Learning Sparsity in Neural Networks and Robust Statistical Analysis

Yanwei Fu, Yikai Wang, Xinwei Sun
School of Data Science
Fudan University

Yuan Yao
HKUST

Wotao Yin
DAMO Academy
Alibaba


Yanwei Fu


Yikai Wang


Xinwei Sun


Yuan Yao


Wotao Yin

https://sparse-learning.github.io

Learning Sparsity in Neural Networks and Robust Statistical Analysis
Lecture 1:

Yanwei Fu

School of Data Science

Fudan University

http://yanweifu.github.io

Motivation

Overview of

ECCV2012 tutorial:

Sparse and Low-Rank Recovery of *Data*

Tutorial on Sparse and Low-rank Modeling, European Conference on Computer Vision, Firenze, Italy, October 2012

Overview of

ECCV2012 tutorial:

Sparse and Low-Rank Recovery of *Data*

Tutorial on Sparse and Low-rank Modeling, European Conference on Computer Vision, Firenze, Italy, October 2012

Sparse and Low-Rank Representation
Lecture I: Motivation and Theory

Yi Ma — MSRA and UIUC
Allen Yang — UC Berkeley
John Wright — Columbia University

European Conference on Computer Vision, October 7, 2012

Our CVPR2022 tutorial:

Prerecorded Sessions			
8:30 - 8:40	**Opening Remarks**	Virtual	Yanwei Fu
8:40 - 9:10	**Introduction**	Virtual	Yanwei Fu
9:10 - 9:30	**Sparsity Learning in Noisy Data Detection**	Virtual	Yanwei Fu and Yikai Wang
9:30 - 10:15	**Inverse Scale Space method and Statistical Properties**	On-site	Yuan Yao
10:15 - 10:30	**Break**		
10:30 - 11:00	**Sparsity Learning in Medical Imaging**	Virtual	Xinwei Sun
11:00 - 12:00	**Learning to Optimize**	On-site	Wotao Yin

https://sparse-learning.github.io

Overview of

ECCV2012 tutorial:

Sparse and Low-Rank Recovery of *Data*

Tutorial on Sparse and Low-rank Modeling, European Conference on Computer Vision, Firenze, Italy, October 2012

Sparse and Low-Rank Representation
Lecture I: Motivation and Theory

Yi Ma MSRA and UIUC

Allen Yang UC Berkeley

John Wright Columbia University

European Conference on Computer Vision, October 7, 2012

Our CVPR2022 tutorial:

Learning Sparsity in Labels/Data for Robust Statistical Analysis

Sparse data/label learning

Prerecorded Sessions			
8:30 - 8:40	**Opening Remarks**	Virtual	Yanwei Fu
8:40 - 9:10	**Introduction**	Virtual	Yanwei Fu
9:10 - 9:30	**Sparsity Learning in Noisy Data Detection**	Virtual	Yanwei Fu and Yikai Wang
9:30 - 10:15	**Inverse Scale Space method and Statistical Properties**	On-site	Yuan Yao
10:15 - 10:30	**Break**		
10:30 - 11:00	**Sparsity Learning in Medical Imaging**	Virtual	Xinwei Sun
11:00 - 12:00	**Learning to Optimize**	On-site	Wotao Yin

https://sparse-learning.github.io

Overview of

ECCV2012 tutorial:

Sparse and Low-Rank Recovery of *Data*

Tutorial on Sparse and Low-rank Modeling, European Conference on Computer Vision, Firenze, Italy, October 2012

Our CVPR2022 tutorial:

Learning Sparsity in Labels/Data for Robust Statistical Analysis

Sparse data/label learning

Learning Sparsity in Deep Models for Compressive Neural Networks

Learning the sparse model

Prerecorded Sessions			
8:30 - 8:40	Opening Remarks	Virtual	Yanwei Fu
8:40 - 9:10	Introduction	Virtual	Yanwei Fu
9:10 - 9:30	Sparsity Learning in Noisy Data Detection	Virtual	Yanwei Fu and Yikai Wang
9:30 - 10:15	Inverse Scale Space method and Statistical Properties	On-site	Yuan Yao
10:15 - 10:30	Break		
10:30 - 11:00	Sparsity Learning in Medical Imaging	Virtual	Xinwei Sun
11:00 - 12:00	Learning to Optimize	On-site	Wotao Yin

https://sparse-learning.github.io

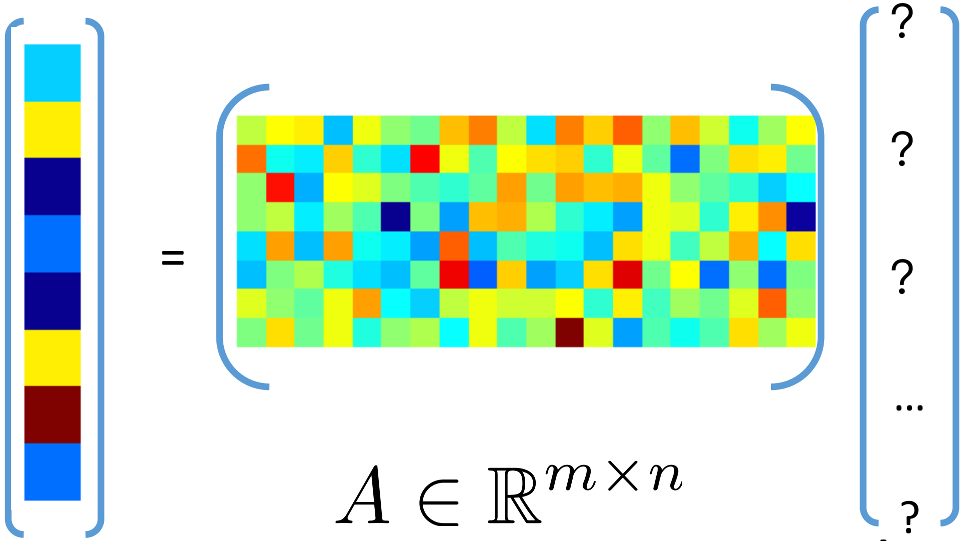
Sparse and Low-Rank Recovery of *Data*

Sparse and Low-Rank Recovery of *Data*

Underdetermined Linear system $y = Ax$

Sparse and Low-Rank Recovery of *Data*

Underdetermined Linear system $y = Ax$



$A \in \mathbb{R}^{m \times n}$

Observation $y \in \mathbb{R}^m$

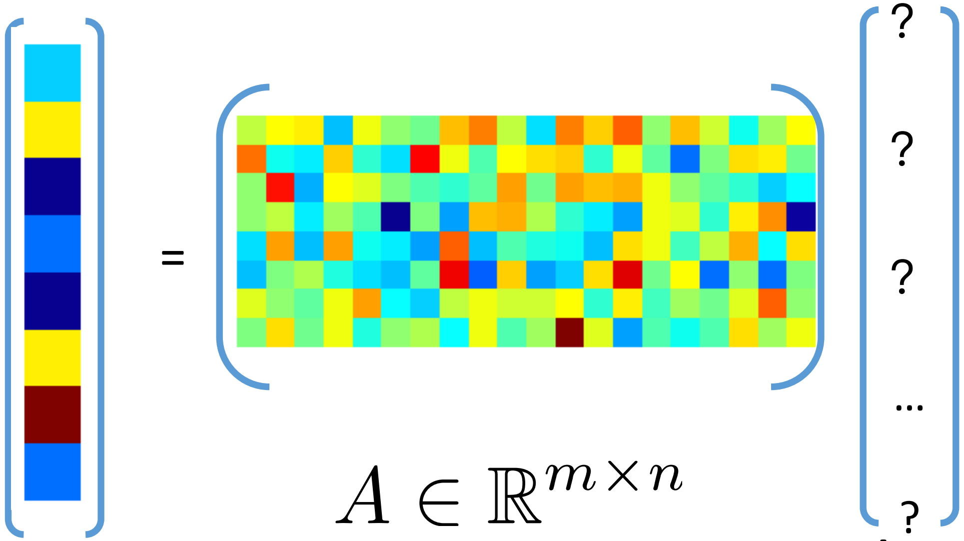
Unknown $x \in \mathbb{R}^n$

$m \ll n$

 # Observations **# unknowns**

Sparse and Low-Rank Recovery of *Data*

Underdetermined Linear system $y = Ax$



$A \in \mathbb{R}^{m \times n}$

Observation $y \in \mathbb{R}^m$

Unknown $x \in \mathbb{R}^n$

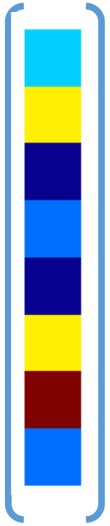
$m \ll n$

 # Observations **# unknowns**

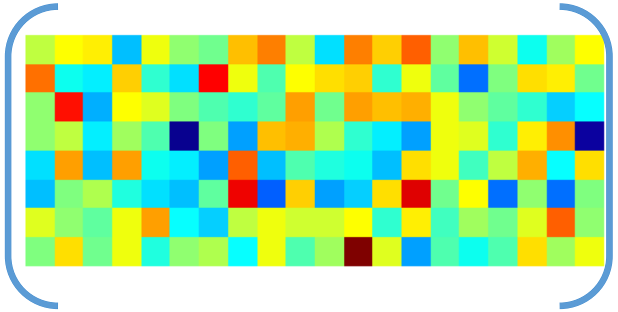
[Wainwright, et al. IEEE TIT 2009]

Sparse and Low-Rank Recovery of *Data*

Underdetermined Linear system $y = Ax$



=



$A \in \mathbb{R}^{m \times n}$

?
?
?
...
?

Observation $y \in \mathbb{R}^m$

$m \ll n$

 # Observations **# unknowns**

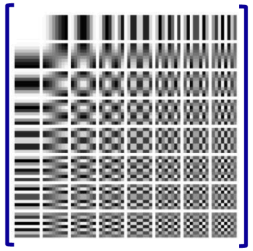
Unknown $x \in \mathbb{R}^n$

[Wainwright, et al. IEEE TIT 2009]

Compression:

y

$\approx$



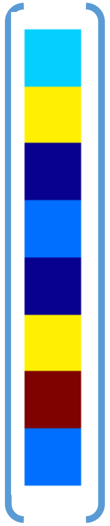
(Patches of) … input image

A DCT basis

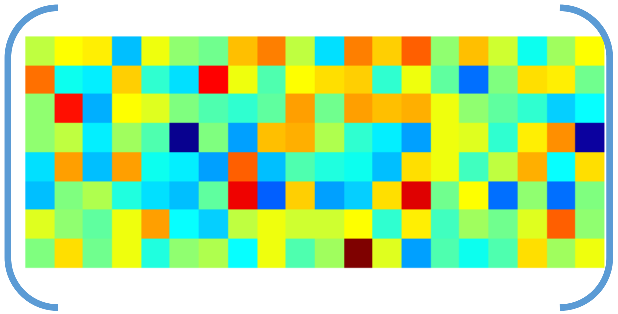
x coefficients

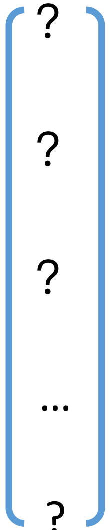
Sparse and Low-Rank Recovery of *Data*

Underdetermined Linear system $y = Ax$



=





$A \in \mathbb{R}^{m \times n}$

Observation

$y \in \mathbb{R}^m$

$m \ll n$

 # Observations **# unknowns**

Unknown

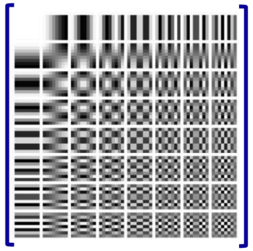
$x \in \mathbb{R}^n$

[Wainwright, et al. IEEE TIT 2009]

Compression:

$\boldsymbol{y}$

$\approx$



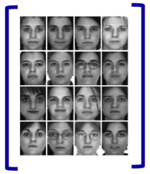
(Patches of) … input image

$\boldsymbol{A}$ DCT basis

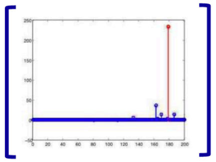
$\boldsymbol{x}$ coefficients

Recognition:

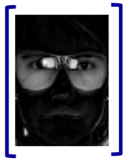
$=$



$\times$



$+$



$\boldsymbol{y} \in \mathbb{R}^m$
Test image

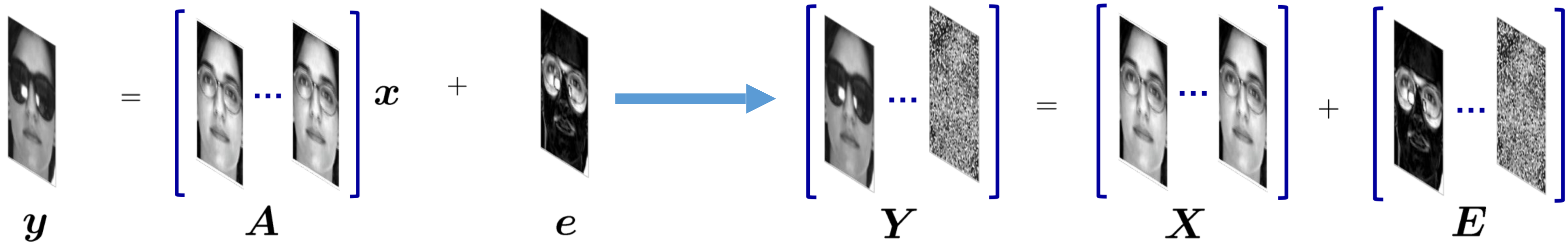
$\boldsymbol{A} = [\boldsymbol{A}_1 \mid \boldsymbol{A}_2 \mid \cdots \mid \boldsymbol{A}_k]$
Combined training dictionary

$\boldsymbol{x} \in \mathbb{R}^n$
coefficients

$\boldsymbol{e} \in \mathbb{R}^m$
corruption, occlusion

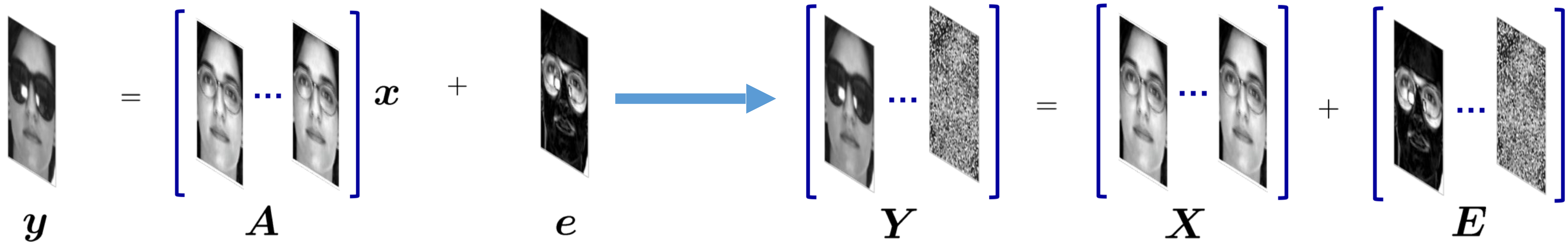
Sparse and Low-Rank Recovery of *Data* (Cont.)

From recovering a *single sparse vector* to recovering low-rank matrix (many correlated vectors):

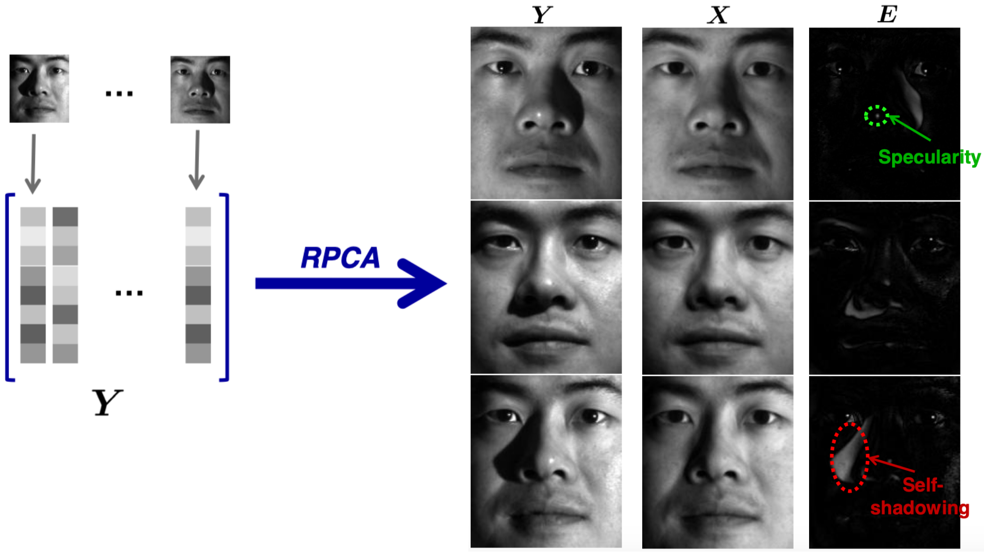


Sparse and Low-Rank Recovery of *Data* (Cont.)

From recovering a *single sparse vector* to recovering low-rank matrix (many correlated vectors):

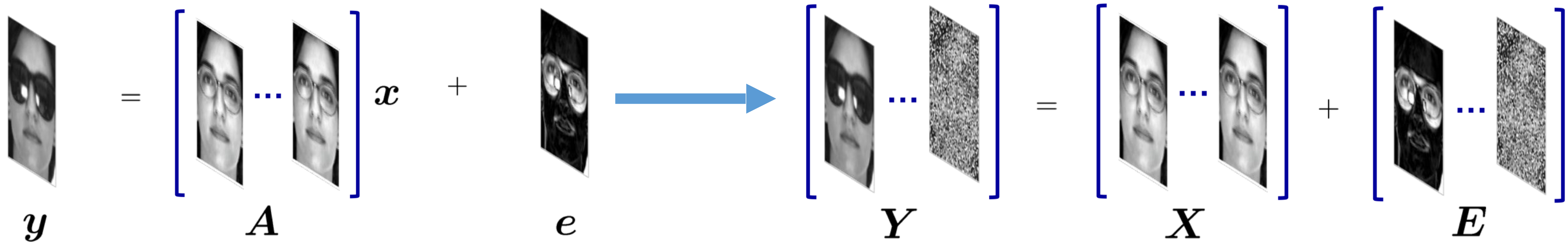


Faces under varying illumination:

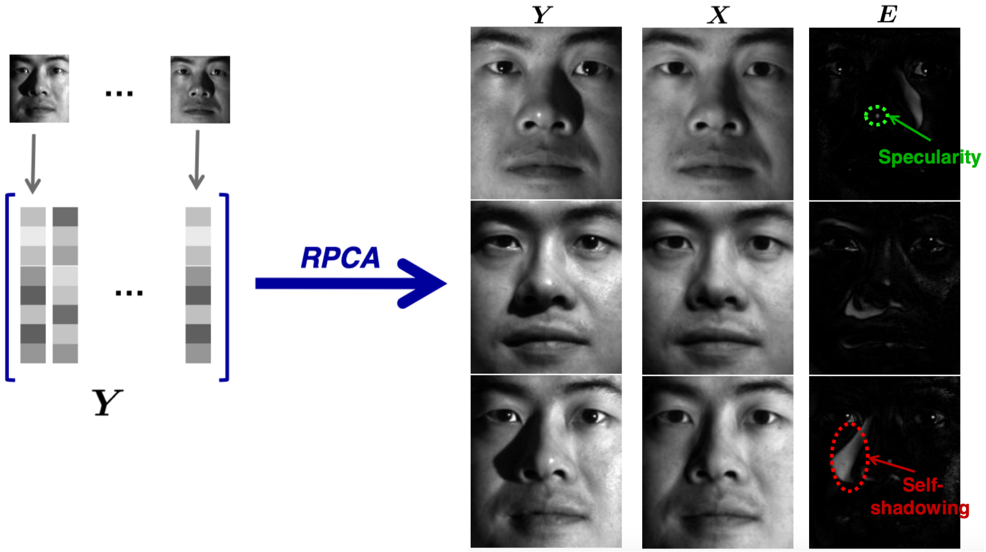


Sparse and Low-Rank Recovery of *Data* (Cont.)

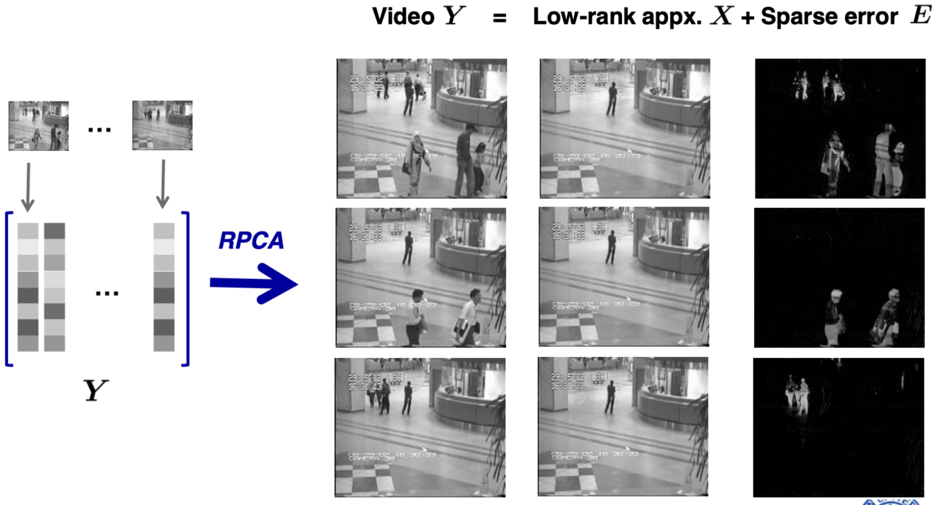
From recovering a *single sparse vector* to recovering low-rank matrix (many correlated vectors):



Faces under varying illumination:



Background modeling from video :

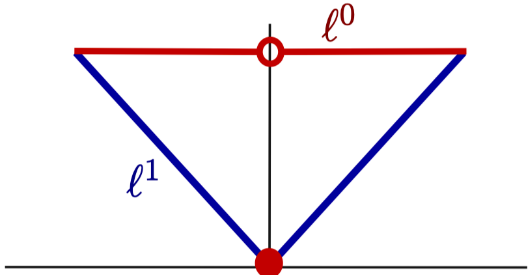


Part of content adapted from Prof. Yi Ma's ECCV2012 tutorial

Sparse Optimization

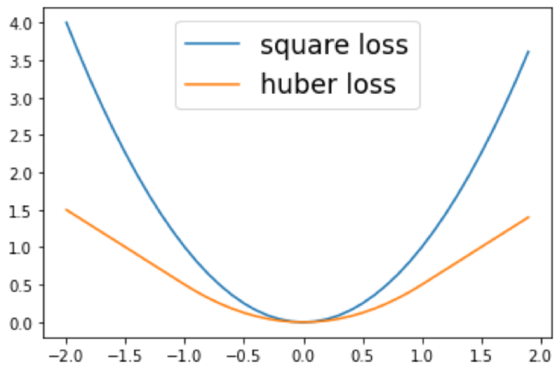
$$\text{minimize} \quad \|\boldsymbol{x}\|_0 \quad \text{subject to} \quad \boldsymbol{A}\boldsymbol{x} = \boldsymbol{y}$$

- $L_1\ norm \quad \|\boldsymbol{x}\|_0 \to \|\boldsymbol{x}\|_1$



- Huber-Loss: $\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 \to L_\delta(\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x})$

  where $L_\delta(x) = \begin{cases} \frac{1}{2}(x)^2 & \text{for } |x| \le \delta \\ \delta \cdot \left(|x| - \frac{1}{2}\delta\right), & \text{otherwise} \end{cases}$

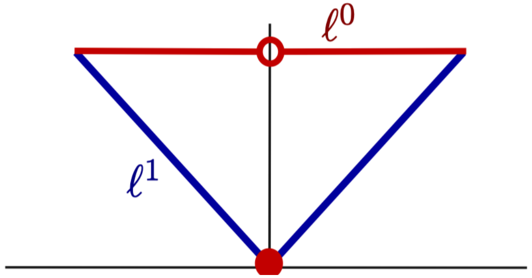


Sparse Optimization

$$\text{minimize} \quad \|\boldsymbol{x}\|_0 \quad \text{subject to} \quad \boldsymbol{A}\boldsymbol{x} = \boldsymbol{y}$$

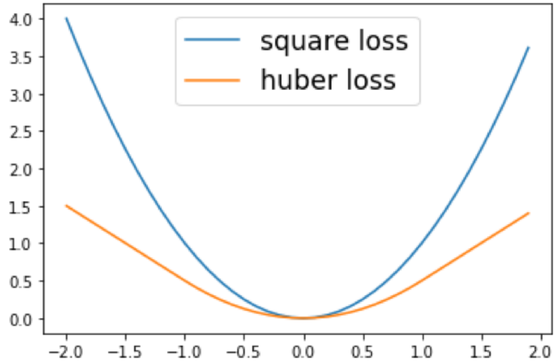
nonconvex ⟶ NP-hard!

- $L_1\ norm \quad \|\boldsymbol{x}\|_0 \rightarrow \|\boldsymbol{x}\|_1$



- Huber-Loss: $\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 \rightarrow L_\delta(\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x})$

  where $L_\delta(x) = \begin{cases} \frac{1}{2}(x)^2 & \text{for } |x| \leq \delta \\ \delta \cdot \left(|x| - \frac{1}{2}\delta\right), & \text{otherwise} \end{cases}$



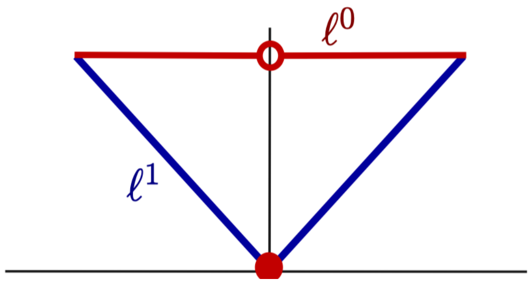
Sparse Optimization

$$\text{minimize} \quad \|\boldsymbol{x}\|_0 \quad \text{subject to} \quad \boldsymbol{A}\boldsymbol{x} = \boldsymbol{y}$$

nonconvex ⟶ NP-hard!

