §	FLOP §	Speedup §
1 (*baseline*)	0.9%	20—50	1—20	100%—100%	1.00×—1.00×
2	0.8%	5—19	1—4	25%—7.6%	1.64×—5.23×
3	1.0%	3—12	1—3	15%—3.6%	1.99×—7.44×

§In the order of *conv1—conv2*

Table 2: Results after learning filter shapes in *LeNet*

LeNet #	Error	Filter size §	Channel #	FLOP	Speedup
1 (*baseline*)	0.9%	25—500	1—20	100%—100%	1.00×—1.00×
4	0.8%	21—41	1—2	8.4%—8.2%	2.33×—6.93×
5	1.0%	7—14	1—1	1.4%—2.8%	5.19×—10.82×

§ The sizes of filters after removing zero shape fibers, in the order of *conv1—conv2*

Wen, Wei, et al. "Learning structured sparsity in deep neural networks." *Advances in neural information processing systems*. 2016.

Experiment results with group Lasso

- AlexNet on ImageNet

Table 4: Sparsity and speedup of *AlexNet* on ILSVRC 2012

#	Method	Top1 err.	Statistics	conv1	conv2	conv3	conv4	conv5
1	ℓ_1	44.67%	sparsity	67.6%	92.4%	97.2%	96.6%	94.3%
			CPU $\times$	0.80	2.91	4.84	3.83	2.76
			GPU $\times$	0.25	0.52	1.38	1.04	1.36
2	SSL	44.66%	column sparsity	0.0%	63.2%	76.9%	84.7%	80.7%
			row sparsity	9.4%	12.9%	40.6%	46.9%	0.0%
			CPU $\times$	1.05	3.37	6.27	9.73	4.93
			GPU $\times$	1.00	2.37	4.94	4.03	3.05

- Less overall sparsity, but higher speedup on GPU

Wen, Wei, et al. "Learning structured sparsity in deep neural networks." *Advances in neural information processing systems*. 2016.

Low-rank decomposition (LoRA)

- Removing filter/channel with structured pruning will affect the shape of output feature map, may cause trouble for models with complicated short cut connections like DenseNet
- Low-rank decomposition simplify DNN structure without influencing output feature map shape
  - Decomposing 1 layer into 2 low-rank layers, with smaller channel/spatial dimension in between
  - Similar to the depthwise separable convolution in MobileNet-V1
- Also used in parameter-efficient finetuning of DNNs via the low rank adaptation (LoRA) method.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, & Weizhu Chen. (2021). LoRA: Low-Rank Adaptation of Large Language Models.

Low-rank decomposition

- 2D Matrix decomposition: SVD

$$W^t = \sum_{i=1}^{rank(W^t)} \sigma_i \cdot U_i \cdot (V_i)^T$$

  - $m \times n \rightarrow (m \times r) \cdot (r \times n)$
  - **Singular values** σ_i: singular values close to zero will be discarded to reduce the rank of the decomposed matrices
- 4D tensor decomposition (Conv layers)
  - Reshape -> SVD -> reshape
  - Channel-wise decomposition

$$O \times I \times H \times W \rightarrow O \times IHW \rightarrow (O \times r) \cdot (r \times IHW) \rightarrow$$
$$r \times I \times H \times W \circ O \times r \times 1 \times 1$$

  - Spatial-wise decomposition

$$O \times I \times H \times W \rightarrow OH \times IW \rightarrow (OH \times r) \cdot (r \times IW) \rightarrow$$
$$r \times I \times 1 \times W \circ O \times r \times H \times 1$$