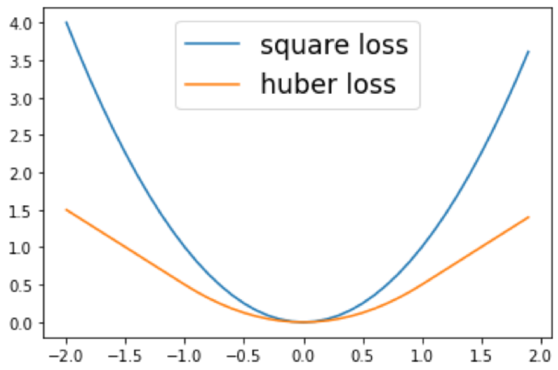
Relax the problem

- $L_1\ norm \quad \|\boldsymbol{x}\|_0 \rightarrow \|\boldsymbol{x}\|_1$



- Huber-Loss: $\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 \rightarrow L_\delta(\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x})$

$$\text{where } L_\delta(x) = \begin{cases} \frac{1}{2}(x)^2 & \text{for } |x| \leq \delta \\ \delta \cdot \left(|x| - \frac{1}{2}\delta\right), & \text{otherwise} \end{cases}$$

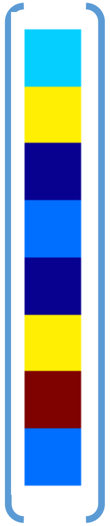


Overview

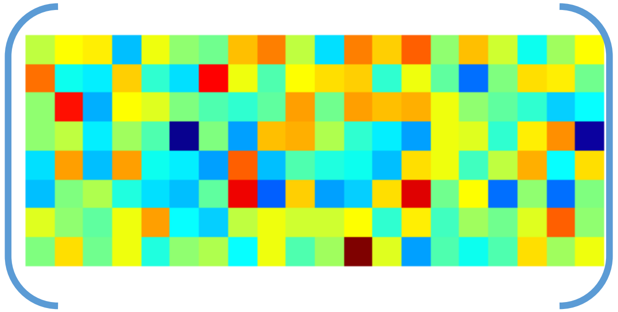
- Sparse Learning in Data/Label
- Sparse Learning in Deep Models

Sparse learning for Noisy Data/Labels

Underdetermined Linear system $y = Ax$



=



$\begin{bmatrix} ? \\ ? \\ ? \\ \cdots \\ ? \end{bmatrix}$

$A \in \mathbb{R}^{m \times n}$

Observation

$y \in \mathbb{R}^m$

Unknown

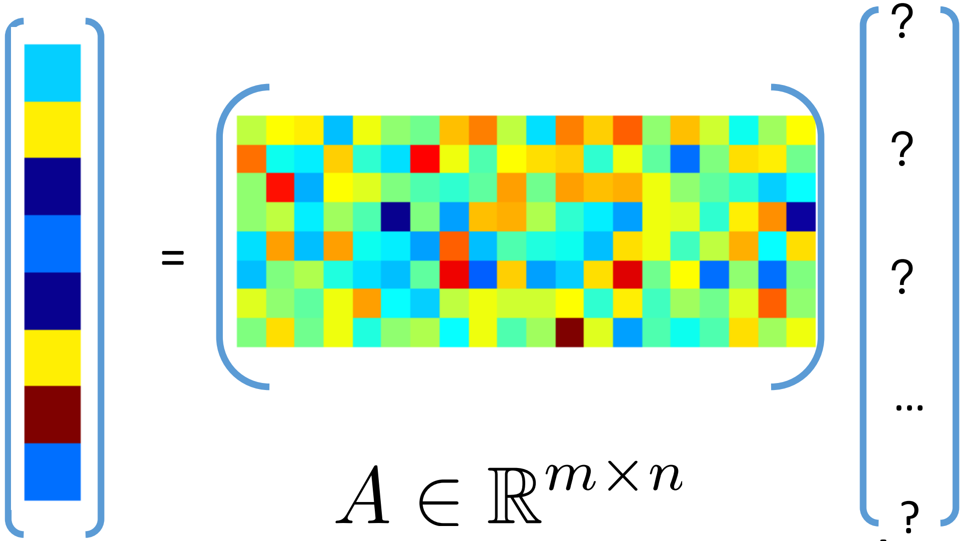
$x \in \mathbb{R}^n$

$m \ll n$

 # Observations **# unknowns**

Sparse learning for Noisy Data/Labels

Underdetermined Linear system $y = Ax$



$A \in \mathbb{R}^{m \times n}$

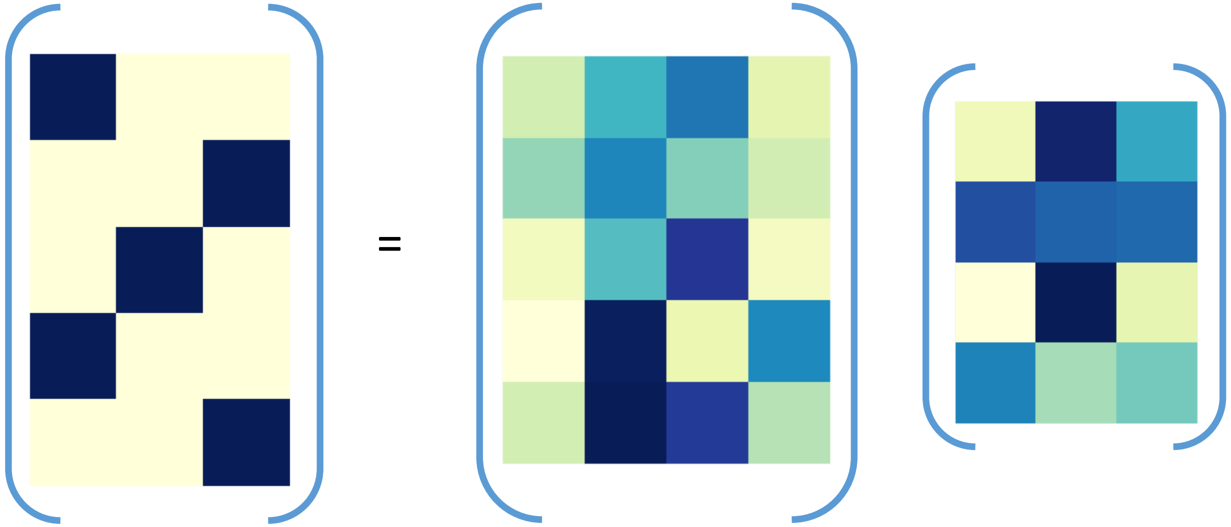
Observation $y \in \mathbb{R}^m$

$m \ll n$

 # Observations **# unknowns**

Unknown $x \in \mathbb{R}^n$

Linear system with Noisy Data/Labels $Y = X\beta$



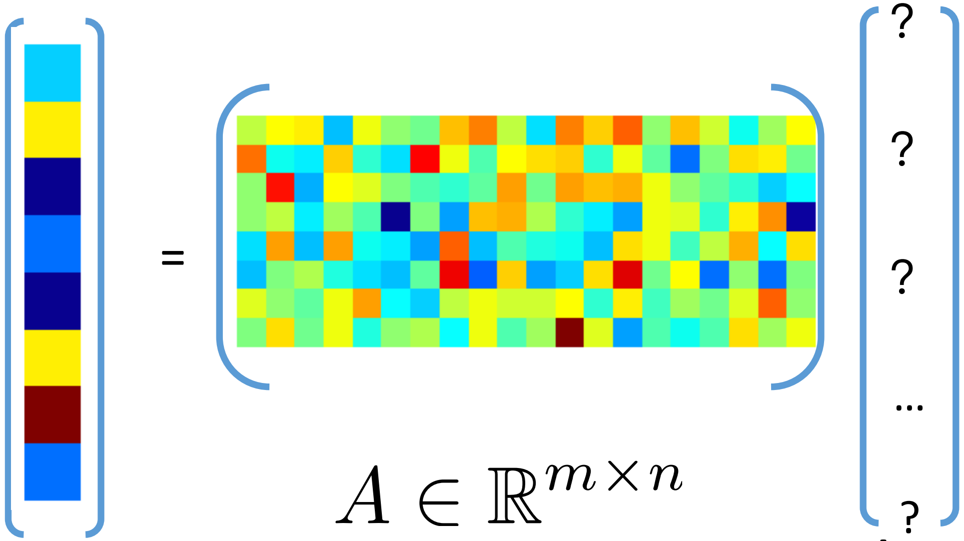
Noisy One-hot Labels $Y \in \mathbb{R}^{n \times c}$

Deep Features $X \in \mathbb{R}^{n \times d}$

Fitted Coef. $\beta \in \mathbb{R}^{d \times c}$

Sparse learning for Noisy Data/Labels

Underdetermined Linear system $y = Ax$



$A \in \mathbb{R}^{m \times n}$

Observation

$y \in \mathbb{R}^m$

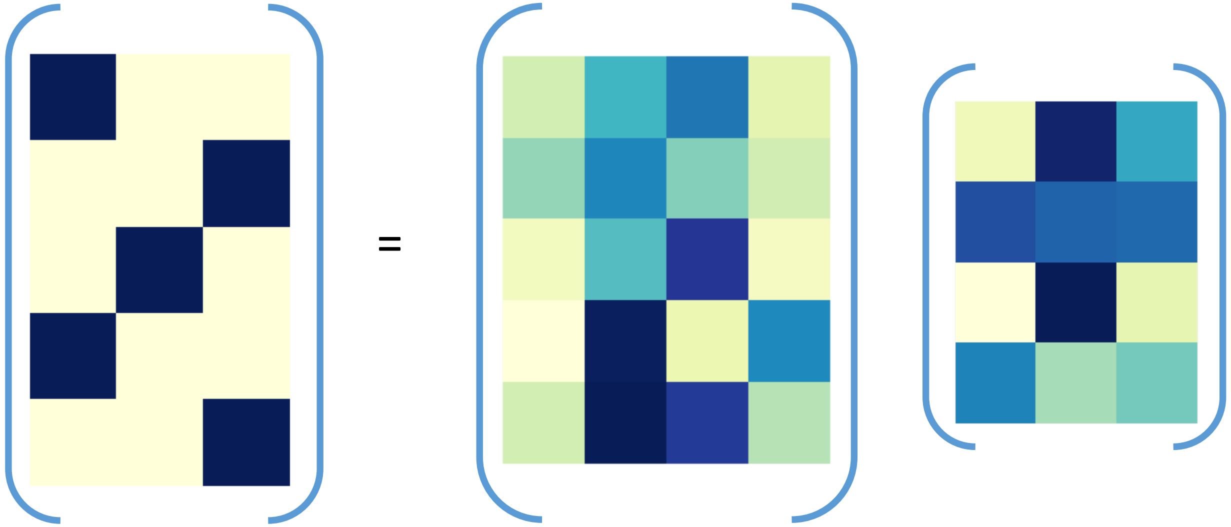
$m \ll n$

 # Observations **# unknowns**

Unknown

$x \in \mathbb{R}^n$

Linear system with Noisy Data/Labels $Y = X\beta$



Noisy One-hot Labels $Y \in \mathbb{R}^{n \times c}$

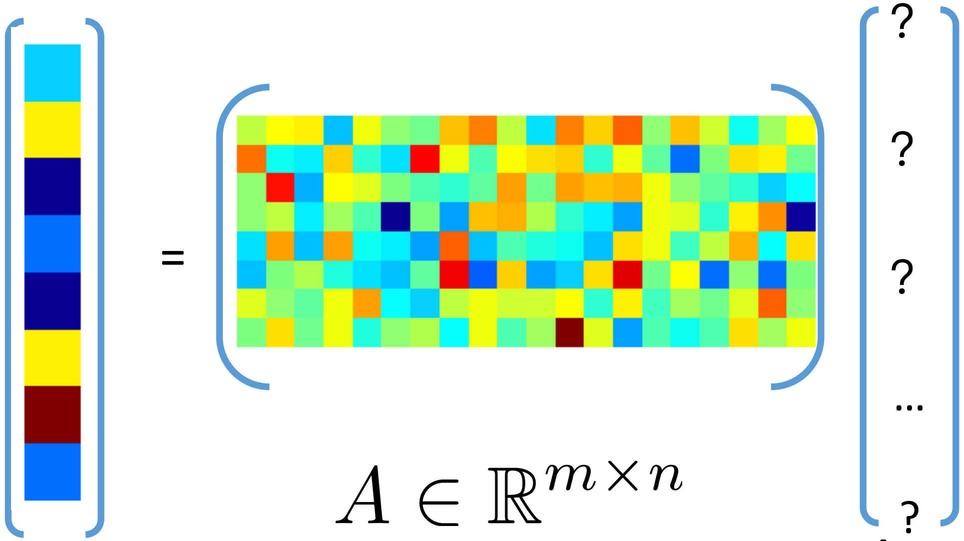
Deep Features $X \in \mathbb{R}^{n \times d}$

Fitted Coef. $\beta \in \mathbb{R}^{d \times c}$

Which Y is noisy?

Sparse learning for Noisy Data/Labels

Underdetermined Linear system $y = Ax$



$A \in \mathbb{R}^{m \times n}$

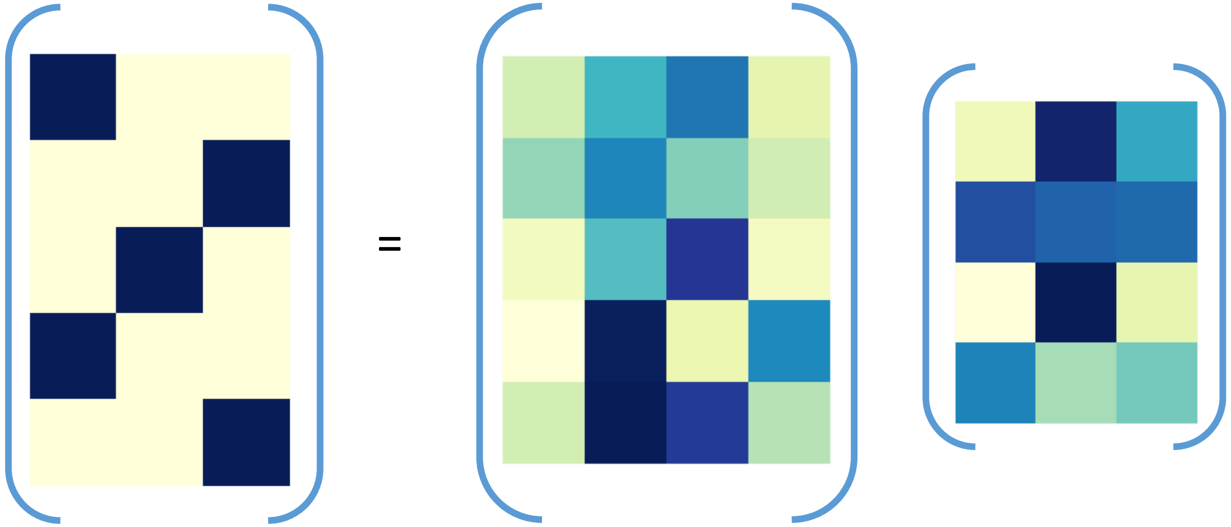
Observation $y \in \mathbb{R}^m$

$m \ll n$

 # Observations **# unknowns**

Unknown $x \in \mathbb{R}^n$

Linear system with Noisy Data/Labels $Y = X\beta$



Noisy One-hot Labels $Y \in \mathbb{R}^{n \times c}$

Deep Features $X \in \mathbb{R}^{n \times d}$

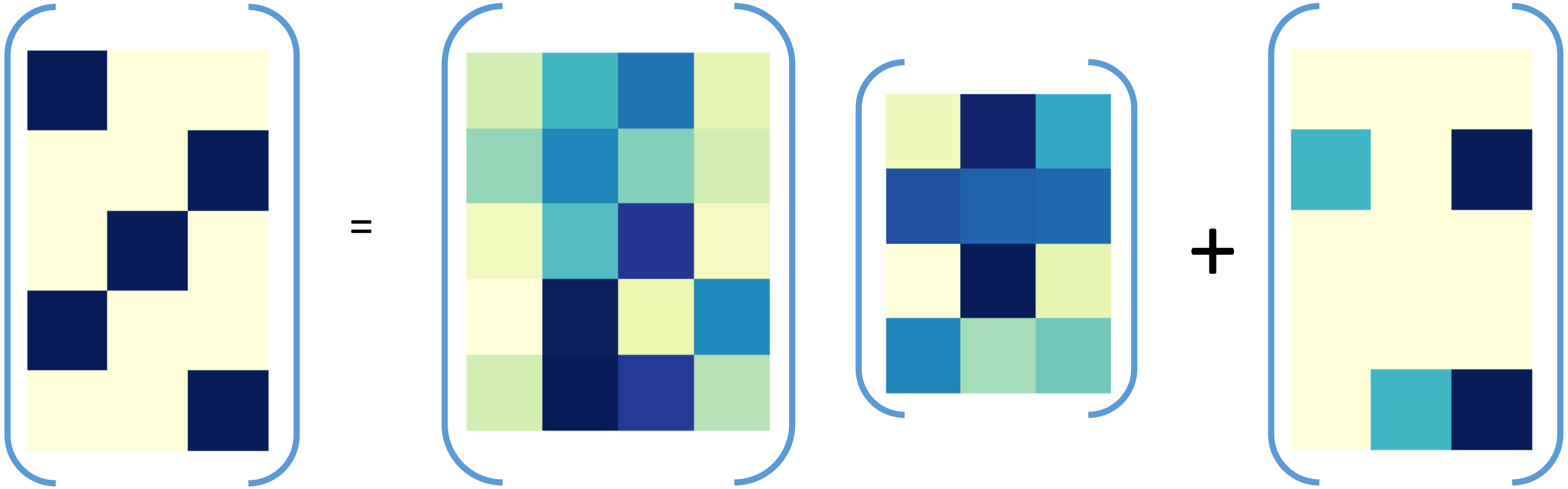
Fitted Coef. $\beta \in \mathbb{R}^{d \times c}$

Which Y is noisy?

We will introduce works of recovering sparse noisy data/labels for Robust Statistical Analysis.

Sparse learning for Noisy Data/Labels: The Indicator

Linear system
with Noisy Data/Labels $Y = X\beta + \gamma$



Noisy One-hot Labels $Y \in \mathbb{R}^{n \times c}$

Deep Features $X \in \mathbb{R}^{n \times d}$

Fitted Coef. $\beta \in \mathbb{R}^{d \times c}$

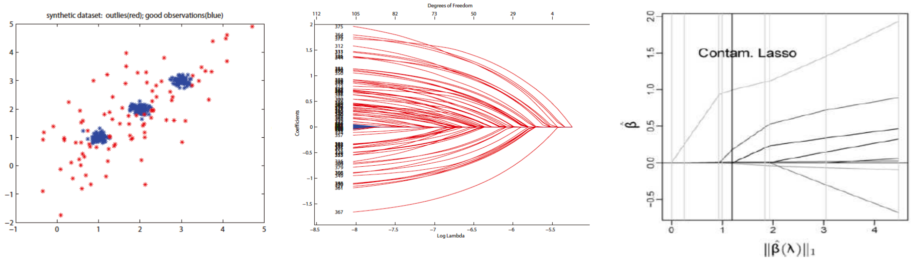
Noisy Data Indicator $\gamma \in \mathbb{R}^{n \times c}$

Part of content adapted from Prof. Yi Ma's ECCV2012 tutorial

Sparsity in Data/Labels: Different Focus

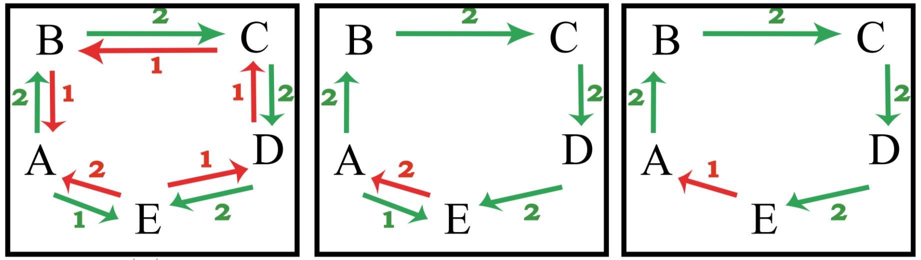
Robust regression/ Classification, [Wang et al. CVPR2020].

$$y = x^\top \beta + \epsilon + \gamma$$



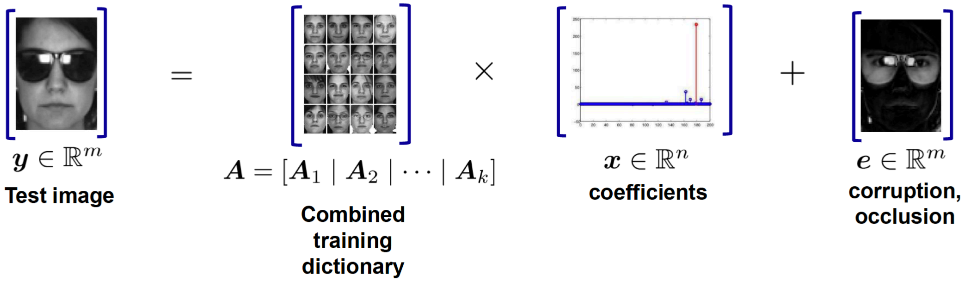
Statistical robust ranking, [Fu et al. TPAMI 16]

$$y = x^\top \beta + \epsilon + \gamma$$



Face Recognition, [Wright et al. TPAMI 09]

$$y = (A, I)\begin{pmatrix} x \\ \gamma \end{pmatrix} + \epsilon$$



[Zhao et al. ICML 2018][Fu et al. ECCV 2014/TPAMI2016], [Wang et al. CVPR2020/TPAMI2021/CVPR2022] [Huang et al. ECCV2014]

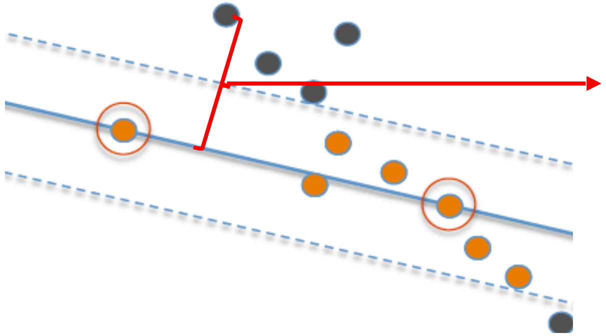
Understanding γ in Statistics

$$y = x^\top \beta + \epsilon + \gamma$$

Yiyuan She and Art B Owen. Outlier detection using nonconvex penalized regression. Journal of the American Statistical Association, 2011.

Understanding γ in Statistics

$$y = x^\top \beta + \epsilon + \gamma$$

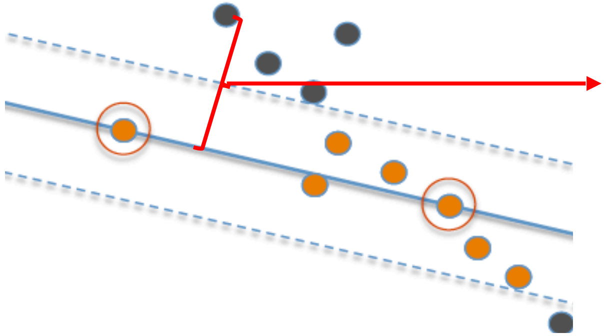


γ_i equals to the residual predict error $\gamma_i = y_i - x_i^\top \hat{\beta}$

Yiyuan She and Art B Owen. Outlier detection using nonconvex penalized regression. Journal of the American Statistical Association, 2011.

Understanding γ in Statistics

$$y = x^\top \beta + \epsilon + \gamma$$



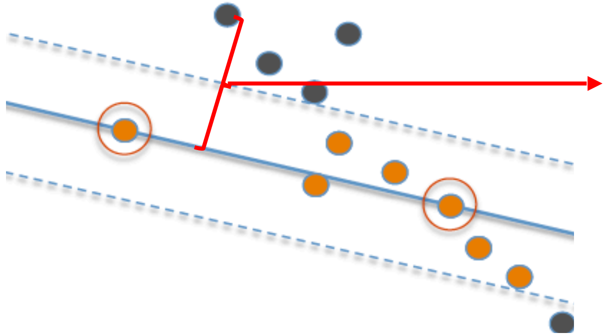
γ_i equals to the residual predict error $\gamma_i = y_i - x_i^\top \hat{\beta}$

Row residuals fail to detect outliers at *leverage points.*

Yiyuan She and Art B Owen. Outlier detection using nonconvex penalized regression. Journal of the American Statistical Association, 2011.

Understanding γ in Statistics

$$y = x^\top \beta + \epsilon + \gamma$$

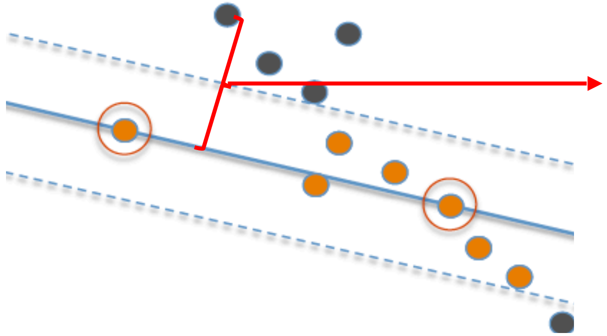


γ_i equals to the residual predict error $\gamma_i = y_i - x_i^\top \hat{\beta}$

Yiyuan She and Art B Owen. Outlier detection using nonconvex penalized regression. Journal of the American Statistical Association, 2011.

Understanding γ in Statistics

$$y = x^\top \beta + \epsilon + \gamma$$



γ_i equals to the residual predict error $\gamma_i = y_i - x_i^\top \hat{\beta}$

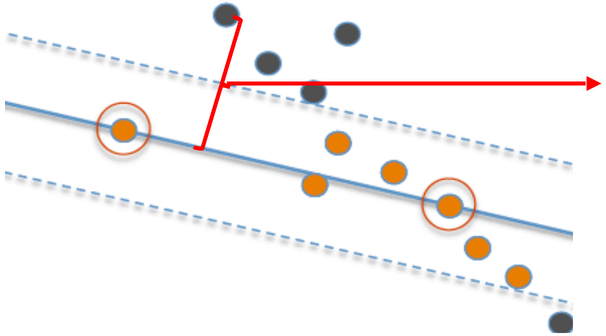
Leave-one-out externally studentized residual

$$t_i = \frac{y_i - \boldsymbol{x}_i^\top \hat{\beta}_{(i)}}{\hat{\sigma}_{(i)}(1+\boldsymbol{x}_i(\boldsymbol{X}_{(i)}^\top \boldsymbol{X}_{(i)})^{-1}\boldsymbol{x}_i)^{1/2}}$$

Yiyuan She and Art B Owen. Outlier detection using nonconvex penalized regression. Journal of the American Statistical Association, 2011.