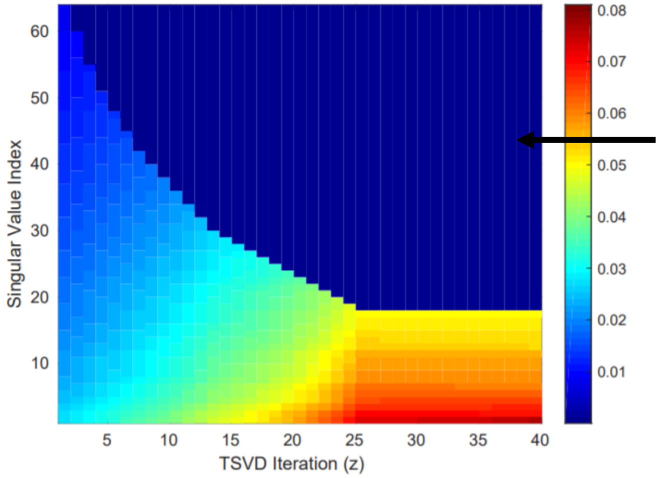
Rank reduction: Nuclear norm

- Lower rank r leads to higher compression rate
  - Need to induce sparsity on the singular values
- Nuclear norm: LASSO on singular values

$$\min\left\{f(x;w) + \lambda \sum_{l=1}^{L} ||W_l||_*\right\} \quad ||W_l||_* = \sum_{i=1}^{rank(W_l)} \sigma_l^i$$

  - Not differentiable due to the SVD process, but can be optimized with approximations (see cited paper for detail)

Least important



Most important

Zero singular values will be discarded for compression

Xu, Yuhui, et al. "TRP: Trained Rank Pruning for Efficient Deep Neural Networks." *arXiv preprint arXiv:2004.14566* (2020).

Figure 2: Visualization of rank selection, taken from the res3-1-2 convolution layer in ResNet-20 trained on CIFAR-10.

Quick summary of Lasso

- Closest convex relaxation of ℓ_0, leads to sparsity
- Advantages
  - Differentiable, convex, easy to optimize
  - Has theoretical guarantee on sparse signal reconstruction
- Disadvantages
  - Shrinking all the variables (including those large and important ones) with same speed, lead to "**biased estimation**" on non-zero parameters (will revisit for proximal operator next lecture)
  - Theoretical guarantee on sparse signal recovery only holds for convex optimization. There could be sub-optimal performance on nonconvex tasks like deep learning

Yin, Penghang, Ernie Esser, and Jack Xin. "Ratio and difference of l1 and l2 norms and sparse representation with coherent dictionaries." *Commun. Inform. Systems* 14.2 (2014): 87-109.

Beyond Lasso: Hoyer and more

- Beyond Lasso, multiple sparsity measures have been discovered and analyzed throughout the years of compressed sensing research
- Not much have been applied as a sparsity-inducing regularizer in deep learning, but could potentially produce superior results
- The Hoyer regularizer has beaten Lasso in recent DNN pruning research

TABLE I
COMMONLY USED SPARSITY MEASURES MODIFIED TO BECOME MORE POSITIVE FOR INCREASING SPARSITY.

Measure	Definition
ℓ^0	$\#\{j, c_j = 0\}$
ℓ^0_ϵ	$\#\{j, c_j \leq \epsilon\}$
$-\ell^1$	$-\left(\sum_j c_j\right)$
$-\ell^p$	$-\left(\sum_j c_j^p\right)^{1/p}, \quad 0 < p < 1$
$\frac{\ell^2}{\ell^1}$	$\frac{\sqrt{\sum_j c_j^2}}{\sum_j c_j}$
$-\tanh_{a,b}$	$-\sum_j \tanh\left((ac_j)^b\right)$
$-\log$	$-\sum_j \log\left(1 + c_j^2\right)$
κ_4	$\frac{\sum_j c_j^4}{\left(\sum_j c_j^2\right)^2}$
u_θ	$1 - \min_{i=1,2,\ldots,N-\lceil\theta N\rceil+1} \frac{c_{(i+\lceil\theta N\rceil-1)} - c_{(i)}}{c_{(N)} - c_{(1)}}$ s.t. $\lceil\theta N\rceil \neq N$ for ordered data, $c_{(1)} \leq c_{(2)} \leq \cdots \leq c_{(N)}$
$-\ell^p_-$	$-\sum_{j, c_j \neq 0} c_j^p, \quad p < 0$
H_G	$-\sum_j \log c_j^2$
H_S	$-\sum_j \tilde{c_j} \log \tilde{c_j}^2$ where $\tilde{c_j} = \frac{c_j^2}{\|\vec{c}\|_2^2}$
H'_S	$-\sum_j c_j \log {c_j}^2$
Hoyer	$(\sqrt{N} - \frac{\sum_j c_j}{\sqrt{\sum_j c_j^2}})(\sqrt{N} - 1)^{-1}$
Gini	$1 - 2\sum_{k=1}^{N} \frac{c_{(k)}}{\|\vec{c}\|_1} \left(\frac{N-k+\frac{1}{2}}{N}\right)$ for ordered data, $c_{(1)} \leq c_{(2)} \leq \cdots \leq c_{(N)}$