Understanding γ in Statistics

$$y = x^\top \beta + \epsilon + \gamma$$



γ_i equals to the residual predict error $\gamma_i = y_i - x_i^\top \hat{\beta}$

Leave-one-out externally studentized residual

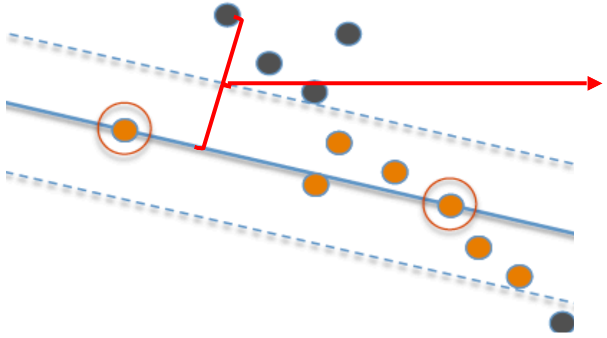
$$t_i = \frac{y_i - \boldsymbol{x}_i^\top \hat{\beta}_{(i)}}{\hat{\sigma}_{(i)}(1+\boldsymbol{x}_i(\boldsymbol{X}_{(i)}^\top \boldsymbol{X}_{(i)})^{-1}\boldsymbol{x}_i)^{1/2}}$$

$\Leftrightarrow$ test whether $\gamma = 0$ in $\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \gamma 1_i + \boldsymbol{\varepsilon}$

Yiyuan She and Art B Owen. Outlier detection using nonconvex penalized regression. Journal of the American Statistical Association, 2011.

Understanding γ in Statistics

$$y = x^{\top}\beta + \epsilon + \gamma$$



γ_i equals to the residual predict error $\gamma_i = y_i - x_i^{\top}\hat{\beta}$

Leave-one-out externally studentized residual

$$t_i = \frac{y_i - \boldsymbol{x}_i^{\top}\hat{\beta}_{(i)}}{\hat{\sigma}_{(i)}(1+\boldsymbol{x}_i(\boldsymbol{X}_{(i)}^{\top}\boldsymbol{X}_{(i)})^{-1}\boldsymbol{x}_i)^{1/2}}$$

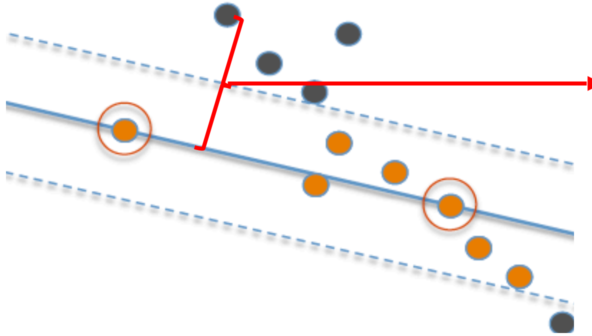
$\Leftrightarrow$ test whether $\gamma = 0$ in $\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \gamma 1_i + \boldsymbol{\varepsilon}$

When there are multiple outliers:

Yiyuan She and Art B Owen. Outlier detection using nonconvex penalized regression. Journal of the American Statistical Association, 2011.

Understanding γ in Statistics

$$y = x^\top \beta + \epsilon + \gamma$$



γ_i equals to the residual predict error $\gamma_i = y_i - x_i^\top \hat{\beta}$

Leave-one-out externally studentized residual

$$t_i = \frac{y_i - \boldsymbol{x}_i^\top \hat{\beta}_{(i)}}{\hat{\sigma}_{(i)}(1+\boldsymbol{x}_i(\boldsymbol{X}_{(i)}^\top \boldsymbol{X}_{(i)})^{-1}\boldsymbol{x}_i)^{1/2}}$$

$\Leftrightarrow$ test whether $\gamma = 0$ in $\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \gamma 1_i + \boldsymbol{\varepsilon}$

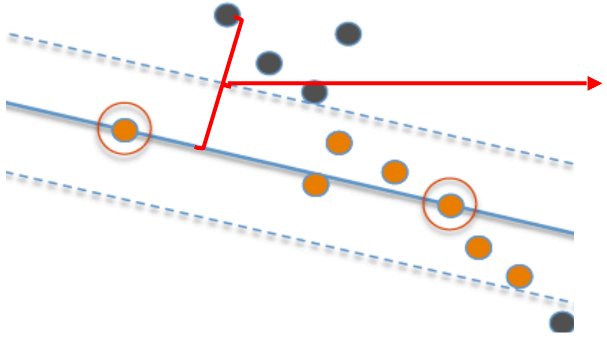
When there are multiple outliers:

1. masking: multiple outliers may mask each other and being undetected;

Yiyuan She and Art B Owen. Outlier detection using nonconvex penalized regression. Journal of the American Statistical Association, 2011.

Understanding γ in Statistics

$$y = x^\top \beta + \epsilon + \gamma$$



γ_i equals to the residual predict error $\gamma_i = y_i - x_i^\top \hat{\beta}$

Leave-one-out externally studentized residual

$$t_i = \frac{y_i - \boldsymbol{x}_i^\top \hat{\beta}_{(i)}}{\hat{\sigma}_{(i)}(1+\boldsymbol{x}_i(\boldsymbol{X}_{(i)}^\top \boldsymbol{X}_{(i)})^{-1}\boldsymbol{x}_i)^{1/2}}$$

$\Leftrightarrow$ test whether $\gamma = 0$ in $\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \gamma 1_i + \boldsymbol{\varepsilon}$

When there are multiple outliers:

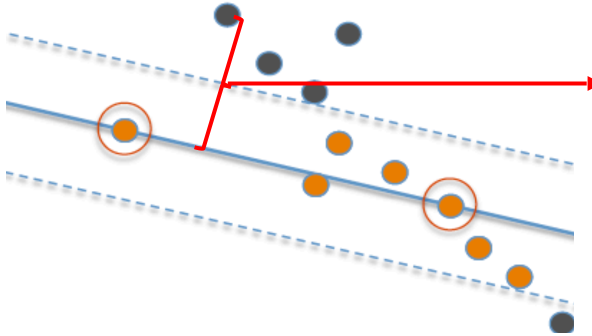
1. masking: multiple outliers may mask each other and being undetected;

2. swamping: multiple outliers may lead the large t_i for clean data.

Yiyuan She and Art B Owen. Outlier detection using nonconvex penalized regression. Journal of the American Statistical Association, 2011.

Understanding γ in Statistics

$$y = x^\top \beta + \epsilon + \gamma$$



γ_i equals to the residual predict error $\gamma_i = y_i - x_i^\top \hat{\beta}$

$\downarrow$

Leave-one-out externally studentized residual

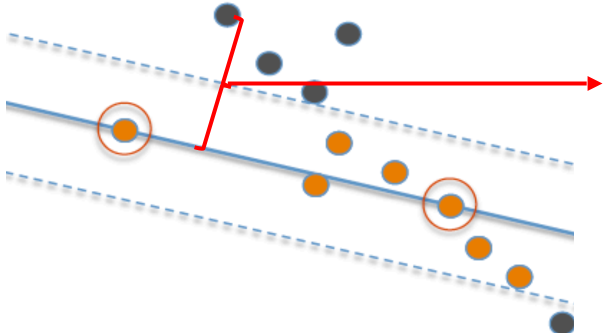
$$t_i = \frac{y_i - \boldsymbol{x}_i^\top \hat{\beta}_{(i)}}{\hat{\sigma}_{(i)} (1 + \boldsymbol{x}_i (\boldsymbol{X}_{(i)}^\top \boldsymbol{X}_{(i)})^{-1} \boldsymbol{x}_i)^{1/2}}$$

$\Leftrightarrow$ test whether $\gamma = 0$ in $\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \gamma 1_i + \boldsymbol{\varepsilon}$

Yiyuan She and Art B Owen. Outlier detection using nonconvex penalized regression. Journal of the American Statistical Association, 2011.

Understanding γ in Statistics

$$y = x^\top \beta + \epsilon + \gamma$$



γ_i equals to the residual predict error $\gamma_i = y_i - x_i^\top \hat{\beta}$

$\downarrow$

Leave-one-out externally studentized residual

$$t_i = \frac{y_i - \boldsymbol{x}_i^\top \hat{\beta}_{(i)}}{\hat{\sigma}_{(i)}(1+\boldsymbol{x}_i(\boldsymbol{X}_{(i)}^\top \boldsymbol{X}_{(i)})^{-1}\boldsymbol{x}_i)^{1/2}}$$

$\Leftrightarrow$ test whether $\gamma = 0$ in $\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \gamma 1_i + \boldsymbol{\varepsilon}$

$\downarrow$

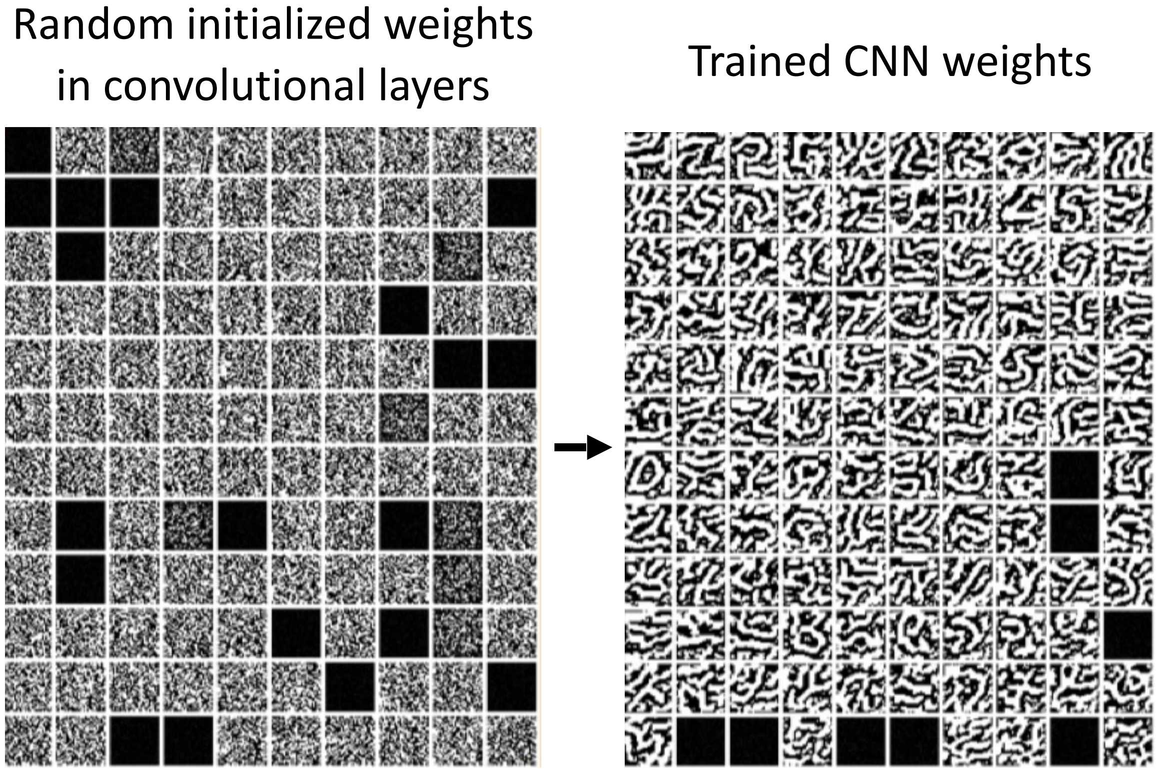
$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\epsilon} + \boldsymbol{\gamma}$$

Yiyuan She and Art B Owen. Outlier detection using nonconvex penalized regression. Journal of the American Statistical Association, 2011.

Overview

- Spare Learning in Data/Label
- Sparse Learning in Deep Models

Learning Sparsity in Neural Networks

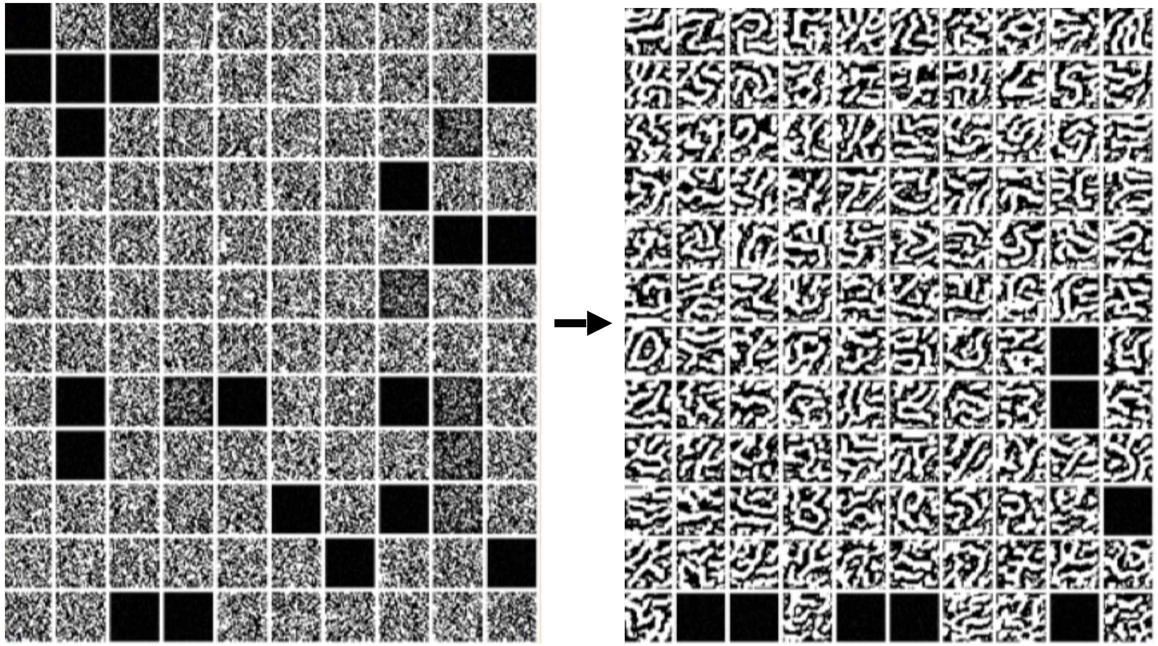


Densely trained model

[Fu et al. ICML 2020/TPAMI2022]

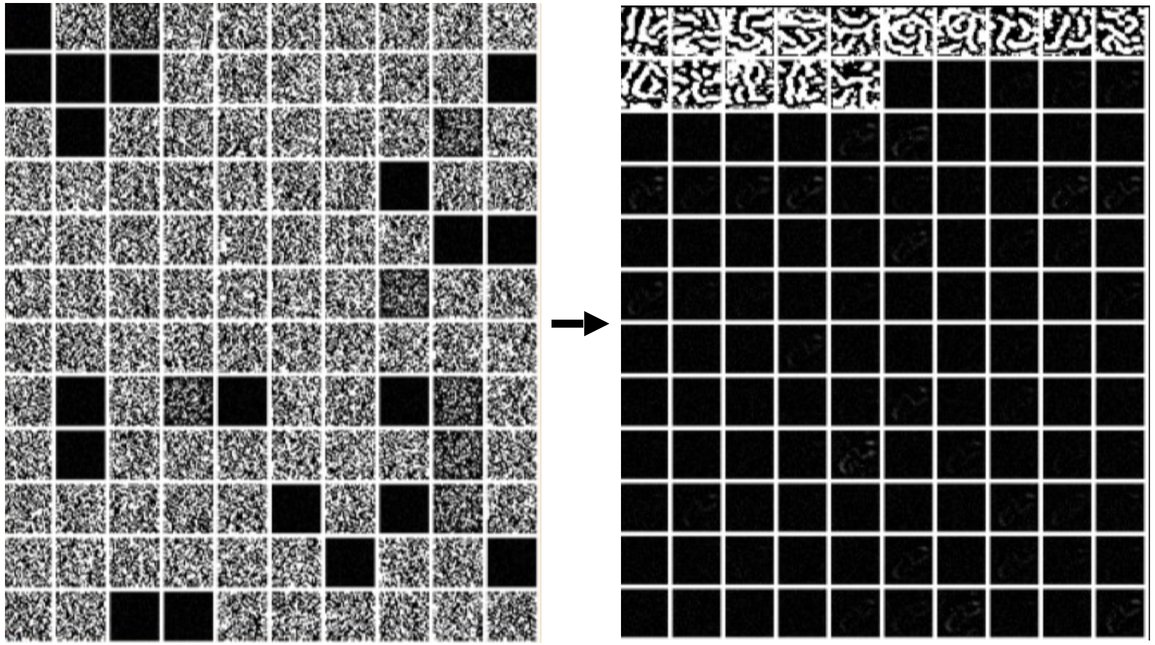
Learning Sparsity in Neural Networks

Random initialized weights in convolutional layers → Trained CNN weights



Densely trained model

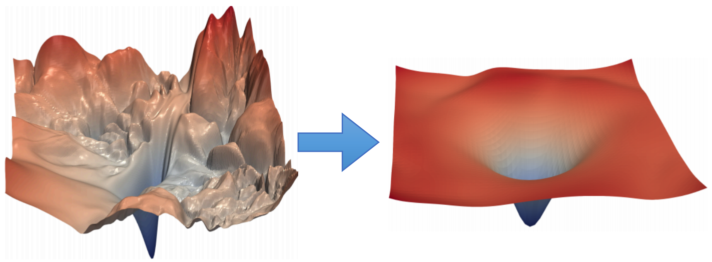
Random initialized weights in convolutional layers → Trained CNN weights



Sparsified model

[Fu et al. ICML 2020/TPAMI2022]

Tradeoff between Overparameterized and Compressive models



Over-parameterized Models

Compressive/Sparse Models

Pros
- Great Expressive Power
- Simplify Loss Landscape

Cons
- Too much parameters
- Even hard to inference on single machine

Pros
- Less memory & Running cost
- Easy to deploy on limited resource machines

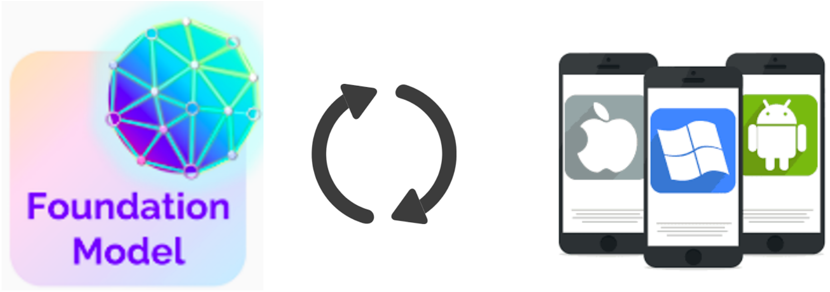
Cons
- Might loss of accuracy

[Fu et al. ICML 2020/TPAMI2022]

Left figures from Li et al. Visualizing the loss landscape of neural nets. NeurPIS 2018

Potential Connection to Foundation model

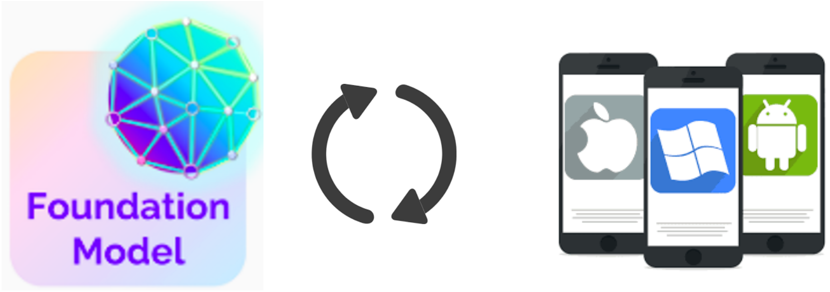
Potential Connection to Foundation model



Training foundation model:

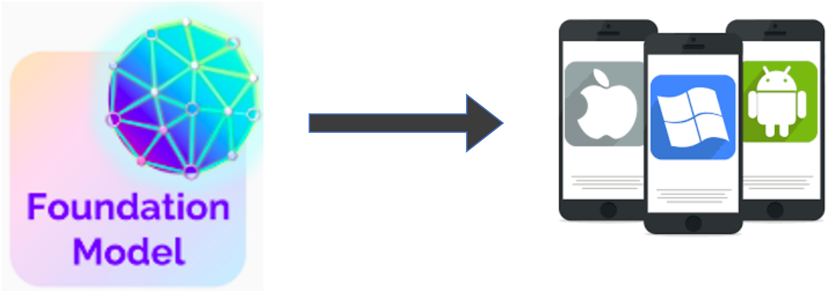
- A routine of compressing and growing network may be beneficial.

Potential Connection to Foundation model



Training foundation model:

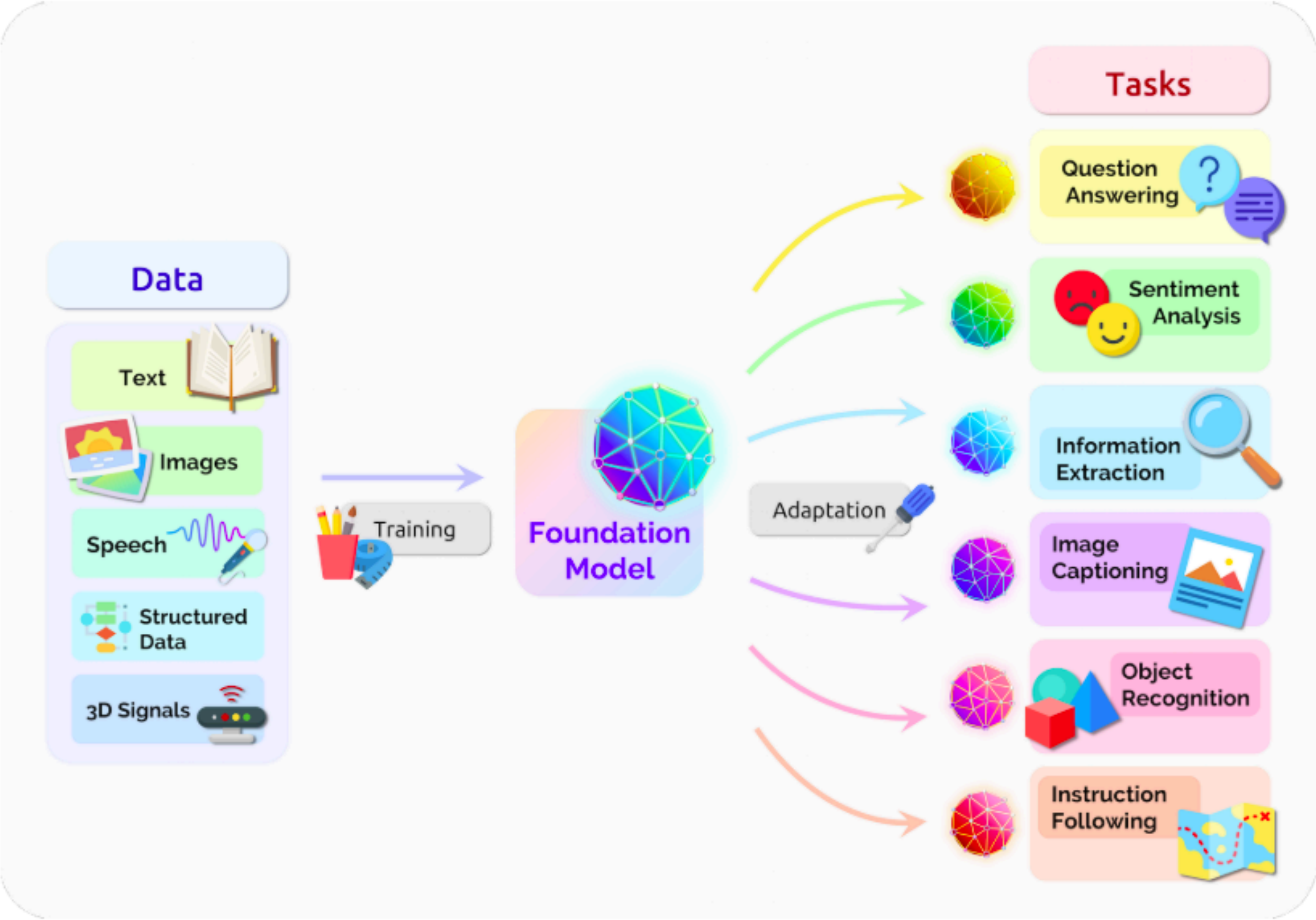
- A routine of compressing and growing network may be beneficial.



Deploying to downstream task:

- Desirable reduced model size for the task of limited resources

The Dilemma of Foundation Models

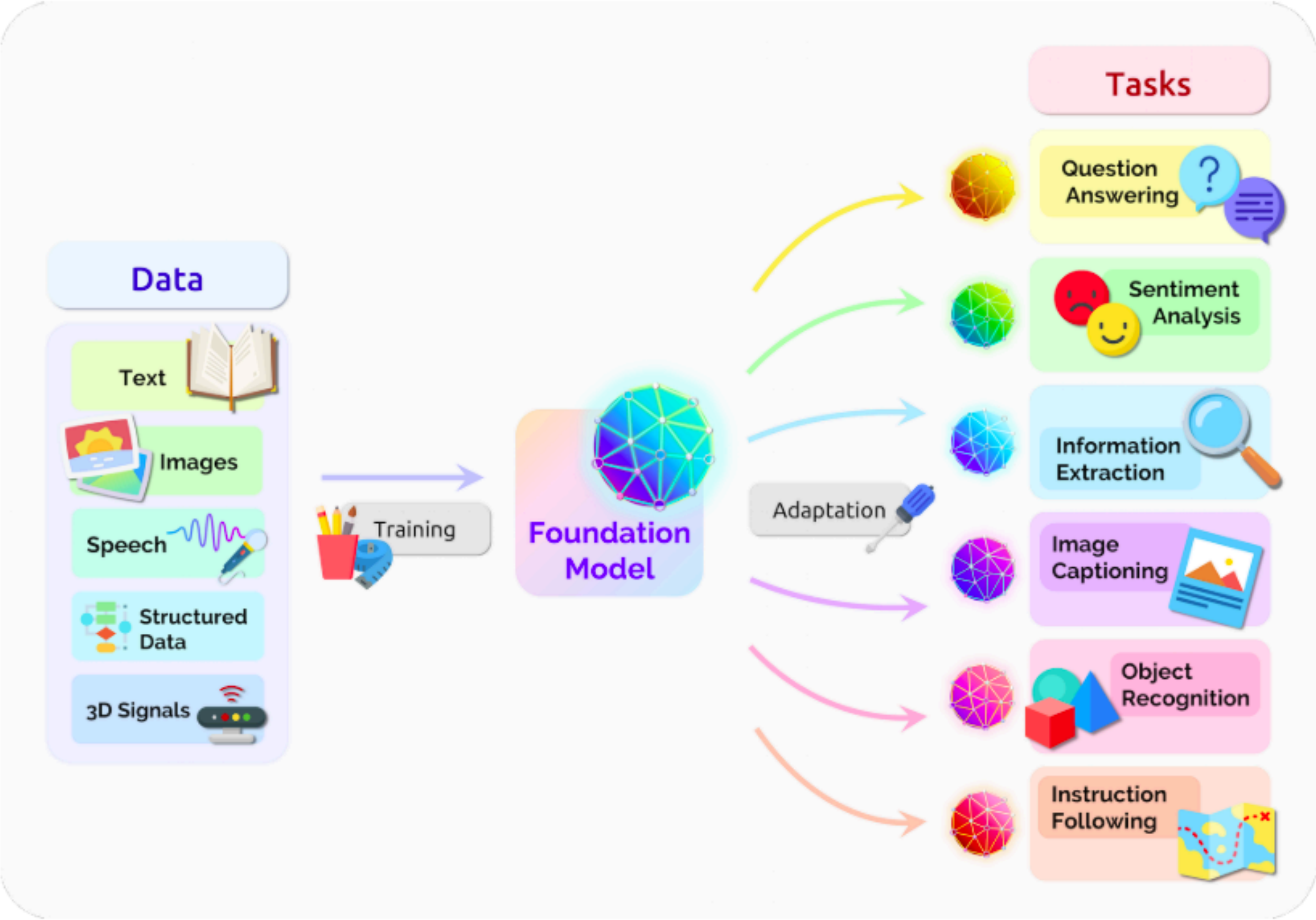


The foundation model is *emergence* and *homogenization, and* should be adapted to different task deployed on various platforms. Figure modified from [Bommasani et al 2021].

Bommasani et al. On the Opportunities and Risks of Foundation Models. arx:108.07258, 2021

The Dilemma of Foundation Models

4B

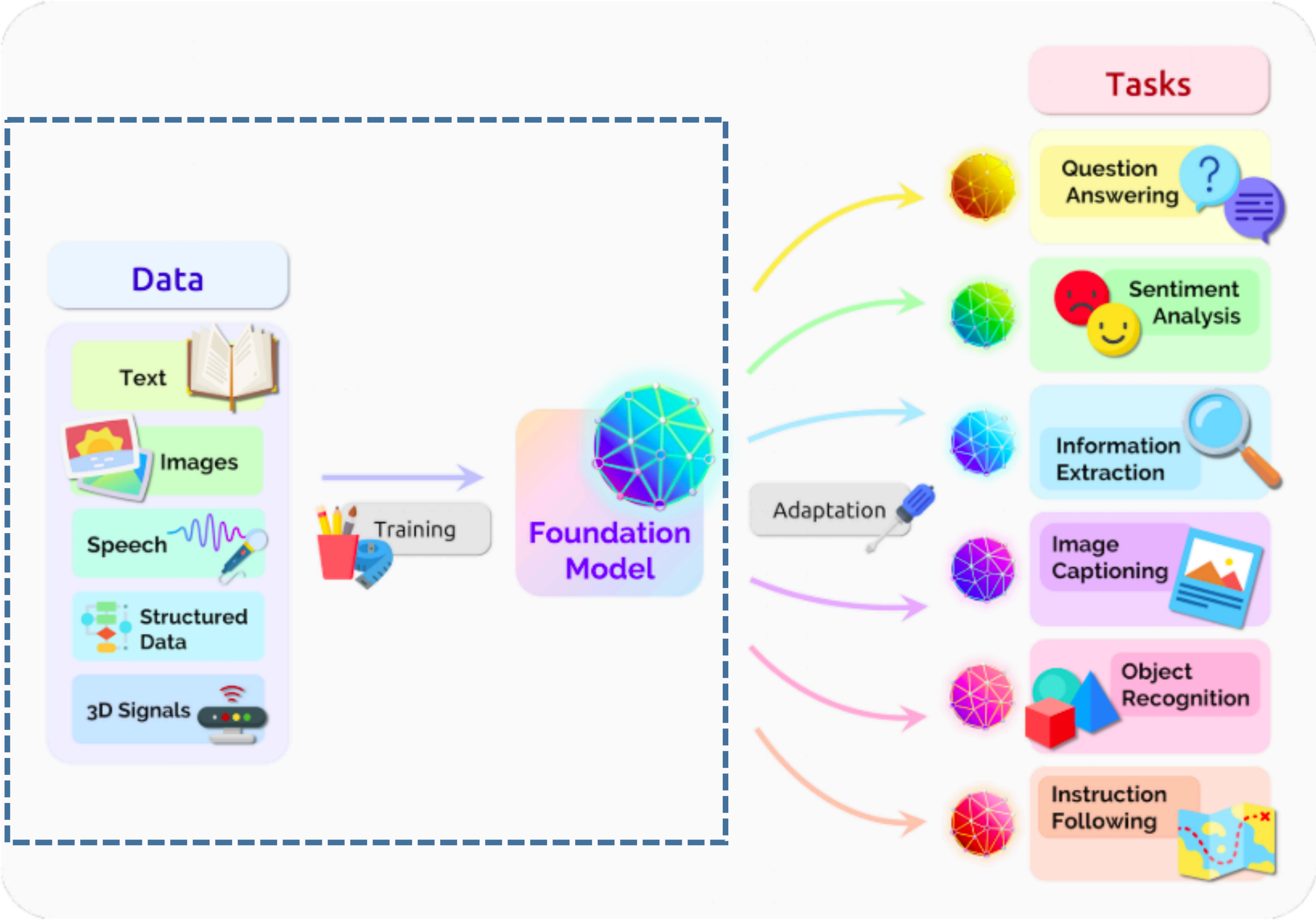


The foundation model is *emergence* and *homogenization, and* should be adapted to different task deployed on various platforms. Figure modified from [Bommasani et al 2021].

Bommasani et al. On the Opportunities and Risks of Foundation Models. arx:108.07258, 2021

The Dilemma of Foundation Models

4B



The foundation model is *emergence* and *homogenization, and* should be adapted to different task deployed on various platforms. Figure modified from [Bommasani et al 2021].

Bommasani et al. On the Opportunities and Risks of Foundation Models. arx:108.07258, 2021

The Dilemma of Foundation Models

4B

Big Data

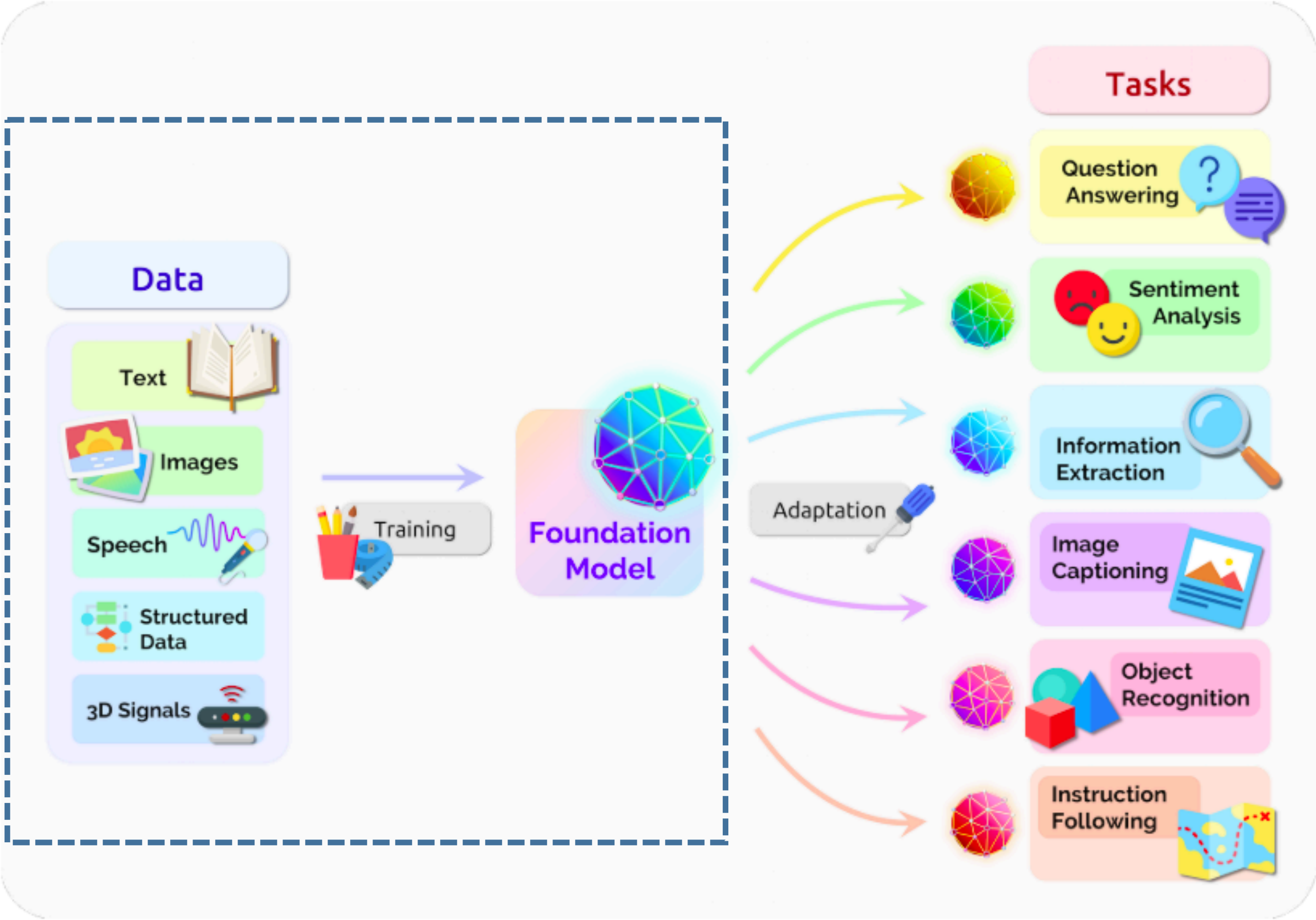
+

Big Machines

+

Big Model

||



The foundation model is *emergence* and *homogenization, and* should be adapted to different task deployed on various platforms. Figure modified from [Bommasani et al 2021].

Bommasani et al. On the Opportunities and Risks of Foundation Models. arx:108.07258, 2021

The Dilemma of Foundation Models

4B

Big Data

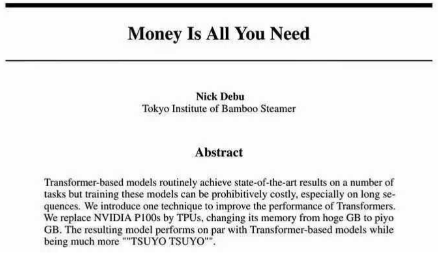
+

Big Machines

+

Big Model

=



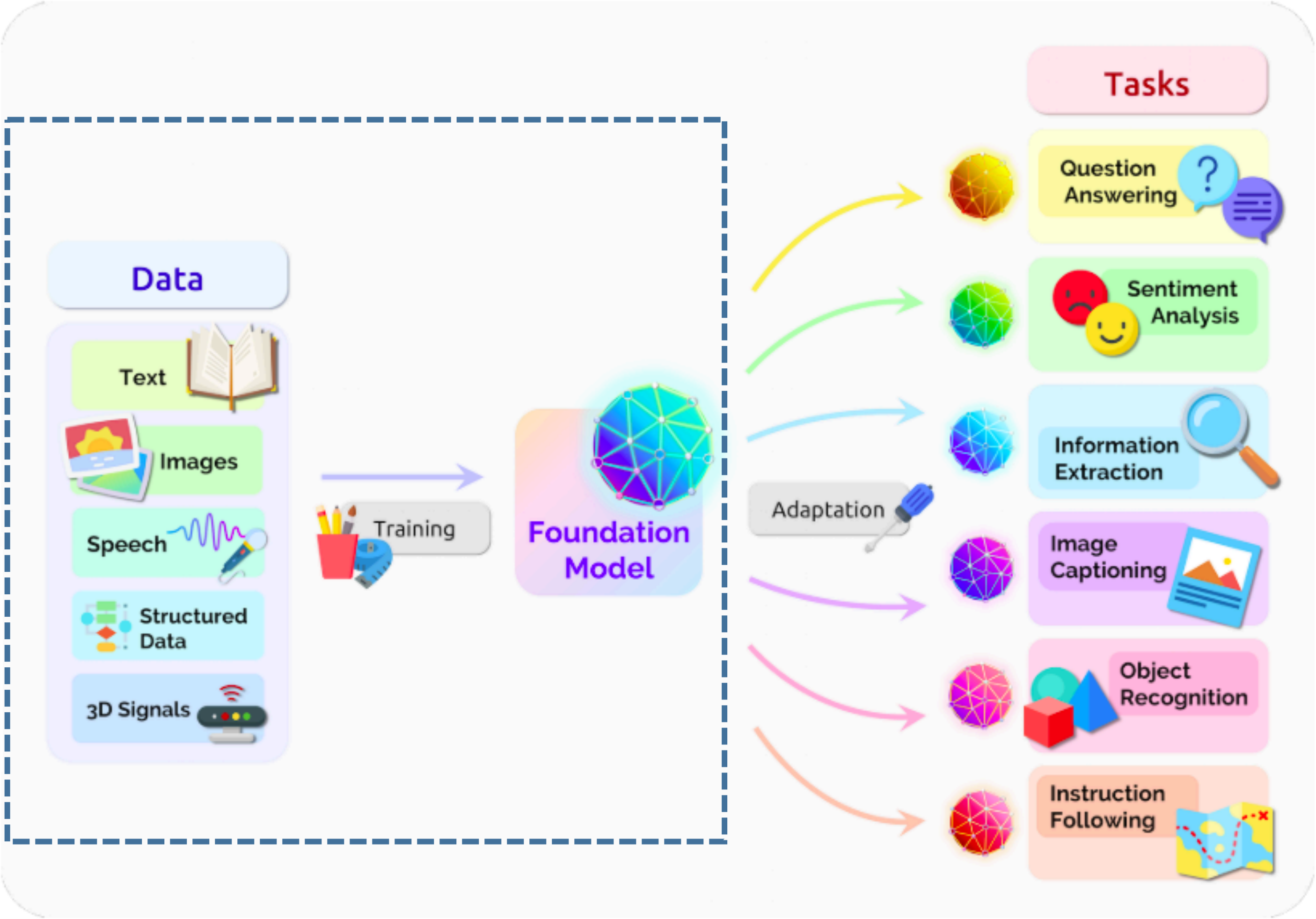
Money Is All You Need

Nick Debu
Tokyo Institute of Bamboo Steamer

Abstract

Transformer-based models routinely achieve state-of-the-art results on a number of tasks but training these models can be prohibitively costly, especially on long sequences. We introduce one technique to improve the performance of Transformers. We replace NVIDIA P100s by TPUs, changing its memory from hoge GB to piyo GB. The resulting model performs on par with Transformer-based models while being much more ""TSUYO TSUYO"".

Big Money



The foundation model is *emergence* and *homogenization, and* should be adapted to different task deployed on various platforms. Figure modified from [Bommasani et al 2021].

Bommasani et al. On the Opportunities and Risks of Foundation Models. arx:108.07258, 2021

The Dilemma of Foundation Models

4B

Big Data

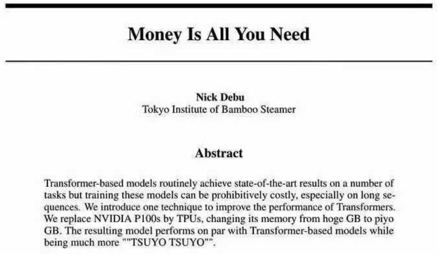
+

Big Machines

+

Big Model

||



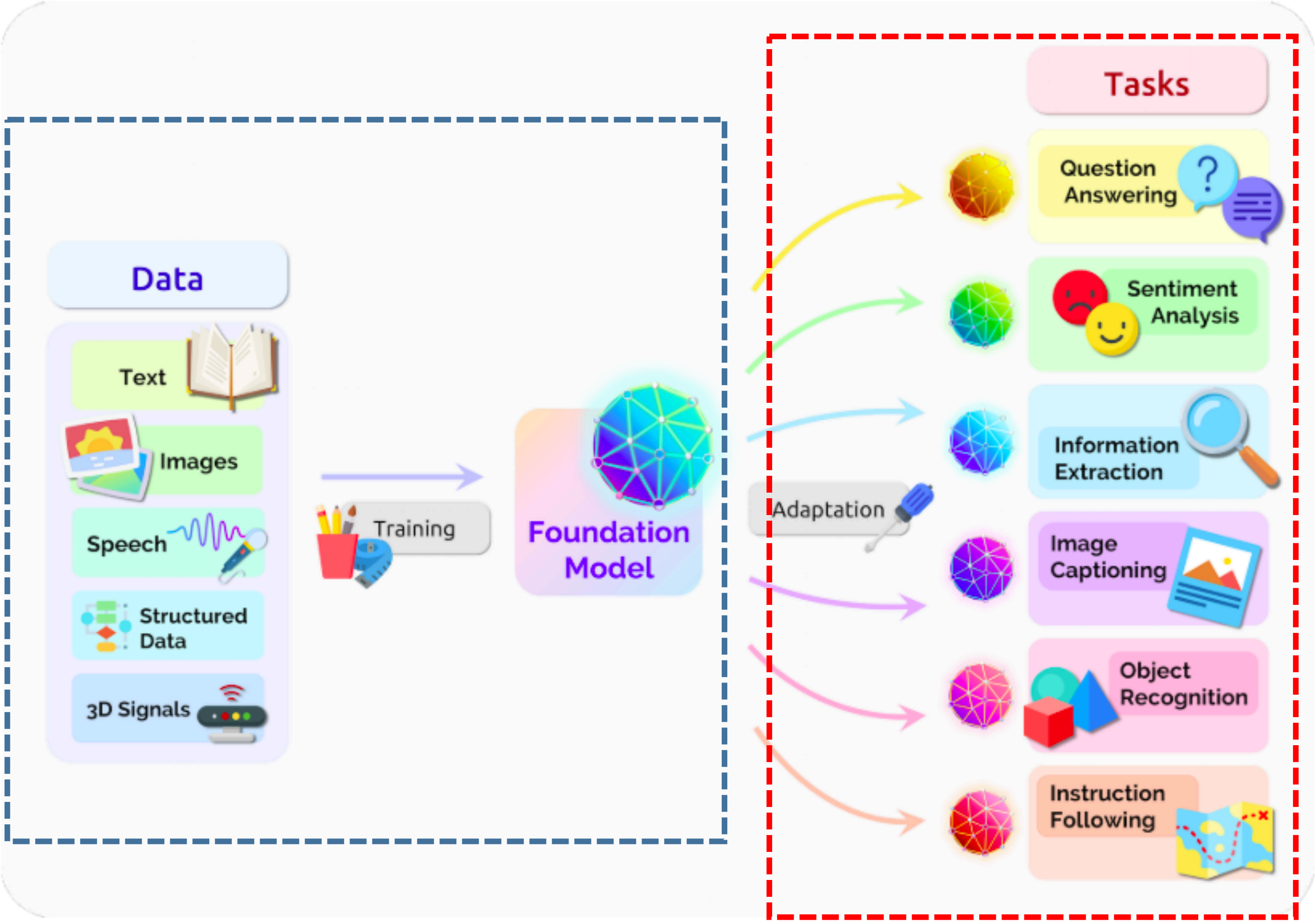
Money Is All You Need

Nick Debu
Tokyo Institute of Bamboo Steamer

Abstract

Transformer-based models routinely achieve state-of-the-art results on a number of tasks but training these models can be prohibitively costly, especially on long sequences. We introduce one technique to improve the performance of Transformers. We replace NVIDIA P100s by TPUs, changing its memory from hoge GB to piyo GB. The resulting model performs on par with Transformer-based models while being much more ""TSUYO TSUYO"".

Big Money



The foundation model is *emergence* and *homogenization, and* should be adapted to different task deployed on various platforms. Figure modified from [Bommasani et al 2021].

Bommasani et al. On the Opportunities and Risks of Foundation Models. arx:108.07258, 2021

大数据学院 School of Data Science

The Dilemma of Foundation Models

4B

Big Data

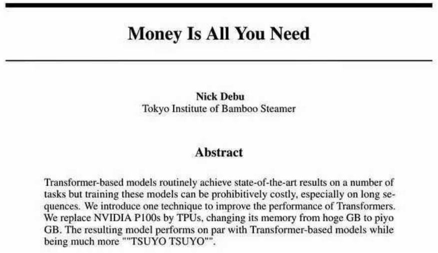
+

Big Machines

+

Big Model

||

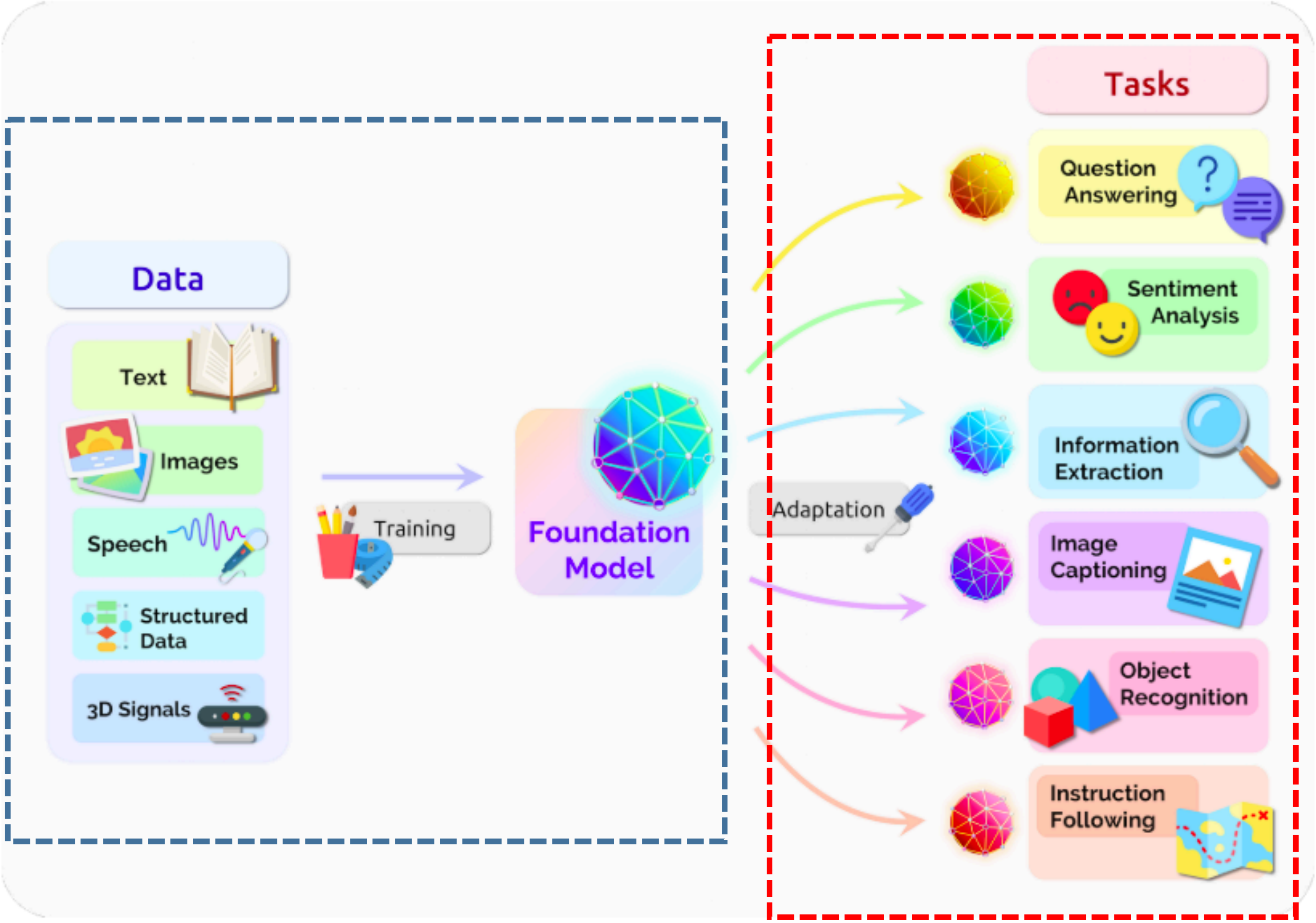

Money Is All You Need

Nick Debu
Tokyo Institute of Bamboo Steamer

Abstract

Transformer-based models routinely achieve state-of-the-art results on a number of tasks but training these models can be prohibitively costly, especially on long sequences. We introduce one technique to improve the performance of Transformers. We replace NVIDIA P100s by TPUs, changing its memory from hoge GB to piyo GB. The resulting model performs on par with Transformer-based models while being much more ""TSUYO TSUYO"".

Big Money



The foundation model is *emergence* and *homogenization, and* should be adapted to different task deployed on various platforms. Figure modified from [Bommasani et al 2021].