Hurley, Niall, and Scott Rickard. "Comparing measures of sparsity." *IEEE Transactions on Information Theory* 55.10 (2009): 4723-4741.

The Hoyer regularizer

- The Hoyer regularizer is defined as the ratio of the ℓ_1 norm and the ℓ_2 norm of a vector, originally proposed for blind deconvolution (image deblur)

$$R(X) = \frac{\sum_i |x_i|}{\sqrt{\sum_i x_i^2}}$$

- Properties of the Hoyer regularizer
  - Bounded
  - Scale-invariant

$$1 \leq \frac{\sum_i |x_i|}{\sqrt{\sum_i x_i^2}} \leq \sqrt{n}, \;\; \forall X \in \mathbb{R}^n.$$

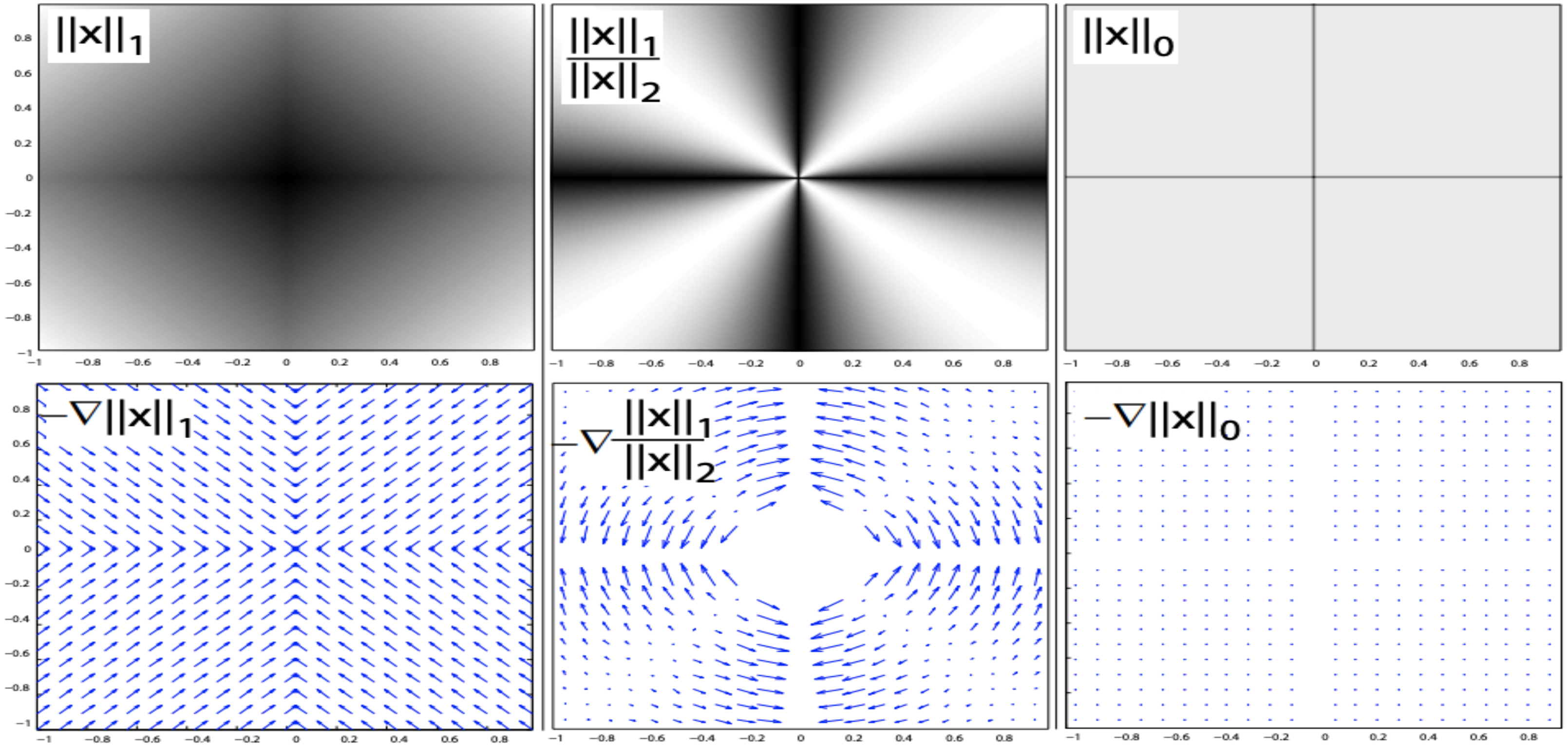
$$R(\alpha X) = R(X)$$

  - Minimized when only 1 element is nonzero, maximized when all elements are equal

Krishnan, Dilip, Terence Tay, and Rob Fergus. "Blind deconvolution using a normalized sparsity measure." *CVPR 2011*. IEEE, 2011.

Comparing with ℓ_1 and ℓ_0

- Almost everywhere differentiable as the ℓ_1 norm, but Hoyer's gradient is radial, which can preserve the scale of the original vector, protect variance