Bommasani et al. On the Opportunities and Risks of Foundation Models. arx:108.07258, 2021

Foundation Models in Computer Vision

- **CLIP**: Learning Transferable Visual Models From Natural Language Supervision, arXiv Feb. 24, ICML2021, OpenAI
  - *Code/model: https://github.com/openai/CLIP*
- **ALIGN** : Scaling Up Visual and Vision-Language Representation Learning With Noisy Text Supervision, ICML2021, Google Research
  - *Code/model: N/A*
- ALBEF : Align before Fuse: Vision and Language Representation Learning with Momentum Distillation, NeurIPS 2021, *Salesforce Research*
  - *Code/model: https://github.com/salesforce/ALBEF ,*
- **Florence**: A New Foundation Model for Computer Vision, arXiv, Nov. 22, 2021, Microsoft Cloud and AI, Microsoft Research Redmond
  - *Code/model: N/A*
- **NUWA**: Visual Synthesis Pre-training for Neural visUal World creAtion, arXiv Nov. 24, 2021, MSRA, Peking Unv.
  - *Code/model: https://github.com/microsoft/NUWA*
- **INTERN**: A New Learning Paradigm Towards General Vision, arXiv Nov. 16, 2021 *Shanghai AI Laboratory, SenseTime, CUKH, SJTU*
  - *Code/model: N/A*
- **Gopher**: Scaling Language Models: Methods, Analysis & Insights from Training Gopher, arXiv Dec. 8, 2021, *DeepMind*
  - *Code/model: N/A*
- **FLAVA:** A Foundational Language And Vision Alignment Model, CVPR 2022, FAIR
  - *Code/model: https://flava-model.github.io*
- **OPT**: Open Pre-trained Transformer Language Models. Meta AI 2022
  - *Code/model: https://github.com/facebookresearch/metaseq/tree/main/projects/OPT*

Foundation Models in Computer Vision

- **CLIP**: Learning Transferable Visual Models From Natural Language Supervision, arXiv Feb. 24, ICML2021, OpenAI
  - *Code/model: https://github.com/openai/CLIP*
- **ALIGN** : Scaling Up Visual and Vision-Language Representation Learning With Noisy Text Supervision, ICML2021, Google Research
  - *Code/model: N/A*
- ALBEF : Align before Fuse: Vision and Language Representation Learning with Momentum Distillation, NeurIPS 2021, *Salesforce Research*
  - *Code/model: https://github.com/salesforce/ALBEF ,*
- **Florence**: A New Foundation Model for Computer Vision, arXiv, Nov. 22, 2021, Microsoft Cloud and AI, Microsoft Research Redmond
  - *Code/model: N/A*
- **NUWA**: Visual Synthesis Pre-training for Neural visUal World creAtion, arXiv Nov. 24, 2021, MSRA, Peking Unv.
  - *Code/model: https://github.com/microsoft/NUWA*
- **INTERN**: A New Learning Paradigm Towards General Vision, arXiv Nov. 16, 2021 *Shanghai AI Laboratory, SenseTime, CUKH, SJTU*
  - *Code/model: N/A*
- **Gopher**: Scaling Language Models: Methods, Analysis & Insights from Training Gopher, arXiv Dec. 8, 2021, *DeepMind*
  - *Code/model: N/A*
- **FLAVA**: A Foundational Language And Vision Alignment Model, CVPR 2022, FAIR
  - *Code/model: https://flava-model.github.io*
- **OPT**: Open Pre-trained Transformer Language Models. Meta AI 2022
  - *Code/model: https://github.com/facebookresearch/metaseq/tree/main/projects/OPT*

It urges us to study *learning sparsity in deep foundation models*

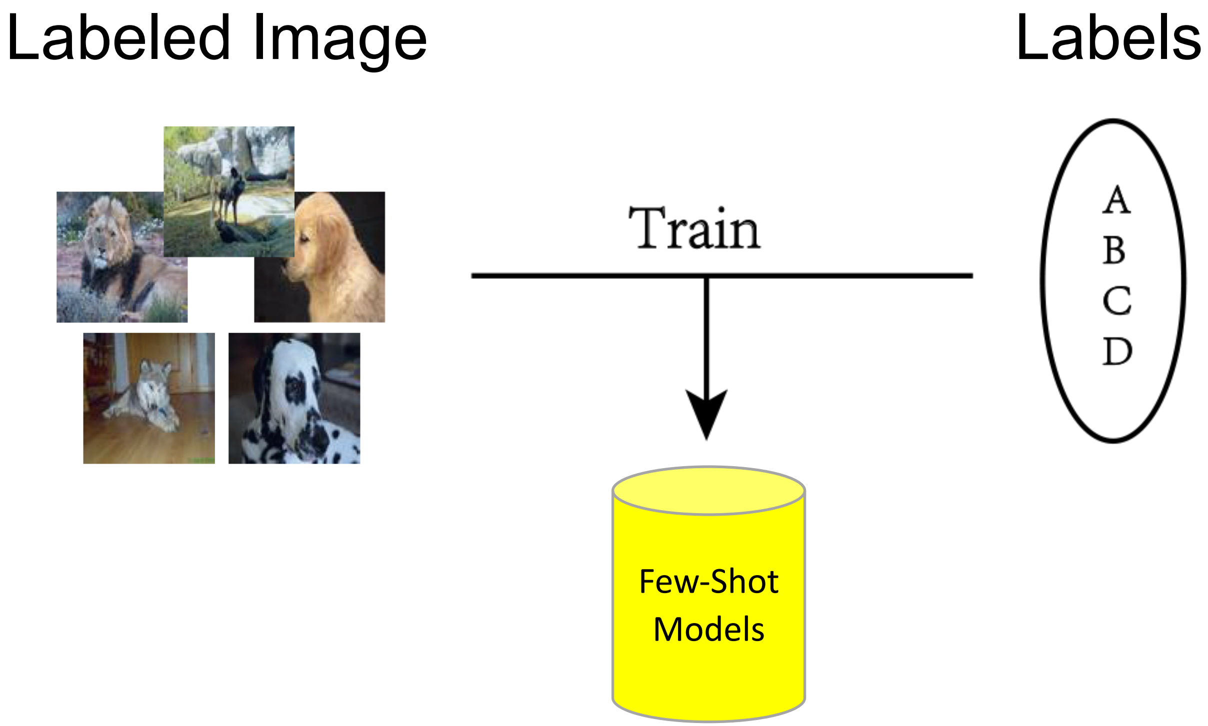
Learning Sparsity in Data/Labels

Few-shot Learning by Unlabeled Data

Wang et al. Instance Credibility Inference for Few-Shot Learning. CVPR 2020

Wang et al. How to Trust Unlabeled Data? Instance Credibility Inference for Few-Shot Learning. IEEE TPAMI 2021

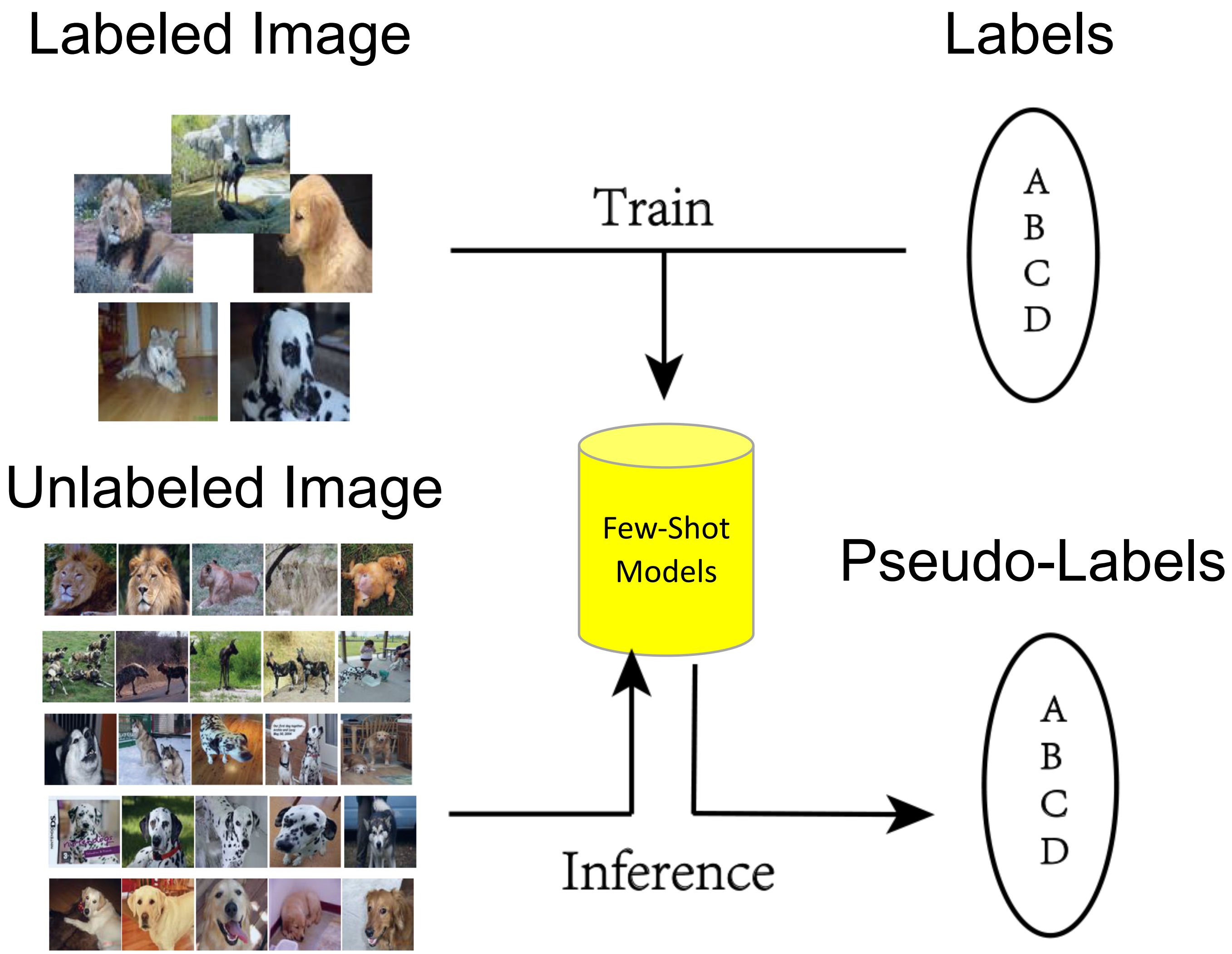
Few-shot Learning by Unlabeled Data



Wang et al. Instance Credibility Inference for Few-Shot Learning. CVPR 2020

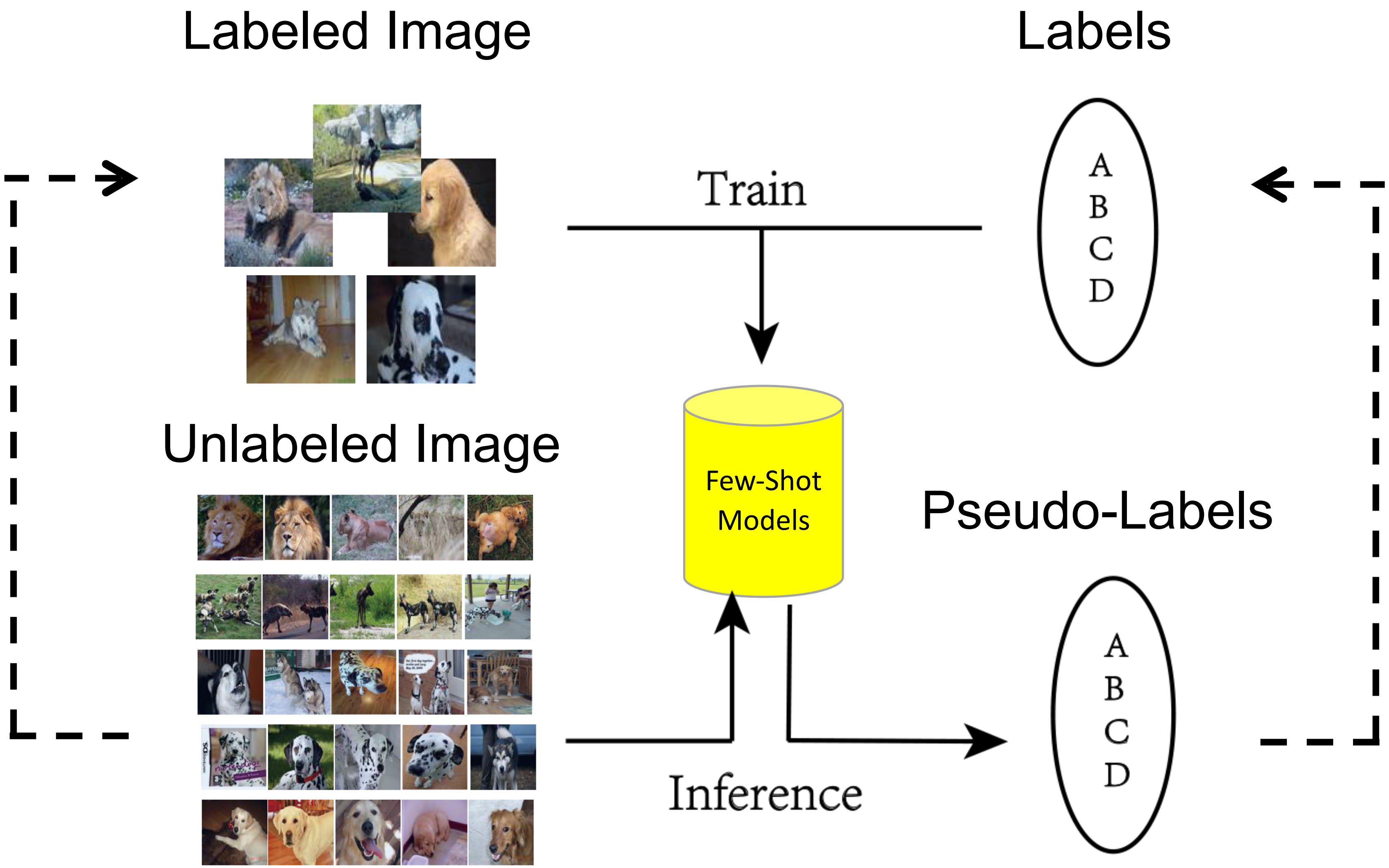
Wang et al. How to Trust Unlabeled Data? Instance Credibility Inference for Few-Shot Learning. IEEE TPAMI 2021

Few-shot Learning by Unlabeled Data



Wang et al. Instance Credibility Inference for Few-Shot Learning. CVPR 2020
Wang et al. How to Trust Unlabeled Data? Instance Credibility Inference for Few-Shot Learning. IEEE TPAMI 2021

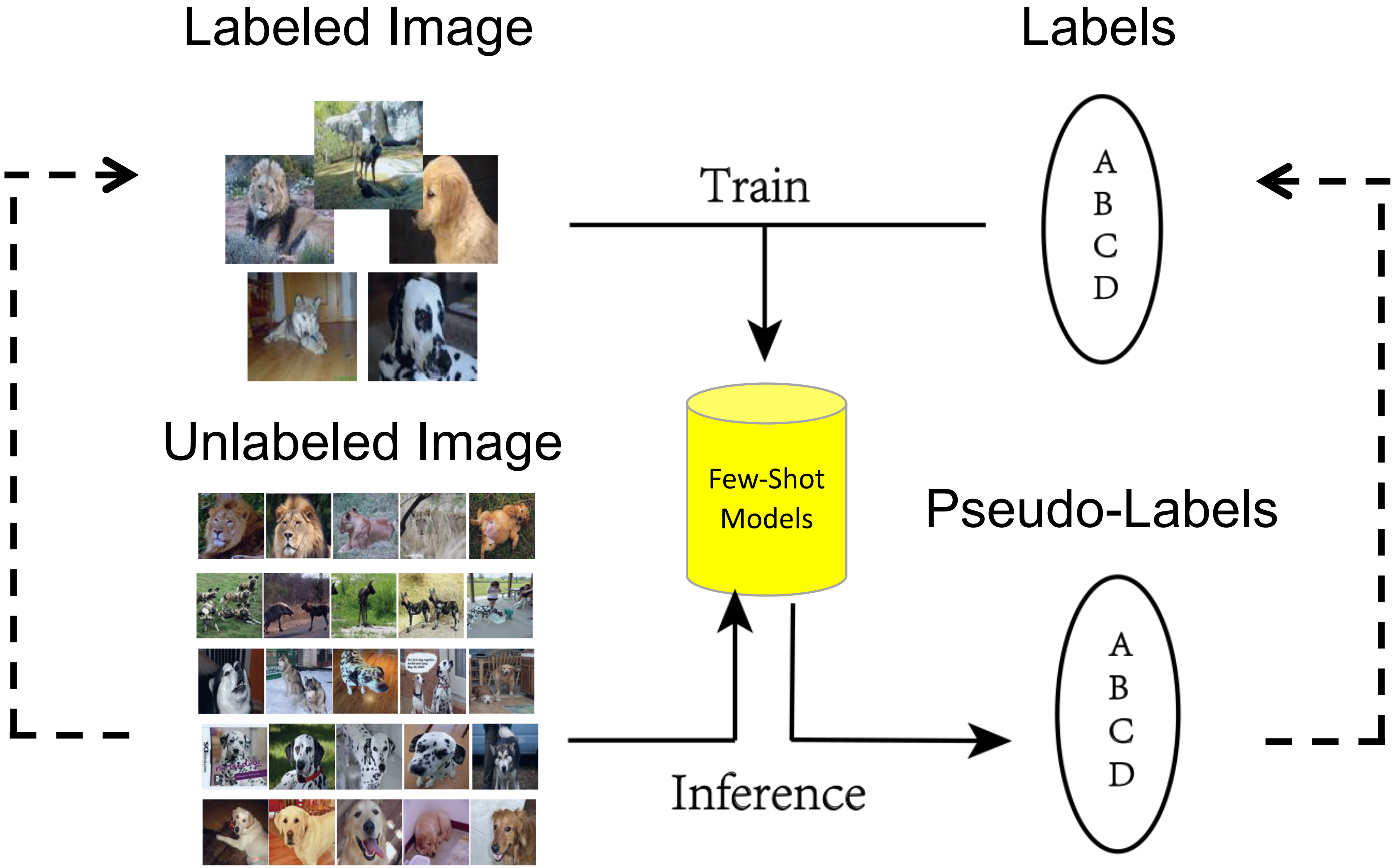
Few-shot Learning by Unlabeled Data

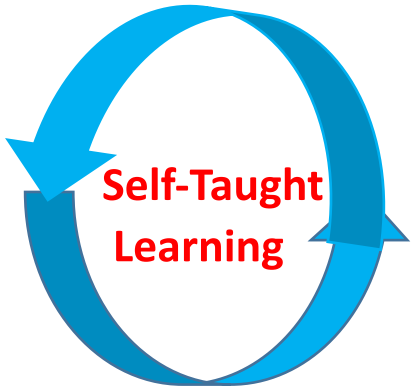


Wang et al. Instance Credibility Inference for Few-Shot Learning. CVPR 2020

Wang et al. How to Trust Unlabeled Data? Instance Credibility Inference for Few-Shot Learning. IEEE TPAMI 2021

Few-shot Learning by Unlabeled Data





Wang et al. Instance Credibility Inference for Few-Shot Learning. CVPR 2020

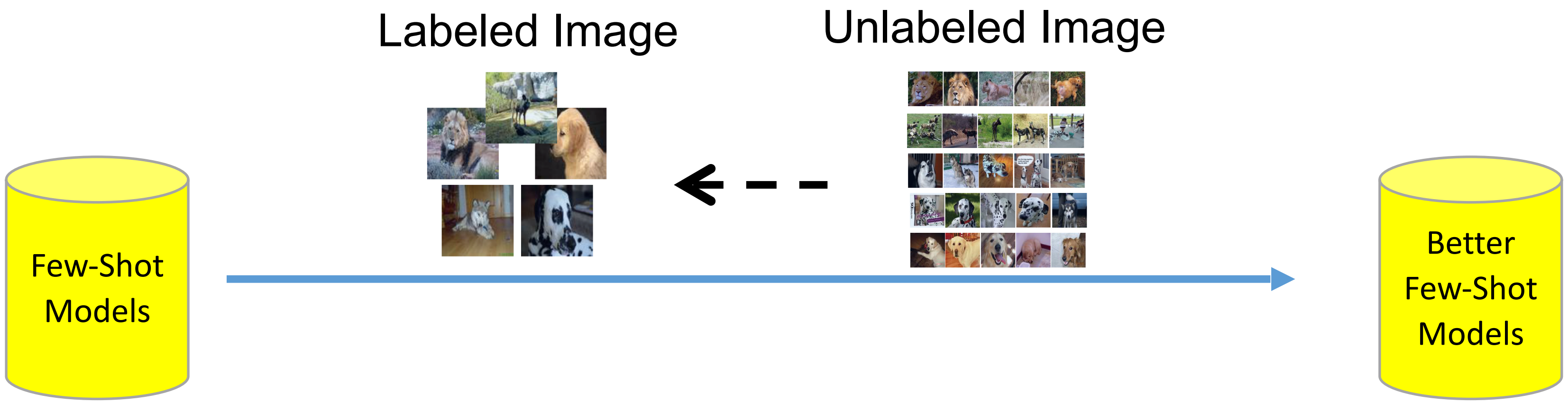
Wang et al. How to Trust Unlabeled Data? Instance Credibility Inference for Few-Shot Learning. IEEE TPAMI 2021

Sparse Labels in Semi-supervised Few-Shot Learning

We will introduce the details in the next talk.

[Wang et al. CVPR2020/TPAMI2021]

Sparse Labels in Semi-supervised Few-Shot Learning

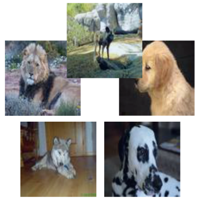


We will introduce the details in the next talk.

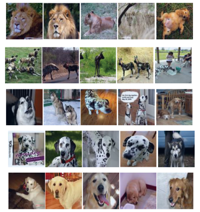
[Wang et al. CVPR2020/TPAMI2021]

Sparse Labels in Semi-supervised Few-Shot Learning

Labeled Image



Unlabeled Image



Few-Shot Models

Better Few-Shot Models

$$y_i = \boldsymbol{x}_i^\top \beta + {\color{red}\gamma_i} + \varepsilon$$

$$\downarrow$$

$$\underset{\boldsymbol{\beta},\boldsymbol{\gamma}}{\operatorname{argmin}} := \|\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{\gamma}\|_{\mathrm{F}}^2 + \lambda R(\boldsymbol{\gamma})$$

$$\downarrow$$

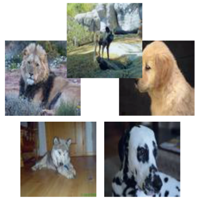
$$\underset{\boldsymbol{\gamma}}{\operatorname{argmin}} \left\|\tilde{\boldsymbol{Y}} - \tilde{\boldsymbol{X}}\boldsymbol{\gamma}\right\|_{\mathrm{F}}^2 + \lambda R(\boldsymbol{\gamma})$$

We will introduce the details in the next talk.

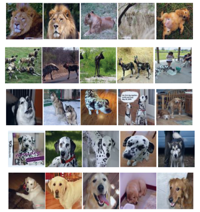
[Wang et al. CVPR2020/TPAMI2021]

Sparse Labels in Semi-supervised Few-Shot Learning

Labeled Image



Unlabeled Image



Few-Shot Models

Better Few-Shot Models

$$y_i = \boldsymbol{x}_i^\top \beta + \gamma_i + \varepsilon$$

$$\downarrow$$

$$\underset{\boldsymbol{\beta},\boldsymbol{\gamma}}{\operatorname{argmin}} := \left\|\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{\gamma}\right\|_{\mathrm{F}}^2 + \lambda R(\boldsymbol{\gamma})$$

$$\downarrow$$

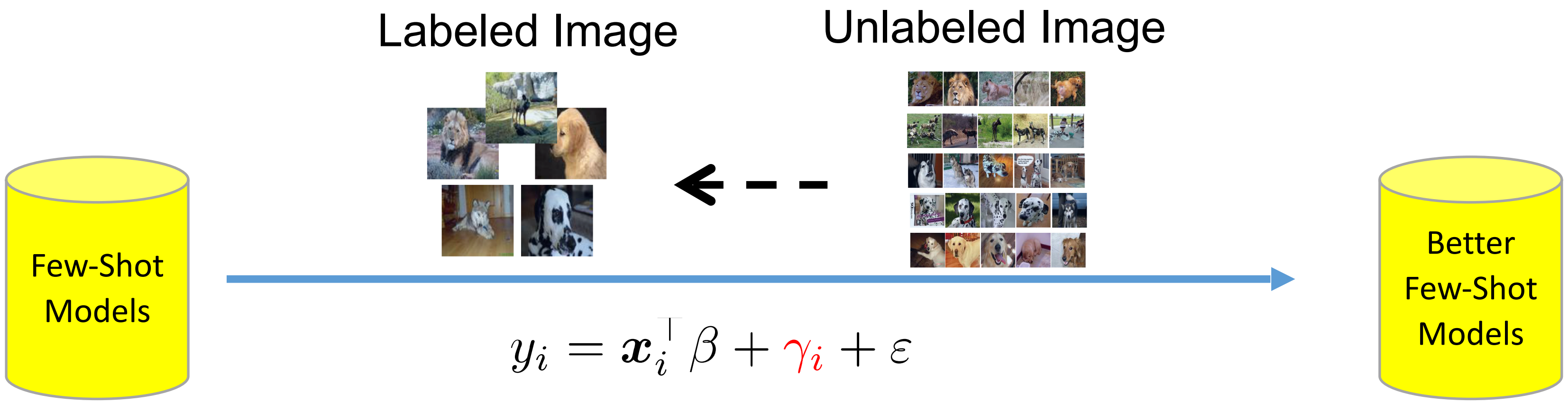
A Linear regression problem!

$$\underset{\boldsymbol{\gamma}}{\operatorname{argmin}} \left\|\tilde{\boldsymbol{Y}} - \tilde{\boldsymbol{X}}\boldsymbol{\gamma}\right\|_{\mathrm{F}}^2 + \lambda R(\boldsymbol{\gamma})$$

We will introduce the details in the next talk.