- Having similar minima structure as the ℓ_0 norm, the square of Hoyer will have the same range as ℓ_0: [1,n]



Krishnan, Dilip, Terence Tay, and Rob Fergus. "Blind deconvolution using a normalized sparsity measure." *CVPR 2011*. IEEE, 2011.

Behavior of the Hoyer-Square

- The Hoyer-Square has the same range and a similar minima structure as the ℓ_0 norm, can be considered as a continuous approximation of the ℓ_0

$$H_S(W) = \frac{(\sum_i |w_i|)^2}{\sum_i w_i^2}.$$

- Almost everywhere differentiable, can be optimized directly with gradient descent

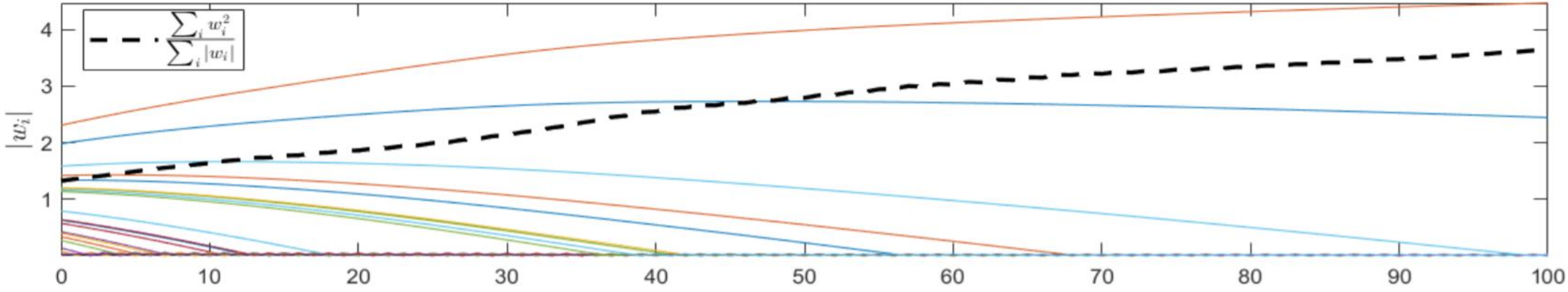
- The gradient of the Hoyer-Square regularizer shows an automatic "trimming" effect that protects larger weight elements while penalizing smaller ones

$$\partial_{w_j} H_S(W) = 2sign(w_j)\frac{\sum_i |w_i|}{(\sum_i w_i^2)^2}(\sum_i w_i^2 - |w_j|\sum_i |w_i|).$$

Behavior of the Hoyer-Square

$$\partial_{w_j} H_S(W) = 2sign(w_j)\frac{\sum_i |w_i|}{(\sum_i w_i^2)^2}\underline{\underline{(\sum_i w_i^2 - |w_j|\sum_i |w_i|)}}.$$

- Gradient descent path, dash line shows the inferred “trimming” threshold



  - Gradually extend trimming threshold as more weights coming close to zero

Yang, Huanrui, Wei Wen, and Hai Li. "DeepHoyer: Learning Sparser Neural Network with Differentiable Scale-Invariant Sparsity Measures." *International Conference on Learning Representations*. 2020.

Effectiveness on sparse DNN

- Hoyer-Square can beat ℓ_1-based and other optimization-based pruning methods, including Hoyer

Table 2: Element-wise pruning results on LeNet-5 model @ accuracy 99.2%

Method	Nonzero wights left after pruning				
	Total	CONV1	CONV2	FC1	FC2
Orig	430.5k	500	25k	400k	5k
(Han et al., 2015b)	36k (8%)	330 (66%)	3k (12%)	32k (8%)	950 (19%)
(Zhang et al., 2018)	6.1k (1.4%)	100 (20%)	2k (8%)	3.6k (0.9%)	350 (7%)
(Lee et al., 2019)	8.6k (2.0%)	Not reported in (Lee et al., 2019)			
(Ma et al., 2019)[1]	5.4k (1.3%)	100 (20%)	690 (2.8%)	4.4k (1.1%)	203 (4.1%)
Hoyer	4.0k (0.9%)	**53 (10.6%)**	**613 (2.5%)**	3.2k (0.8%)	**136 (2.7%)**
Hoyer-Square	**3.5k (0.8%)**	67 (13.4%)	848 (3.4%)	**2.4k (0.6%)**	234 (4.7%)

Especially effective for large FC layers

Table 7: Element-wise pruning results on AlexNet without accuracy loss. Refer to Table 2 for the full reference of the mentined methods.

Layer	Nonzero wights left after pruning					
	Baseline	Han et al.	Zhang et al.	Ma et al.	Hoyer	HS
CONV1	34.8K	29.3K	28.2K	24.2K	**21.3K**	31.6K
CONV2	307.2K	116.7K	**61.4K**	109.9K	77.2K	148.4K
CONV3	884.7K	309.7K	**168.1K**	241.2K	192.0K	299.3K
CONV4	663.5K	245.5K	**132.7K**	207.4K	182.6K	275.6K
CONV5	442.2K	163.7K	**88.5K**	134.7K	116.6K	197.1K
FC1	37.7M	3.40M	1.06M	**0.763M**	1.566M	**0.781M**
FC2	16.8M	1.51M	0.99M	1.070M	0.974M	**0.650M**
FC3	4.10M	1.02M	0.38M	0.505M	0.490M	**0.472M**
Total	60.9M	6.8M	2.9M	3.05M	3.62M	**2.85M**

Yang, Huanrui, Wei Wen, and Hai Li. "DeepHoyer: Learning Sparser Neural Network with Differentiable Scale-Invariant Sparsity Measures." *International Conference on Learning Representations*. 2020.