[Wang et al. CVPR2020/TPAMI2021]

Sparse Labels in Semi-supervised Few-Shot Learning



$$y_i = \boldsymbol{x}_i^\top \beta + \gamma_i + \varepsilon$$

$$\downarrow$$

$$\underset{\boldsymbol{\beta},\boldsymbol{\gamma}}{\operatorname{argmin}} := \|\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{\gamma}\|_{\mathrm{F}}^2 + \lambda R(\boldsymbol{\gamma})$$

How to select λ?

$$\downarrow$$

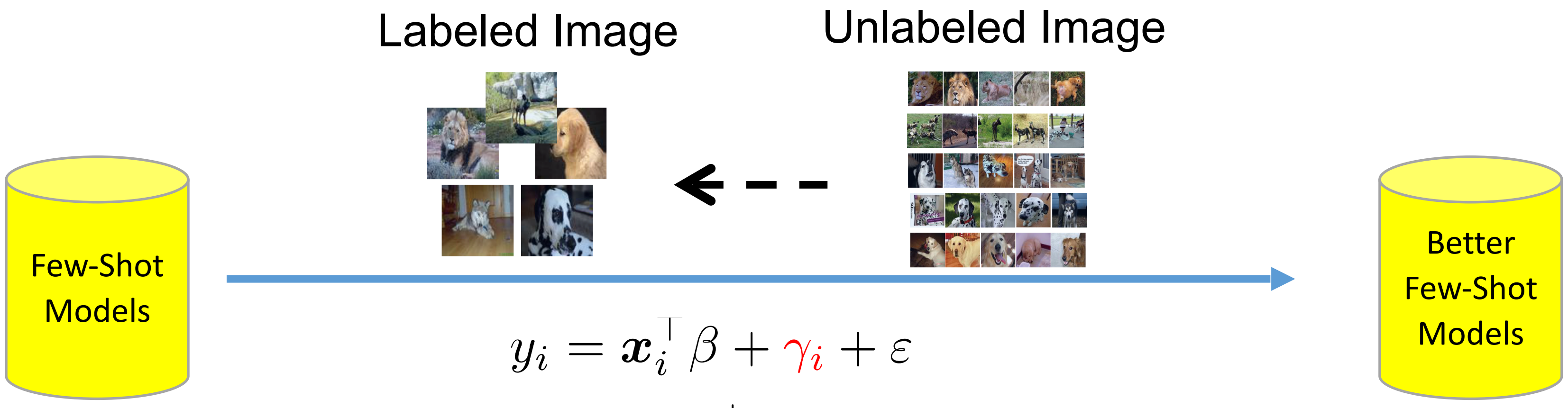
A Linear regression problem!

$$\underset{\boldsymbol{\gamma}}{\operatorname{argmin}} \left\|\tilde{\boldsymbol{Y}} - \tilde{\boldsymbol{X}}\boldsymbol{\gamma}\right\|_{\mathrm{F}}^2 + \lambda R(\boldsymbol{\gamma})$$

We will introduce the details in the next talk.

[Wang et al. CVPR2020/TPAMI2021]

Sparse Labels in Semi-supervised Few-Shot Learning



$$y_i = \boldsymbol{x}_i^\top \beta + \gamma_i + \varepsilon$$

$$\downarrow$$

$$\underset{\boldsymbol{\beta},\boldsymbol{\gamma}}{\operatorname{argmin}} := \|\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{\gamma}\|_{\mathrm{F}}^2 + \lambda R(\boldsymbol{\gamma})$$

$$\downarrow$$

A Linear regression problem!

$$\underset{\boldsymbol{\gamma}}{\operatorname{argmin}} \left\|\tilde{\boldsymbol{Y}} - \tilde{\boldsymbol{X}}\boldsymbol{\gamma}\right\|_{\mathrm{F}}^2 + \lambda R(\boldsymbol{\gamma})$$

How to select λ?

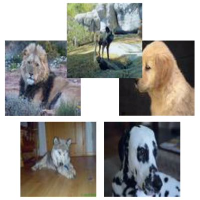
- heuristics rules $\lambda = 2.5\hat{\sigma}$?

We will introduce the details in the next talk.

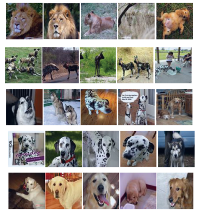
[Wang et al. CVPR2020/TPAMI2021]

Sparse Labels in Semi-supervised Few-Shot Learning

Labeled Image



Unlabeled Image



Few-Shot Models

Better Few-Shot Models

$$y_i = \boldsymbol{x}_i^\top \beta + \textcolor{red}{\gamma_i} + \varepsilon$$

$$\downarrow$$

$$\underset{\boldsymbol{\beta},\boldsymbol{\gamma}}{\operatorname{argmin}} := \|\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{\gamma}\|_{\mathrm{F}}^2 + \lambda R(\boldsymbol{\gamma})$$

$$\downarrow$$

A Linear regression problem!

$$\underset{\boldsymbol{\gamma}}{\operatorname{argmin}} \left\| \tilde{\boldsymbol{Y}} - \tilde{\boldsymbol{X}}\boldsymbol{\gamma} \right\|_{\mathrm{F}}^2 + \lambda R(\boldsymbol{\gamma})$$

How to select λ?

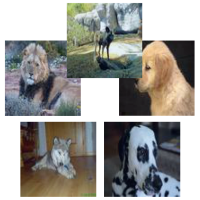
- heuristics rules $\lambda = 2.5\hat{\sigma}$?
- Cross-validation?

We will introduce the details in the next talk.

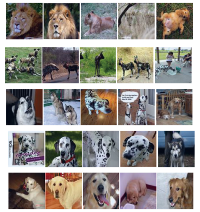
[Wang et al. CVPR2020/TPAMI2021]

Sparse Labels in Semi-supervised Few-Shot Learning

Labeled Image



Unlabeled Image



Few-Shot Models

Better Few-Shot Models

$$y_i = \boldsymbol{x}_i^\top \beta + {\color{red}\gamma_i} + \varepsilon$$

$$\downarrow$$

$$\underset{\boldsymbol{\beta},\boldsymbol{\gamma}}{\operatorname{argmin}} := \|\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{\gamma}\|_{\mathrm{F}}^2 + \lambda R(\boldsymbol{\gamma})$$

$$\downarrow$$

A Linear regression problem!

$$\underset{\boldsymbol{\gamma}}{\operatorname{argmin}} \left\|\tilde{\boldsymbol{Y}} - \tilde{\boldsymbol{X}}\boldsymbol{\gamma}\right\|_{\mathrm{F}}^2 + \lambda R(\boldsymbol{\gamma})$$

How to select λ?

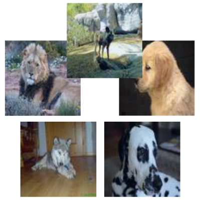
- heuristics rules $\lambda = 2.5\hat{\sigma}$?
- Cross-validation?
- Data adaptive techniques?

We will introduce the details in the next talk.

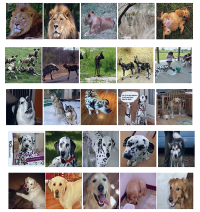
[Wang et al. CVPR2020/TPAMI2021]

Sparse Labels in Semi-supervised Few-Shot Learning

Labeled Image



Unlabeled Image



Few-Shot Models

Better Few-Shot Models

$$y_i = \boldsymbol{x}_i^\top \beta + {\color{red}\gamma_i} + \varepsilon$$

$$\downarrow$$

$$\underset{\boldsymbol{\beta},\boldsymbol{\gamma}}{\operatorname{argmin}} := \|\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{\gamma}\|_{\mathrm{F}}^2 + \lambda R(\boldsymbol{\gamma})$$

$$\downarrow$$

A Linear regression problem!

$$\underset{\boldsymbol{\gamma}}{\operatorname{argmin}} \left\| \tilde{\boldsymbol{Y}} - \tilde{\boldsymbol{X}}\boldsymbol{\gamma} \right\|_{\mathrm{F}}^2 + \lambda R(\boldsymbol{\gamma})$$

How to select λ?

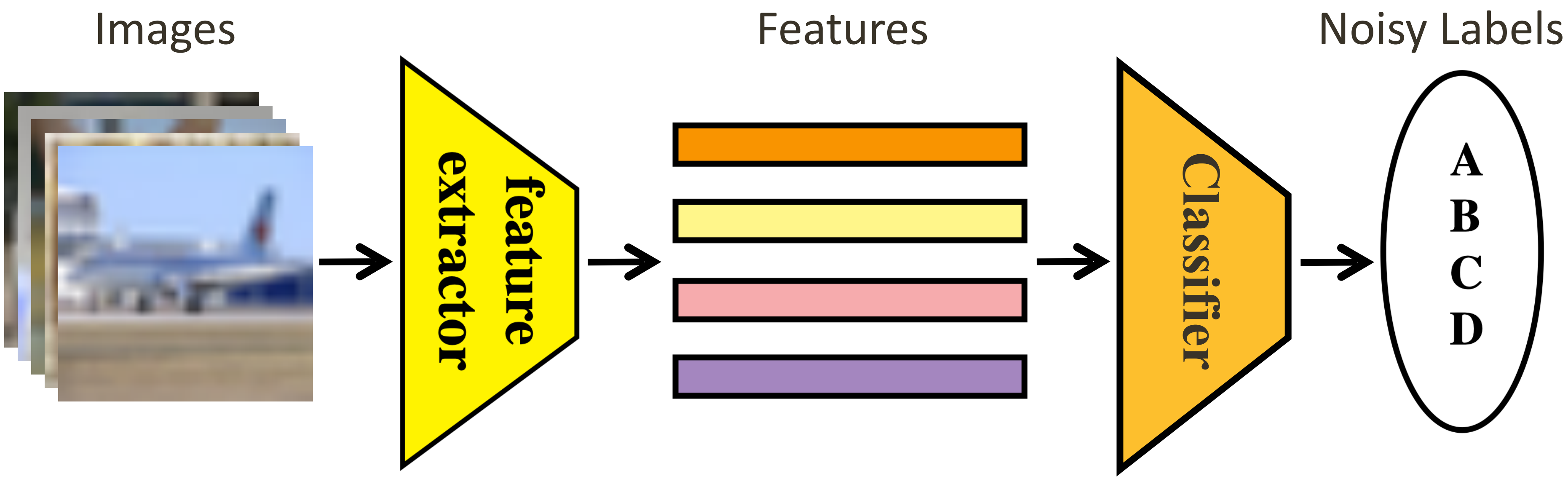
- heuristics rules $\lambda = 2.5\hat{\sigma}$?
- Cross-validation?
- Data adaptive techniques?
- AIC, BIC?

We will introduce the details in the next talk.

[Wang et al. CVPR2020/TPAMI2021]

Learning Sparsity in Learning with Noisy Labels

Stage 1:
Feature Learning

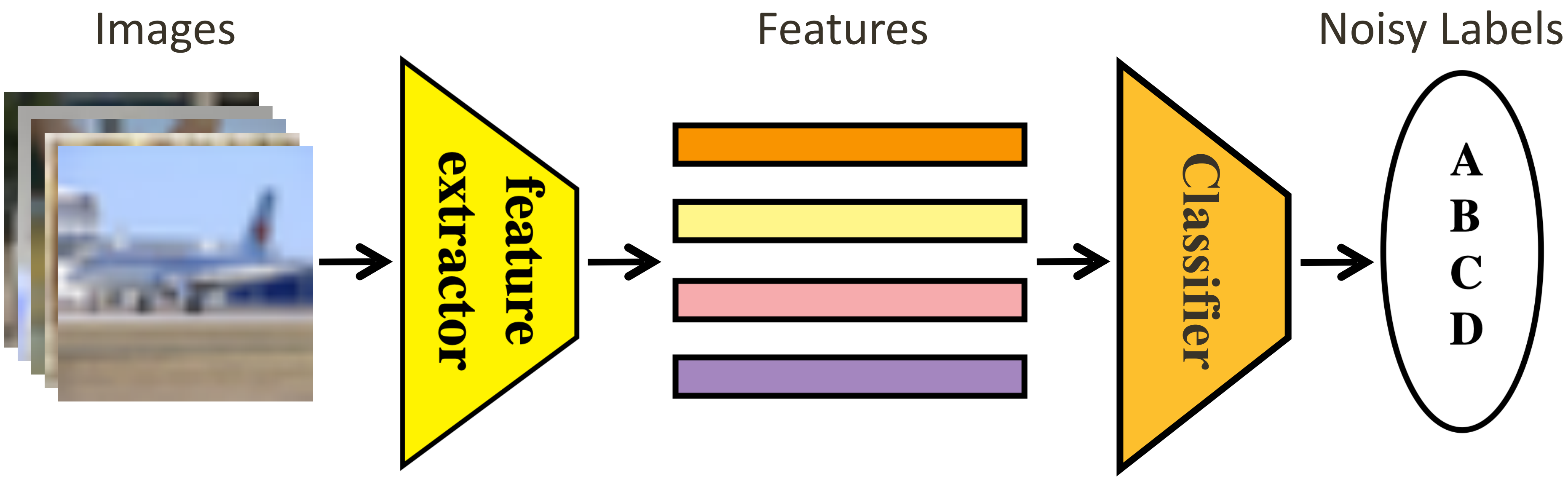


Stage 2:
Sample Selection

[Wang et al. CVPR2022]

Learning Sparsity in Learning with Noisy Labels

Stage 1:
Feature Learning

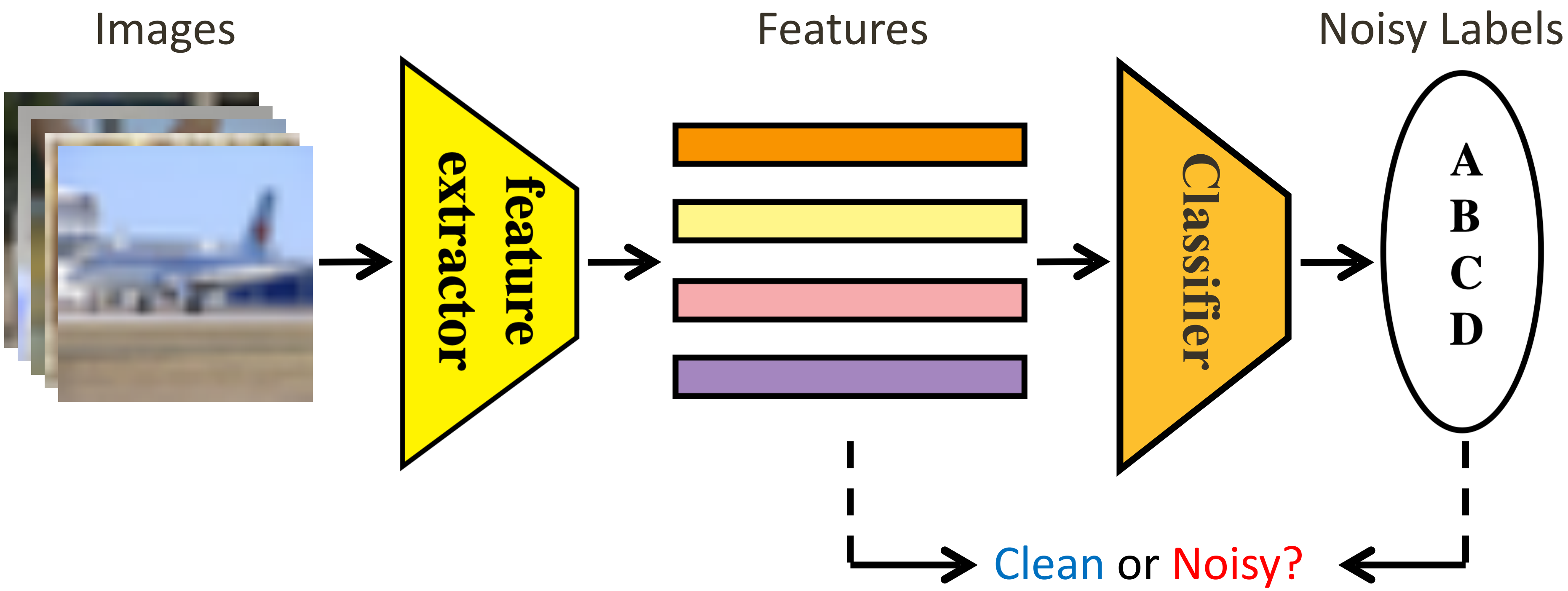


Stage 2:
Sample Selection

[Wang et al. CVPR2022]

Learning Sparsity in Learning with Noisy Labels

Stage 1:
Feature Learning

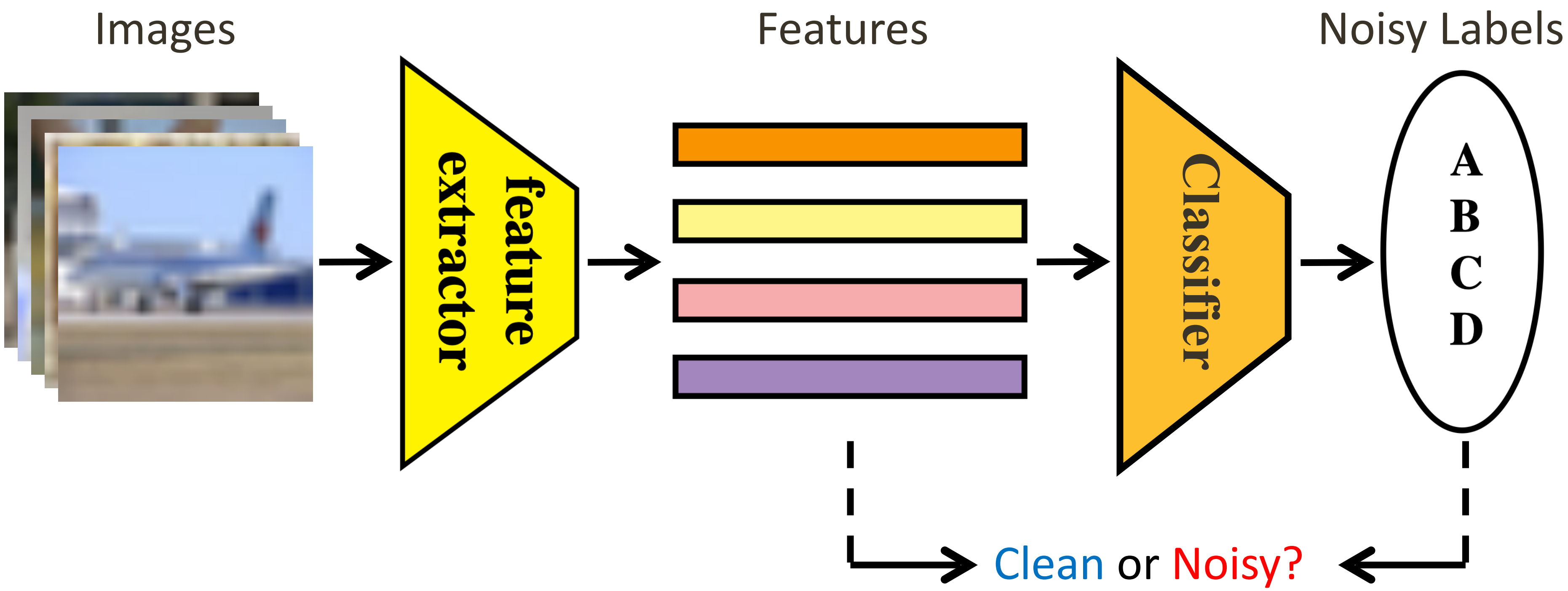


Stage 2:
Sample Selection

[Wang et al. CVPR2022]

Learning Sparsity in Learning with Noisy Labels

Stage 1:
Feature Learning



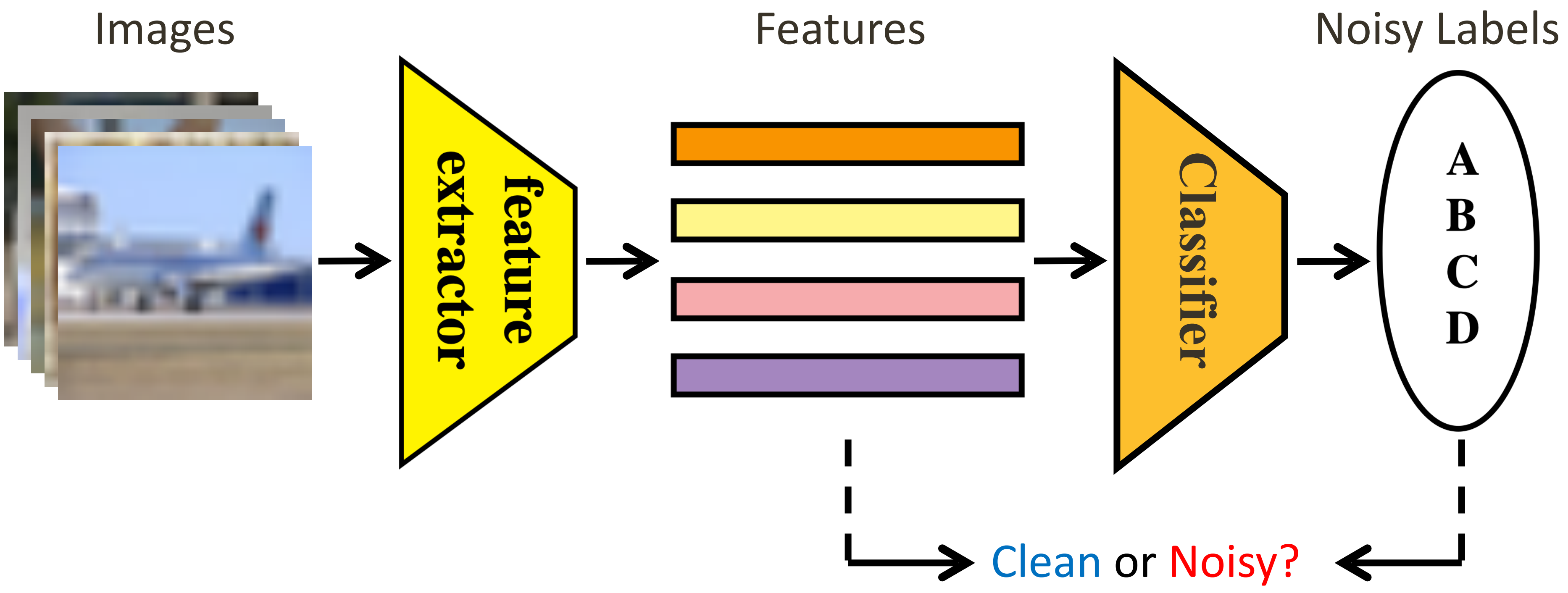
$$y_i = \boldsymbol{x}_i^\top \beta + \gamma_i + \varepsilon$$

Stage 2:
Sample Selection

[Wang et al. CVPR2022]

Learning Sparsity in Learning with Noisy Labels

Stage 1:
Feature Learning



Stage 2:
Sample Selection

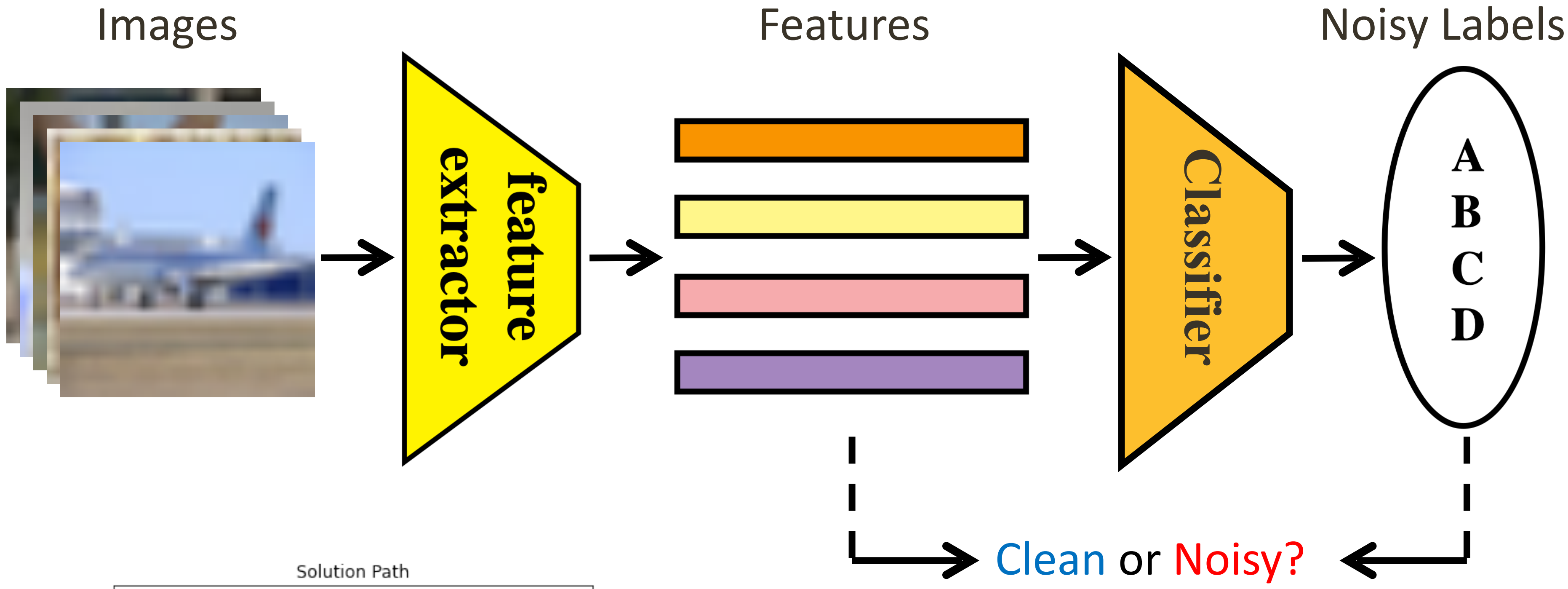
$$y_i = \boldsymbol{x}_i^\top \beta + \gamma_i + \varepsilon$$

$$\underset{\boldsymbol{\gamma}}{\operatorname{argmin}} \left\| \tilde{\boldsymbol{Y}} - \tilde{\boldsymbol{X}} \boldsymbol{\gamma} \right\|_{\mathrm{F}}^2 + \sum_{i=1}^{n} P\left(\gamma_i; \lambda_i\right)$$

[Wang et al. CVPR2022]

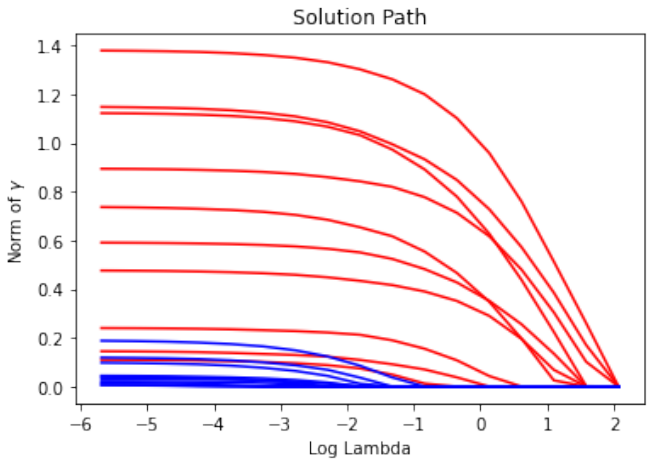
Learning Sparsity in Learning with Noisy Labels