Extending to group regularization

- Recall the Group Lasso $\sum_{j=1}^{J} \left|\left|\theta_j\right|\right|_2$, which is the Lasso over the ℓ_2 norms of each group
- Follow the same intuition, we can replace the outer Lasso in the Group Lasso with the Hoyer-Square regularizer, which will lead to the Group-HS regularizer that can be used for structural pruning

$$G_H(W) = \frac{(\sum_{g=1}^{G} ||w^{(g)}||_2)^2}{\sum_{g=1}^{G} ||w^{(g)}||_2^2} = \frac{(\sum_{g=1}^{G} ||w^{(g)}||_2)^2}{||W||_2^2}.$$

Effectiveness of Group-HS

- Can be applied the same as the Group Lasso
- Outperform the Pareto frontier of performance-#FLOPs tradeoff

Table 4: Structural pruning results on LeNet-300-100 model

Method	Accuracy	#FLOPs	Pruned structure
Orig	98.4%	266.2k	784-300-100
Sparse VD (Molchanov et al., 2017)	98.2%	67.3k (25.28%)	512-114-72
BC-GNJ (Louizos et al., 2017a)	98.2%	28.6k (10.76%)	278-98-13
BC-GHS (Louizos et al., 2017a)	98.2%	28.1k (10.55%)	311-86-14
$\ell_{0_{hc}}$ (Louizos et al., 2017b)	98.2%	26.6k (10.01%)	266-88-33
Bayes $\ell_{1_{trim}}$ (Yun et al., 2019)	98.3%	20.5k (7.70%)	245-75-25
Group-HS	98.2%	**16.5k (6.19%)**	353-45-11

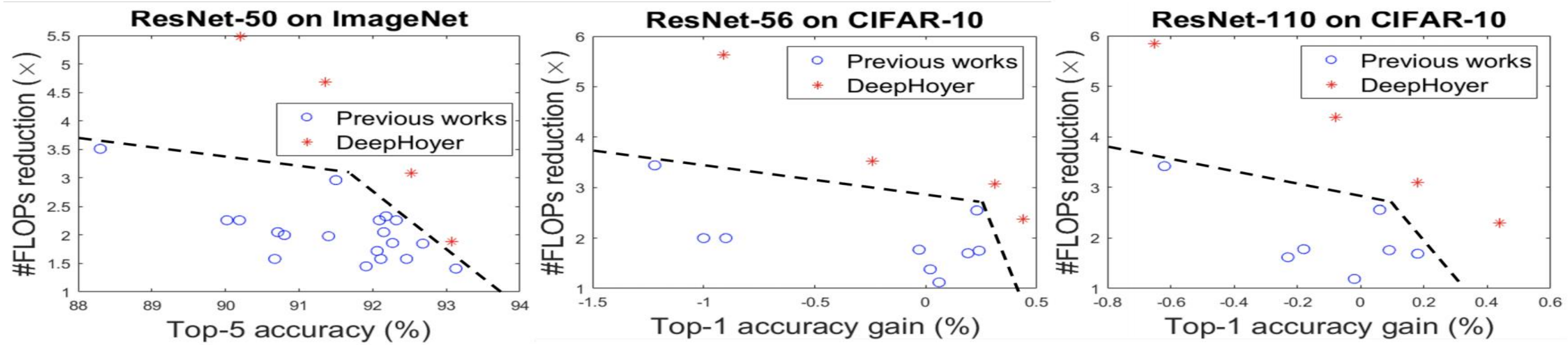


Figure 3: Comparisons of accuracy-#FLOPs tradeoff on ImageNet and CIFAR-10, black dash lines mark the Pareto frontiers. The exact data for the points are listed in **Appendix C.3**.

Yang, Huanrui, Wei Wen, and Hai Li. "DeepHoyer: Learning Sparser Neural Network with Differentiable Scale-Invariant Sparsity Measures." *International Conference on Learning Representations*. 2020.

Summary

- The ℓ_0 norm is not suitable for direct optimization
- Alternative methods learned so far:
  - Continuous sparsity-inducing regularizer
    - ℓ_1 norm (Lasso), Group Lasso, Nuclear norm, Hoyer/Hoyer-Square/Group-HS, etc.
- Coming up next:
  - Proximal optimization method
    - PGD (previously introduced), proximal operator
  - Dynamic neural network model
    - Channel gating, early exit etc.