Stage 1:
Feature Learning



$$y_i = \boldsymbol{x}_i^\top \beta + \gamma_i + \varepsilon$$

Stage 2:
Sample Selection

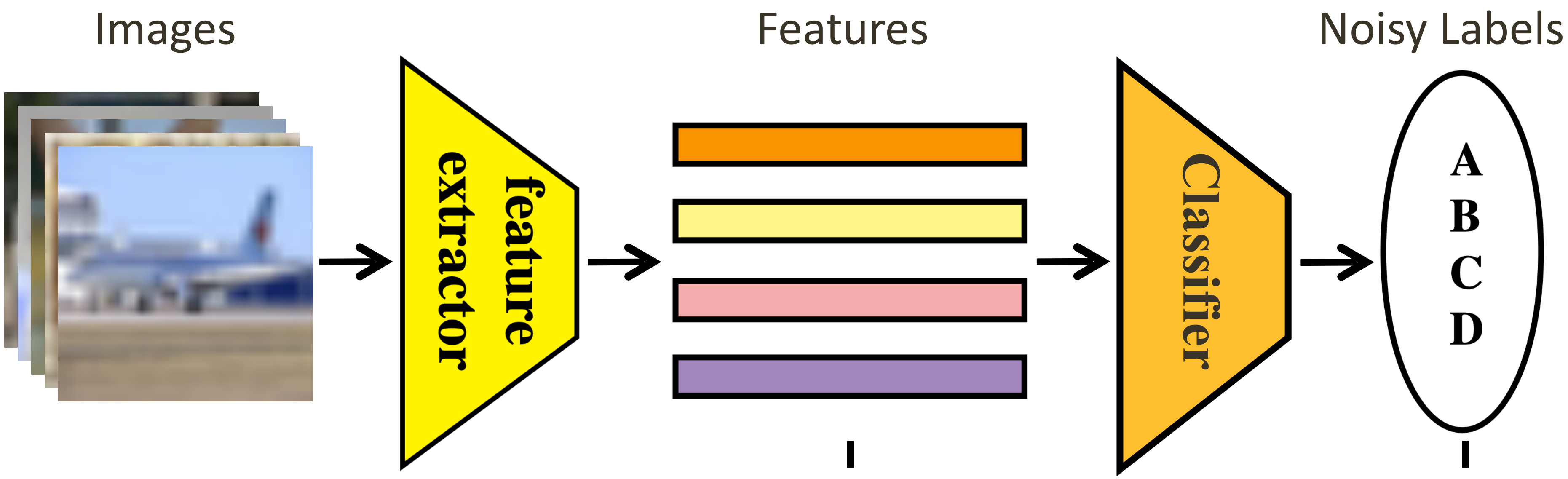


$$\underset{\boldsymbol{\gamma}}{\operatorname{argmin}} \left\| \tilde{\boldsymbol{Y}} - \tilde{\boldsymbol{X}}\boldsymbol{\gamma} \right\|_{\mathrm{F}}^2 + \sum_{i=1}^{n} P\left(\gamma_i; \lambda_i\right)$$

[Wang et al. CVPR2022]

Learning Sparsity in Learning with Noisy Labels

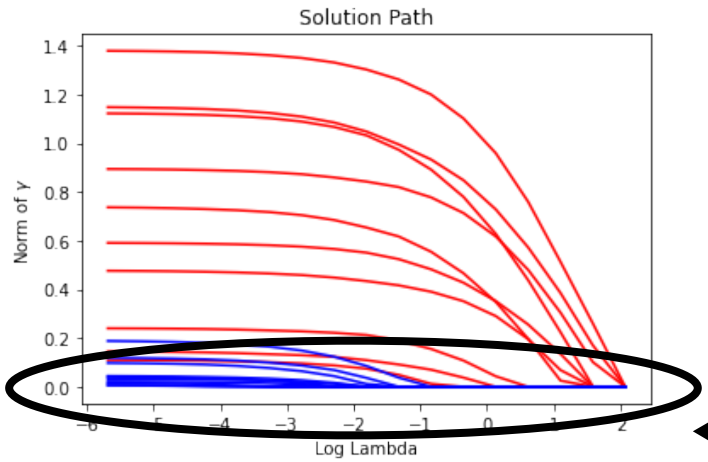
Stage 1:
Feature Learning



Clean or Noisy?

$$y_i = \boldsymbol{x}_i^\top \beta + \gamma_i + \varepsilon$$

Stage 2:
Sample Selection



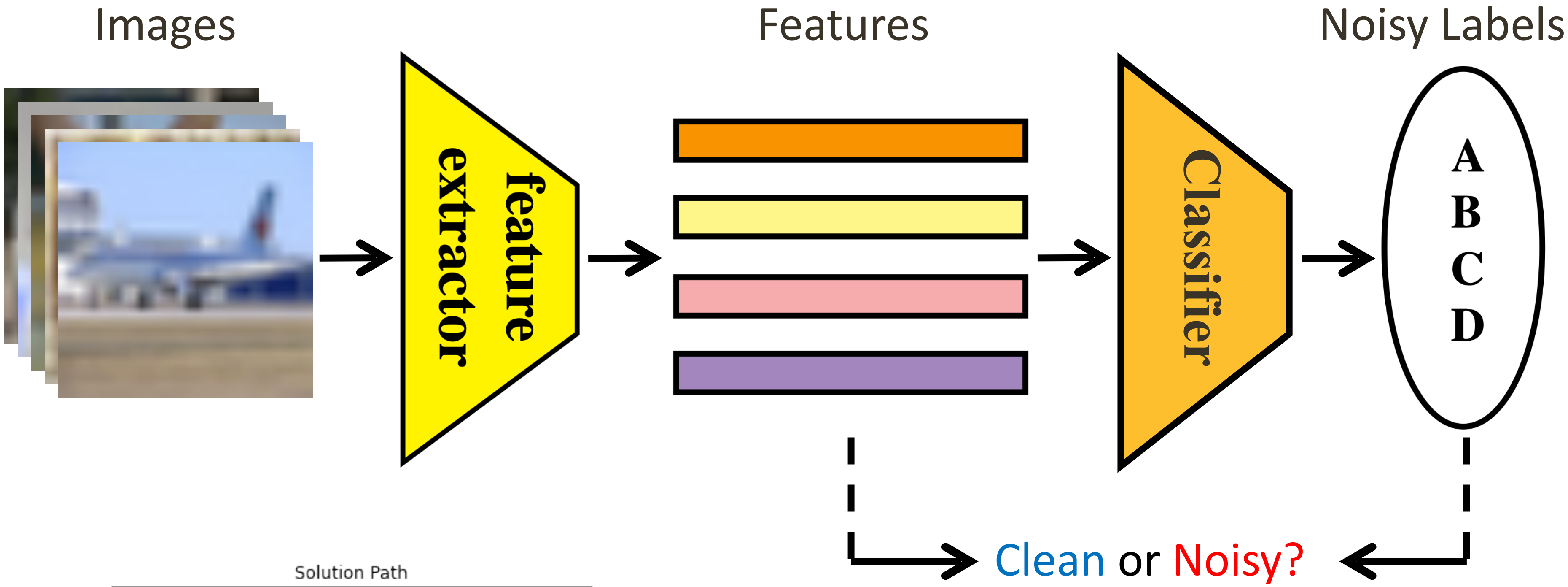
$$\underset{\gamma}{\operatorname{argmin}} \left\| \tilde{\boldsymbol{Y}} - \tilde{\boldsymbol{X}} \boldsymbol{\gamma} \right\|_F^2 + \sum_{i=1}^{n} P(\gamma_i; \lambda_i)$$

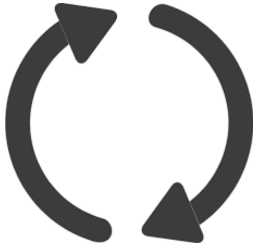
[Wang et al. CVPR2022]

大数据学院 School of Data Science

Learning Sparsity in Learning with Noisy Labels

Stage 1:
Feature Learning

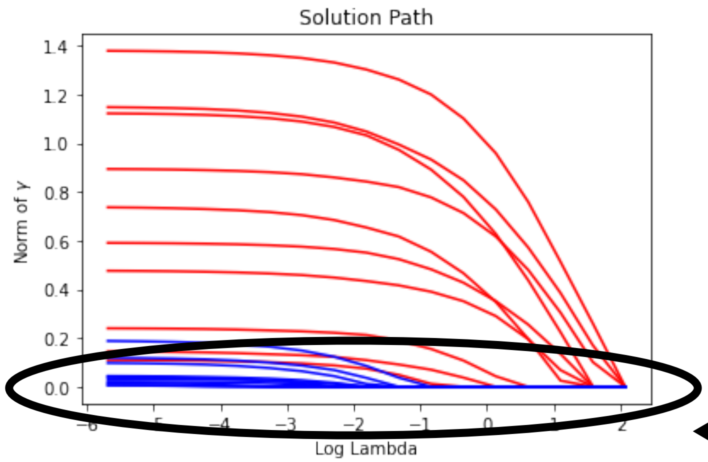




Stage 2:
Sample Selection

$$y_i = \boldsymbol{x}_i^\top \beta + \gamma_i + \varepsilon$$

$$\underset{\gamma}{\operatorname{argmin}} \left\| \tilde{\boldsymbol{Y}} - \tilde{\boldsymbol{X}} \boldsymbol{\gamma} \right\|_{\mathrm{F}}^2 + \sum_{i=1}^{n} P(\gamma_i; \lambda_i)$$



[Wang et al. CVPR2022]

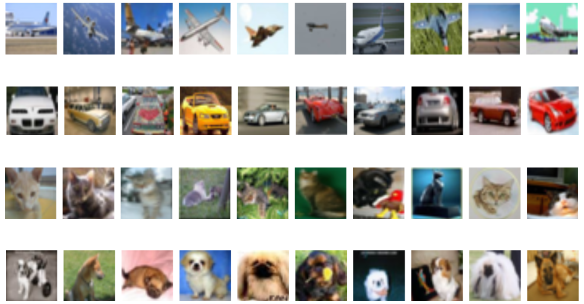
Make it scalable to large datasets

Wang et al. Scalable Penalized Regression for Noise Detection in Learning with Noisy Labels. CVPR 2022

Make it scalable to large datasets

Wang et al. Scalable Penalized Regression for Noise Detection in Learning with Noisy Labels. CVPR 2022

Make it scalable to large datasets

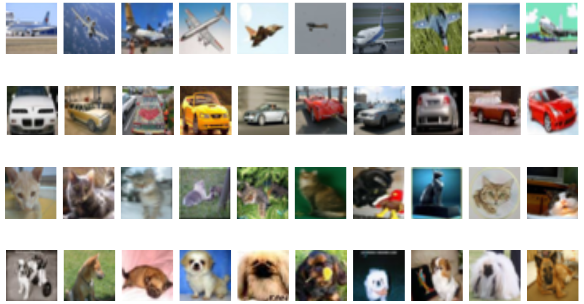


Compute

Class Similarity

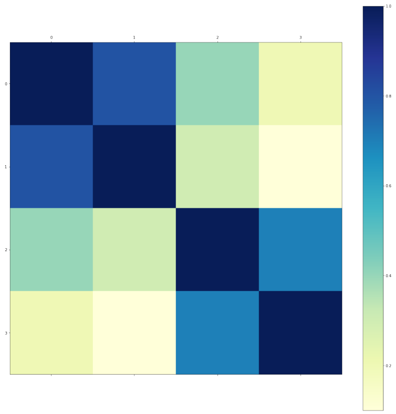
Wang et al. Scalable Penalized Regression for Noise Detection in Learning with Noisy Labels. CVPR 2022

Make it scalable to large datasets



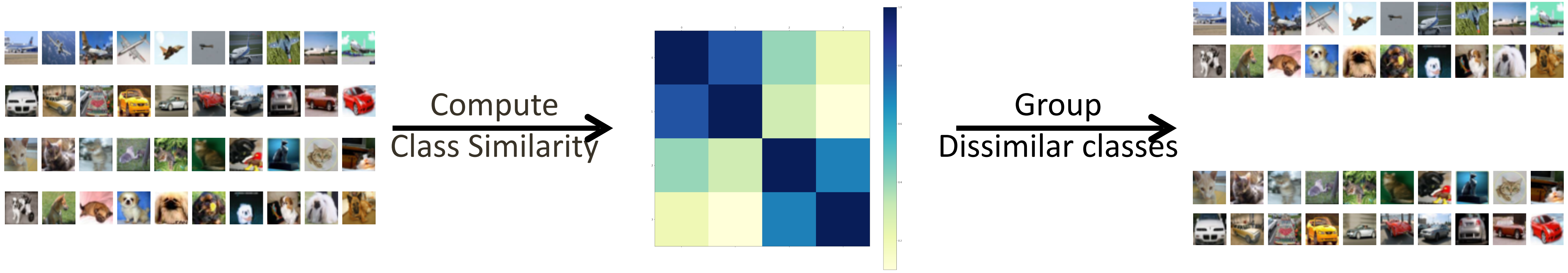
Compute

Class Similarity



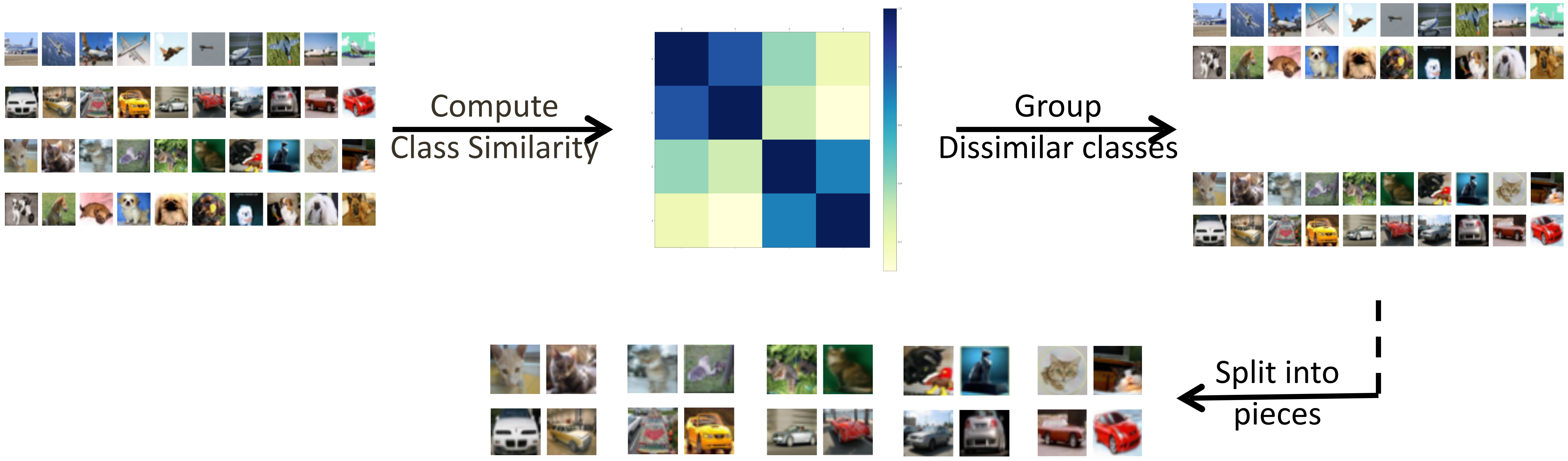
Wang et al. Scalable Penalized Regression for Noise Detection in Learning with Noisy Labels. CVPR 2022

Make it scalable to large datasets



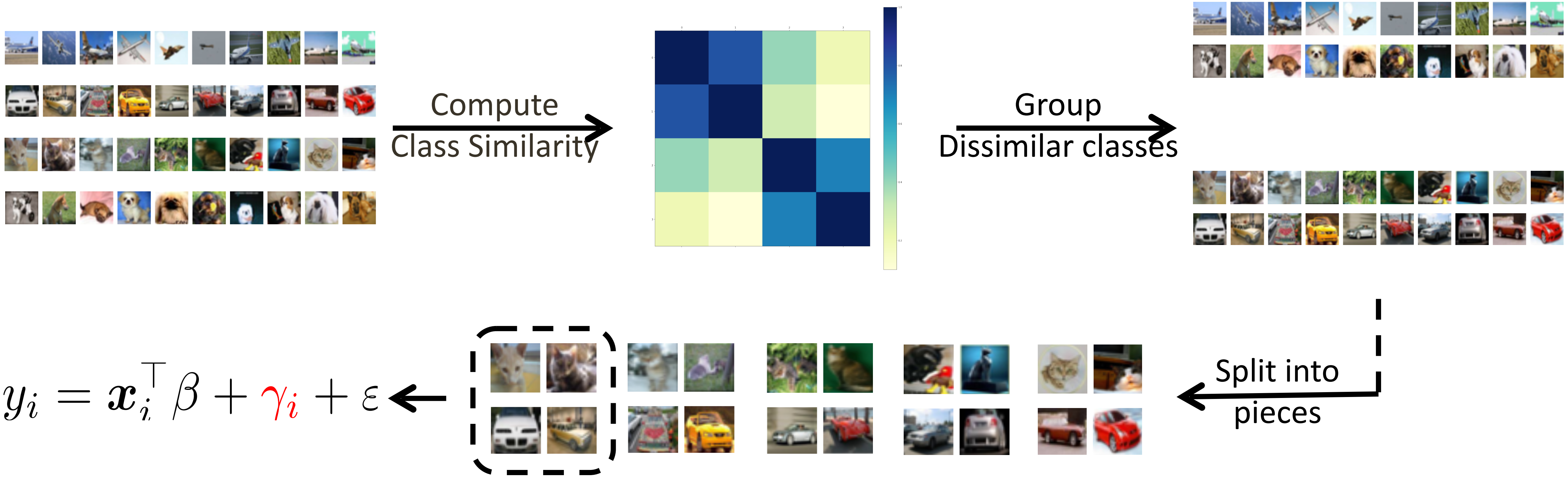
Wang et al. Scalable Penalized Regression for Noise Detection in Learning with Noisy Labels. CVPR 2022

Make it scalable to large datasets



Wang et al. Scalable Penalized Regression for Noise Detection in Learning with Noisy Labels. CVPR 2022

Make it scalable to large datasets



$$y_i = \boldsymbol{x}_i^\top \beta + {\color{red}\gamma_i} + \varepsilon$$

Wang et al. Scalable Penalized Regression for Noise Detection in Learning with Noisy Labels. CVPR 2022

Sparse Learning in Robust Ranking

Fu et al. Interestingness Prediction by Robust Learning to Rank. ECCV 2014
Fu et al. Robust Subjective Visual Property Prediction from Crowdsourced Pairwise Labels. IEEE TPAMI 2016

Sparse Learning in Robust Ranking

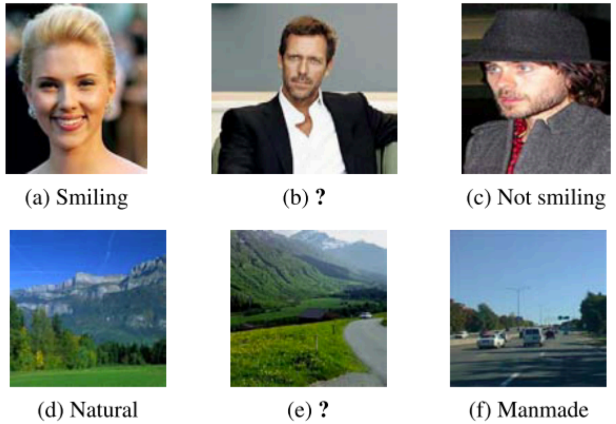
?

Who is smiling more?

Fu et al. Interestingness Prediction by Robust Learning to Rank. ECCV 2014
Fu et al. Robust Subjective Visual Property Prediction from Crowdsourced Pairwise Labels. IEEE TPAMI 2016

Sparse Learning in Robust Ranking

Who is smiling more?

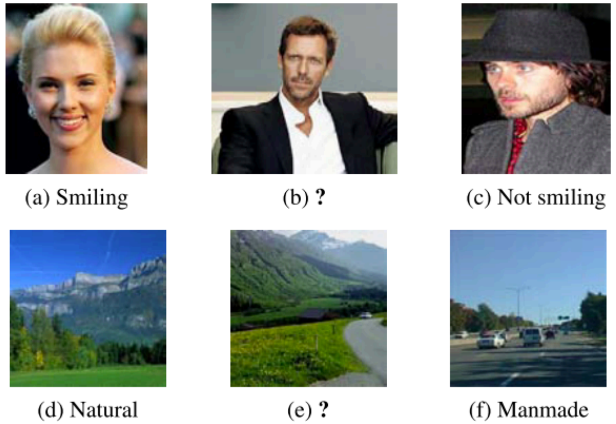


Parikh et al. Relative Attributes, ICCV 2011, Marr Prize Paper.

Fu et al. Interestingness Prediction by Robust Learning to Rank. ECCV 2014

Fu et al. Robust Subjective Visual Property Prediction from Crowdsourced Pairwise Labels. IEEE TPAMI 2016

Sparse Learning in Robust Ranking



Parikh et al. Relative Attributes, ICCV 2011, Marr Prize Paper.

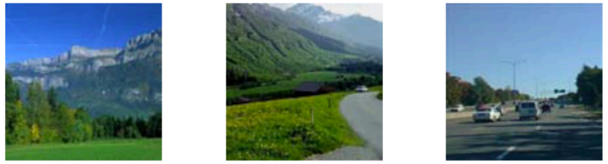
Who is smiling more?

1. Cultural factors
2. Psychological factors: Halo Effects
3. Ambiguous comparisons
4. Malicious/Lazy annotators

Fu et al. Interestingness Prediction by Robust Learning to Rank. ECCV 2014
Fu et al. Robust Subjective Visual Property Prediction from Crowdsourced Pairwise Labels. IEEE TPAMI 2016

Sparse Learning in Robust Ranking

(a) Smiling (b) ? (c) Not smiling



(d) Natural (e) ? (f) Manmade

Parikh et al. Relative Attributes, ICCV 2011, Marr Prize Paper.

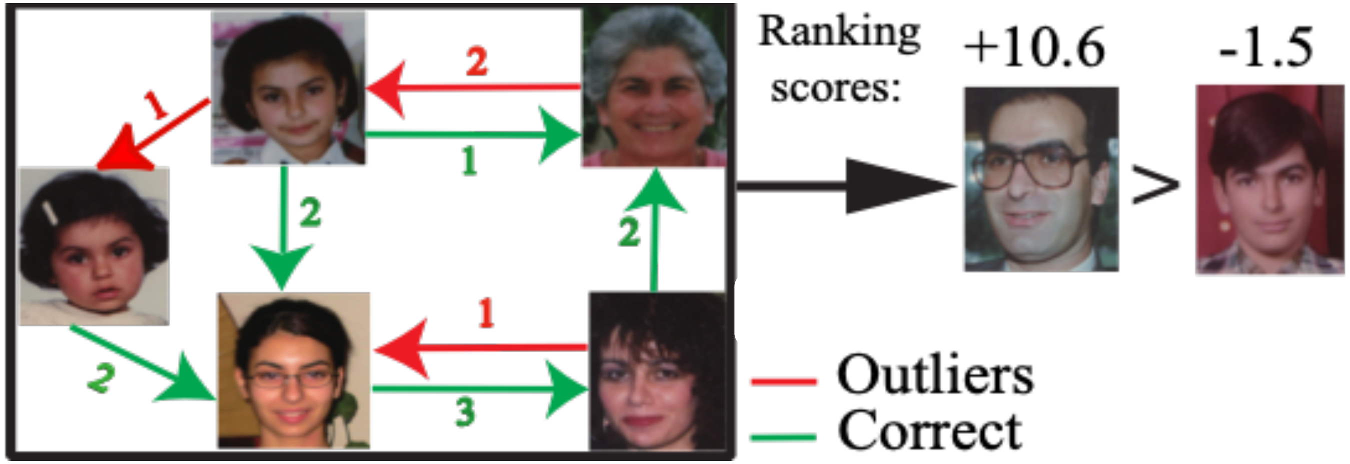
Who is smiling more?

1. Cultural factors
2. Psychological factors: Halo Effects
3. Ambiguous comparisons
4. Malicious/Lazy annotators

Fu et al. Interestingness Prediction by Robust Learning to Rank. ECCV 2014

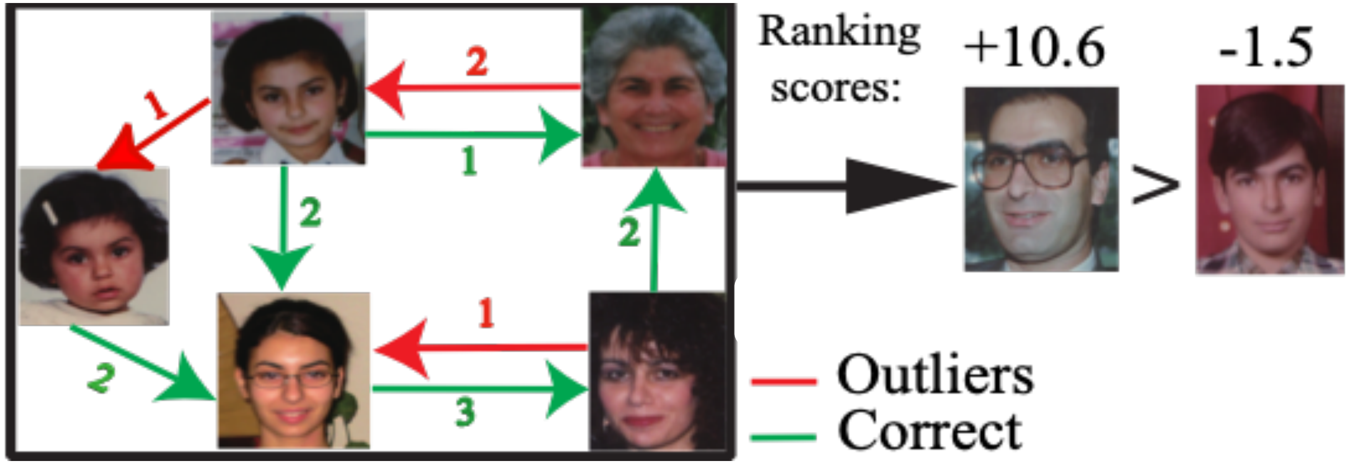
Fu et al. Robust Subjective Visual Property Prediction from Crowdsourced Pairwise Labels. IEEE TPAMI 2016

Sparse Learning in Robust Ranking



Fu et al. Interestingness Prediction by Robust Learning to Rank. ECCV 2014
Fu et al. Robust Subjective Visual Property Prediction from Crowdsourced Pairwise Labels. IEEE TPAMI 2016

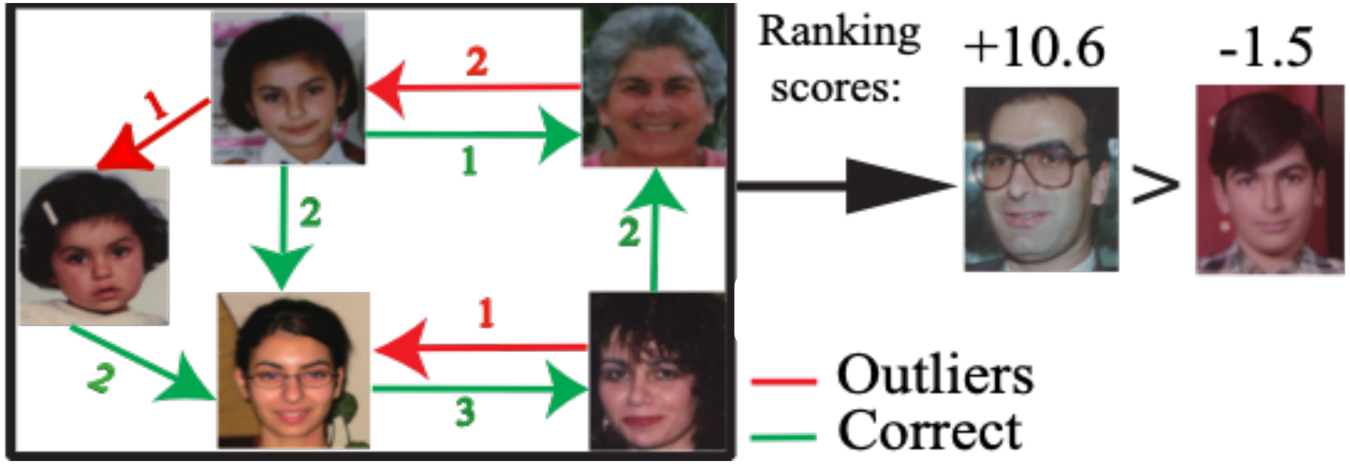
Sparse Learning in Robust Ranking



Ranking age of images by learning from crowdsourced pairs as the directed graph $G = (V, E)$

Fu et al. Interestingness Prediction by Robust Learning to Rank. ECCV 2014
Fu et al. Robust Subjective Visual Property Prediction from Crowdsourced Pairwise Labels. IEEE TPAMI 2016

Sparse Learning in Robust Ranking



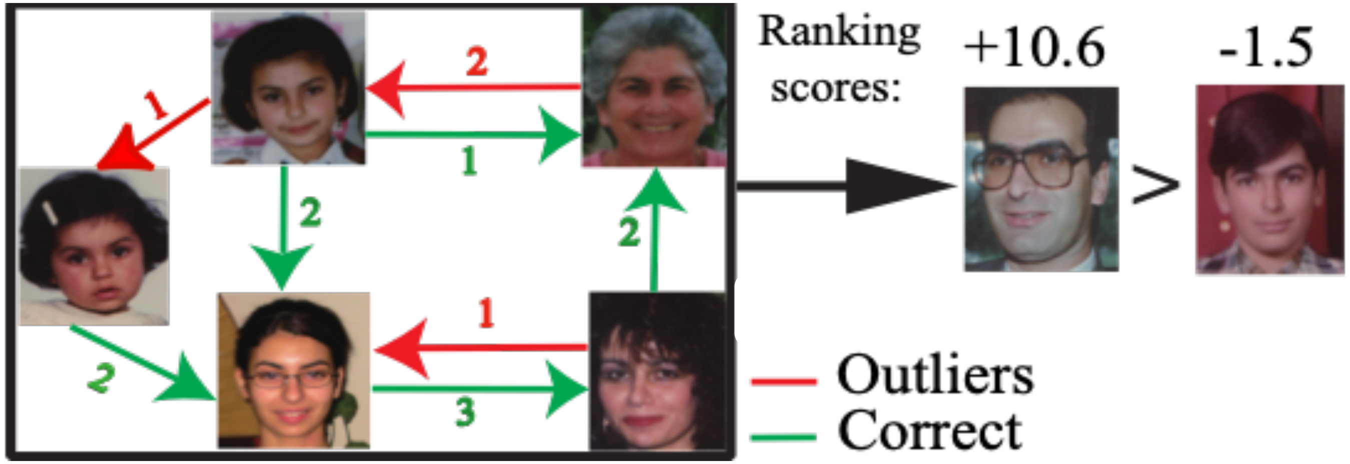
Ranking age of images by learning from crowdsourced pairs as the directed graph $G = (V, E)$

Each edge is modeled as $$Y_{ij} = \beta^T \phi_i - \beta^T \phi_j + \gamma_{ij}$$

Fu et al. Interestingness Prediction by Robust Learning to Rank. ECCV 2014

Fu et al. Robust Subjective Visual Property Prediction from Crowdsourced Pairwise Labels. IEEE TPAMI 2016

Sparse Learning in Robust Ranking



Ranking age of images by learning from crowdsourced pairs as the directed graph $G = (V, E)$

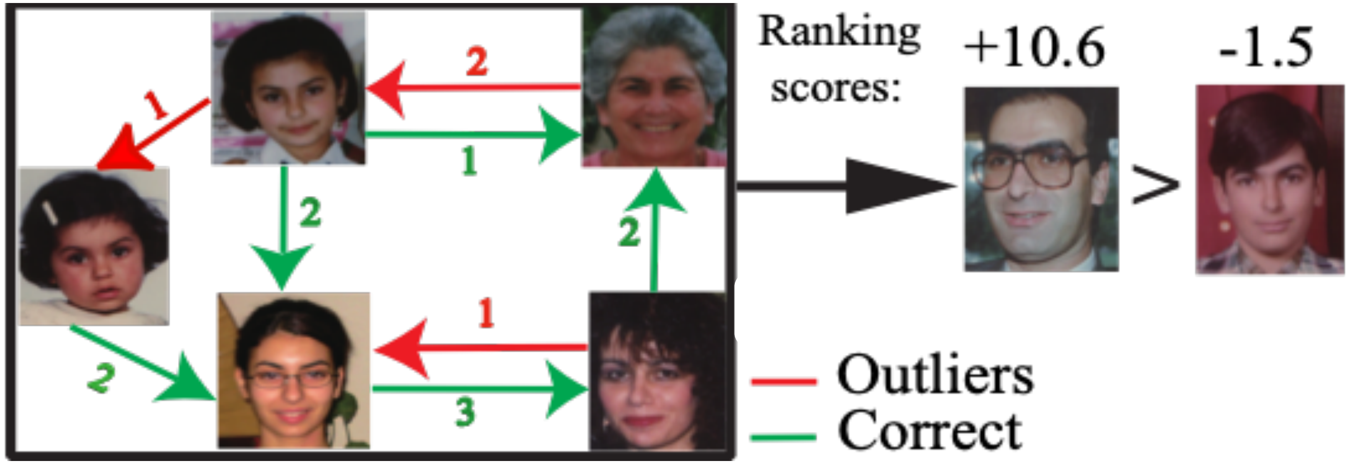
Each edge is modeled as $$Y_{ij} = \beta^T \phi_i - \beta^T \phi_j + \gamma_{ij}$$

The edge of whole dataset is $$Y = C\Phi\beta + \epsilon + \Gamma$$

Fu et al. Interestingness Prediction by Robust Learning to Rank. ECCV 2014

Fu et al. Robust Subjective Visual Property Prediction from Crowdsourced Pairwise Labels. IEEE TPAMI 2016

Sparse Learning in Robust Ranking



Ranking age of images by learning from crowdsourced pairs as the directed graph $G = (V, E)$

Each edge is modeled as $Y_{ij} = \beta^T \phi_i - \beta^T \phi_j + \gamma_{ij}$

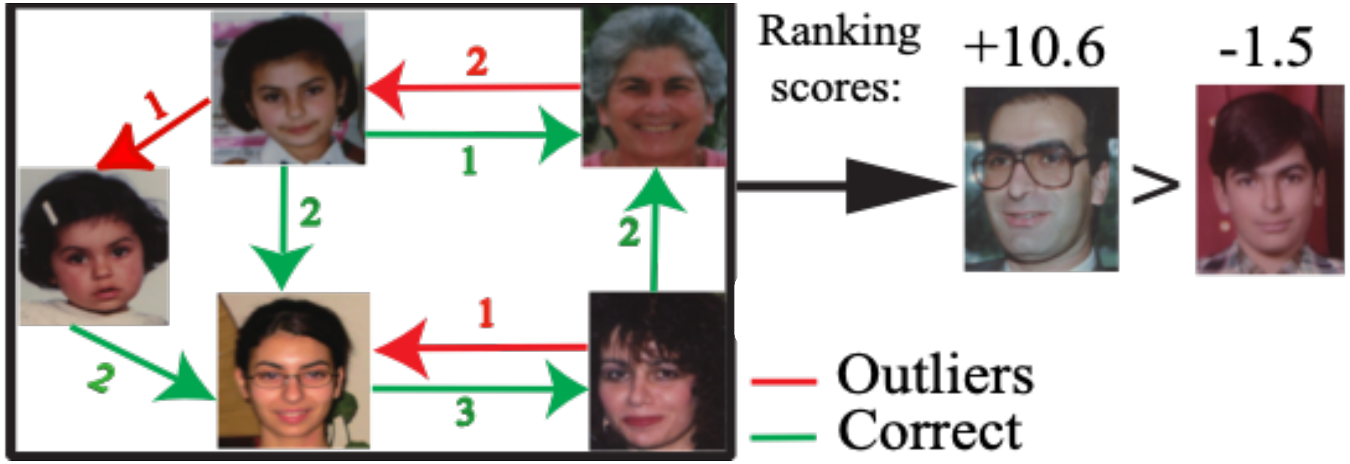
The edge of whole dataset is $Y = C\Phi\beta + \epsilon + \Gamma$

$$\min_{\beta,\Gamma} \frac{1}{2} \|Y - C\Phi\beta - \Gamma\|_2^2 + \lambda \|\Gamma\|_1$$

Fu et al. Interestingness Prediction by Robust Learning to Rank. ECCV 2014

Fu et al. Robust Subjective Visual Property Prediction from Crowdsourced Pairwise Labels. IEEE TPAMI 2016

Sparse Learning in Robust Ranking



Ranking age of images by learning from crowdsourced pairs as the directed graph $G = (V, E)$

Each edge is modeled as $Y_{ij} = \beta^T \phi_i - \beta^T \phi_j + \gamma_{ij}$

The edge of whole dataset is $Y = C\Phi\beta + \epsilon + \Gamma$

$$\min_{\beta,\Gamma} \frac{1}{2} \|Y - C\Phi\beta - \Gamma\|_2^2 + \lambda \|\Gamma\|_1 \longrightarrow \hat{\Gamma} = \arg\min_{\Gamma} \|U_2^T Y - U_2^T \Gamma\| + \lambda \|\Gamma\|_1$$

Fu et al. Interestingness Prediction by Robust Learning to Rank. ECCV 2014

Fu et al. Robust Subjective Visual Property Prediction from Crowdsourced Pairwise Labels. IEEE TPAMI 2016

Checking Regularisation Path

[*2] Fabian L. Wauthier, Nebojsa Jojic and Michael I. Jordan, A Comparative Framework for Preconditioned Lasso Algorithms, NIPS 2013.

Checking Regularisation Path

$$\hat{\Gamma} \quad = \quad \arg\min_{\Gamma} \| U_2^T Y - U_2^T \Gamma \| + \lambda \|\Gamma\|_1$$

[*2] Fabian L. Wauthier, Nebojsa Jojic and Michael I. Jordan, A Comparative Framework for Preconditioned Lasso Algorithms, NIPS 2013.

Checking Regularisation Path

$$\hat{\Gamma} = \arg\min_{\Gamma} \| U_2^T Y - U_2^T \Gamma \| + \lambda \| \Gamma \|_1$$

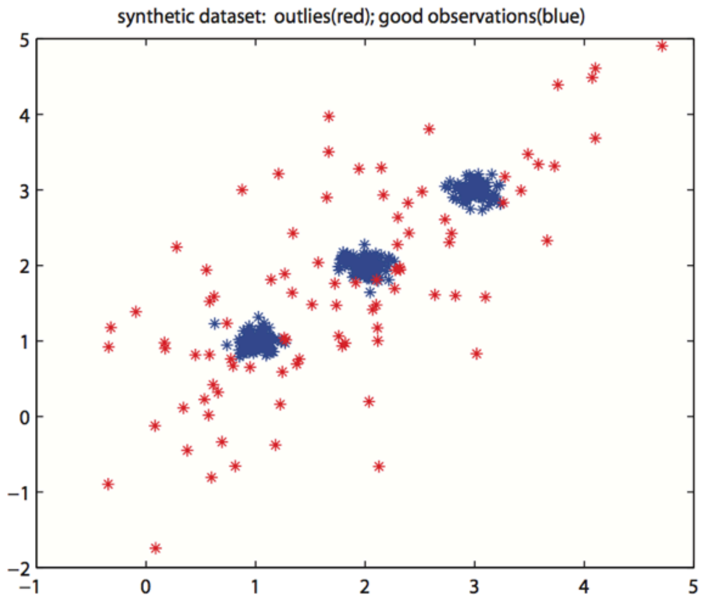
It's one type of Preconditioned Lasso![*2]. We solve Γ checking regularisation path,

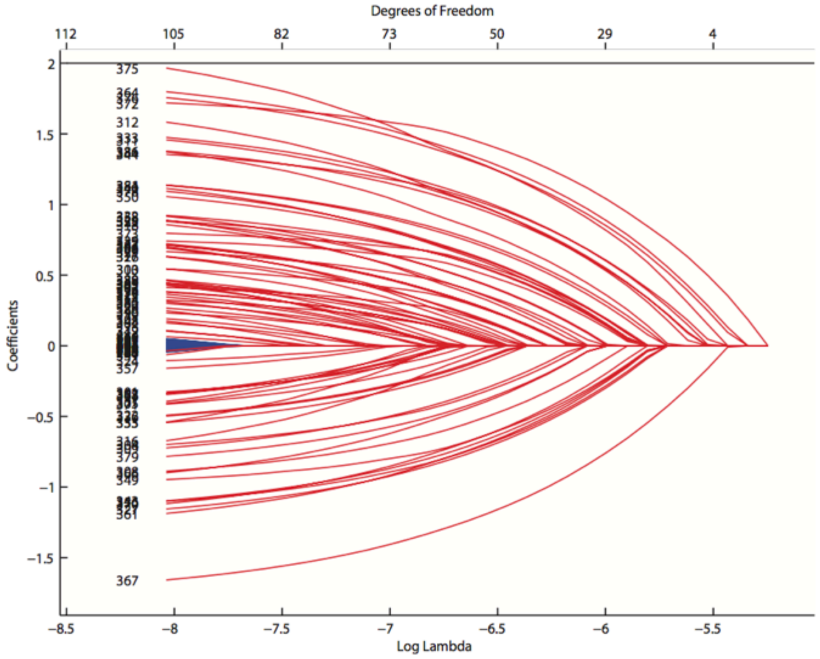
[*2] Fabian L. Wauthier, Nebojsa Jojic and Michael I. Jordan, A Comparative Framework for Preconditioned Lasso Algorithms, NIPS 2013.

Checking Regularisation Path

$$\hat{\Gamma} = \arg\min_{\Gamma} \| U_2^T Y - U_2^T \Gamma \| + \lambda \|\Gamma\|_1$$

It's one type of Preconditioned Lasso![*2]. We solve Γ checking regularisation path,



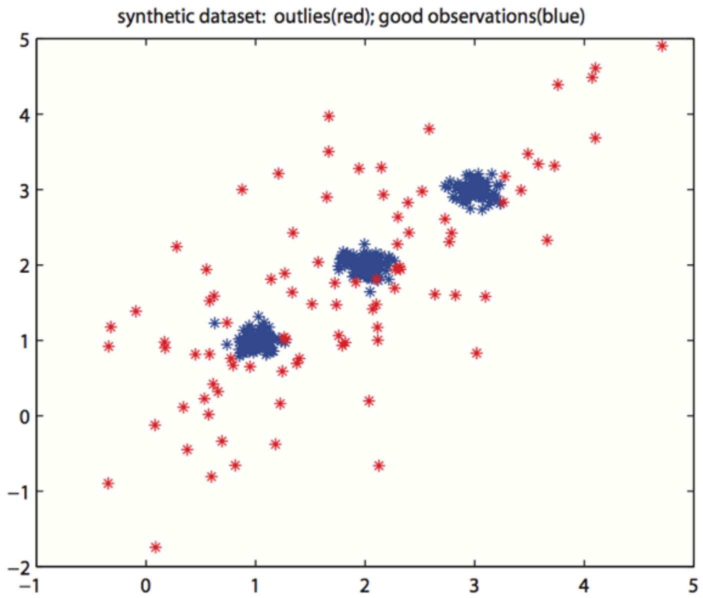


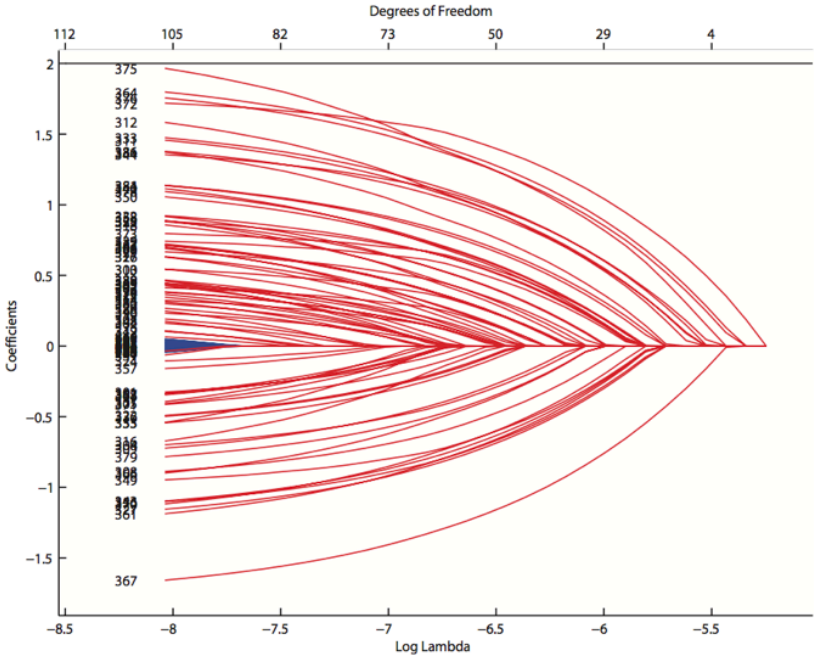
[*2] Fabian L. Wauthier, Nebojsa Jojic and Michael I. Jordan, A Comparative Framework for Preconditioned Lasso Algorithms, NIPS 2013.

Checking Regularisation Path

$$\hat{\Gamma} \quad = \quad \arg\min_{\Gamma} \| U_2^T Y - U_2^T \Gamma \| + \lambda \| \Gamma \|_1$$

It’s one type of Preconditioned Lasso![*2]. We solve Γ checking regularisation path,





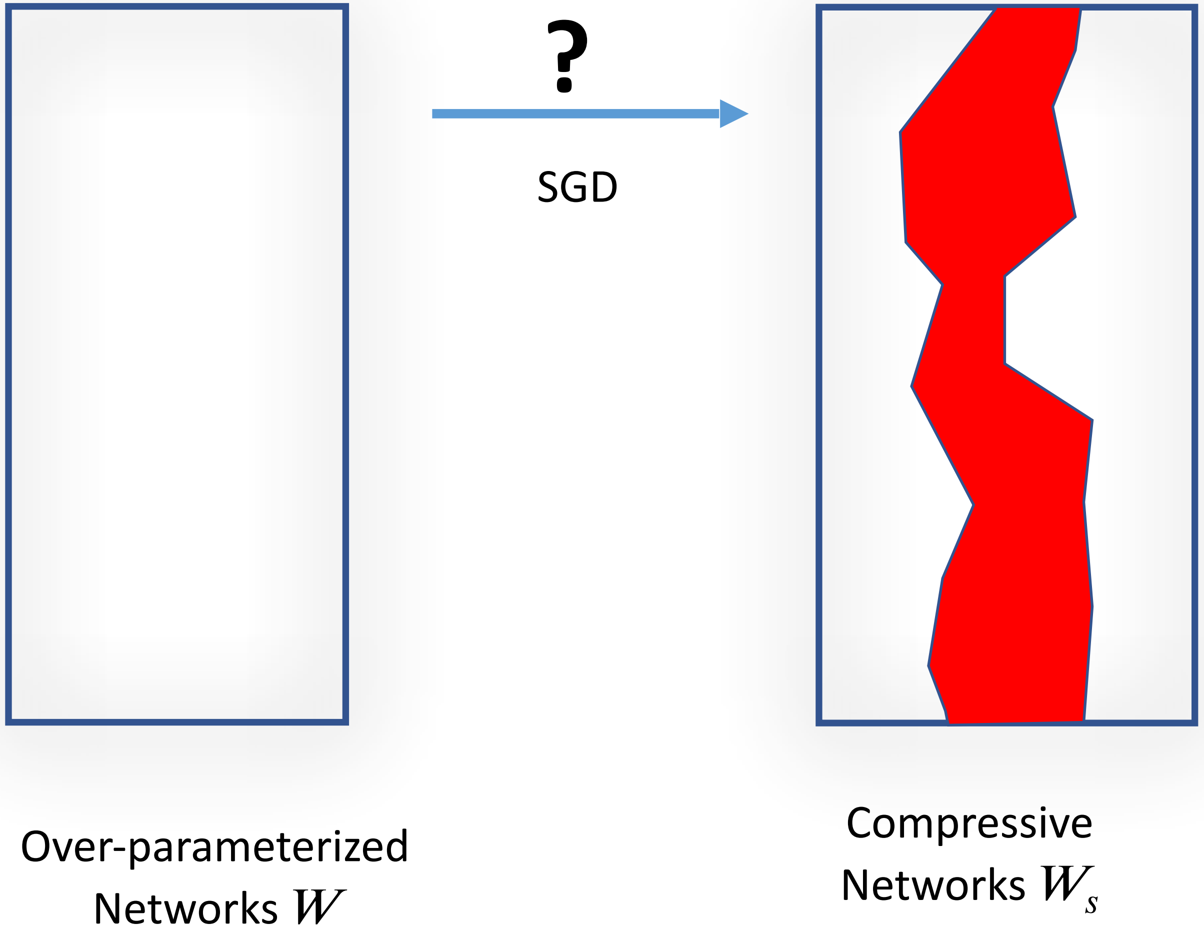
Red lines&points indicate outliers;
Blue lines&points are inliers.

[*2] Fabian L. Wauthier, Nebojsa Jojic and Michael I. Jordan, A Comparative Framework for Preconditioned Lasso Algorithms, NIPS 2013.

Learning Sparsity in Neural Network

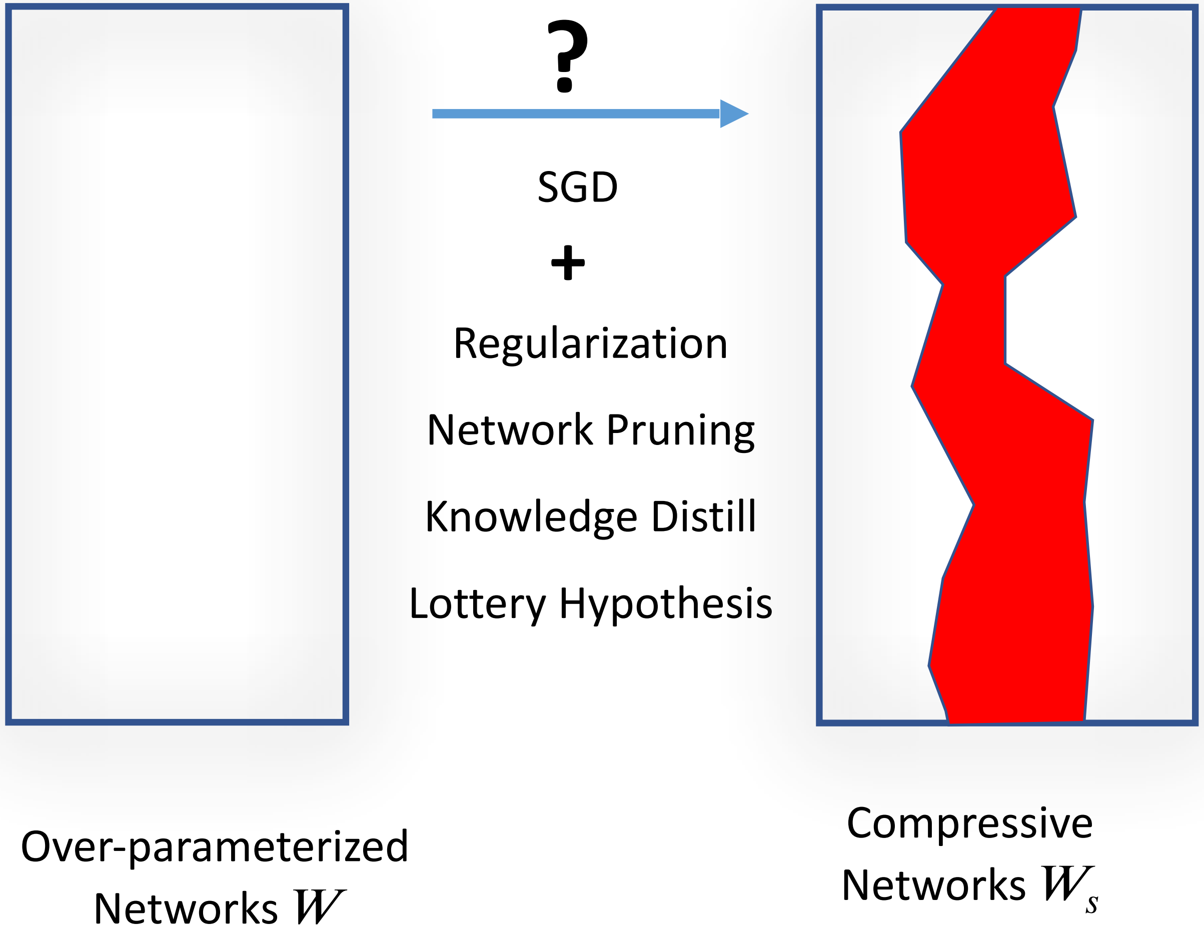
大数据学院
School of Data Science

"OLD SCHOOL" For A Compromise



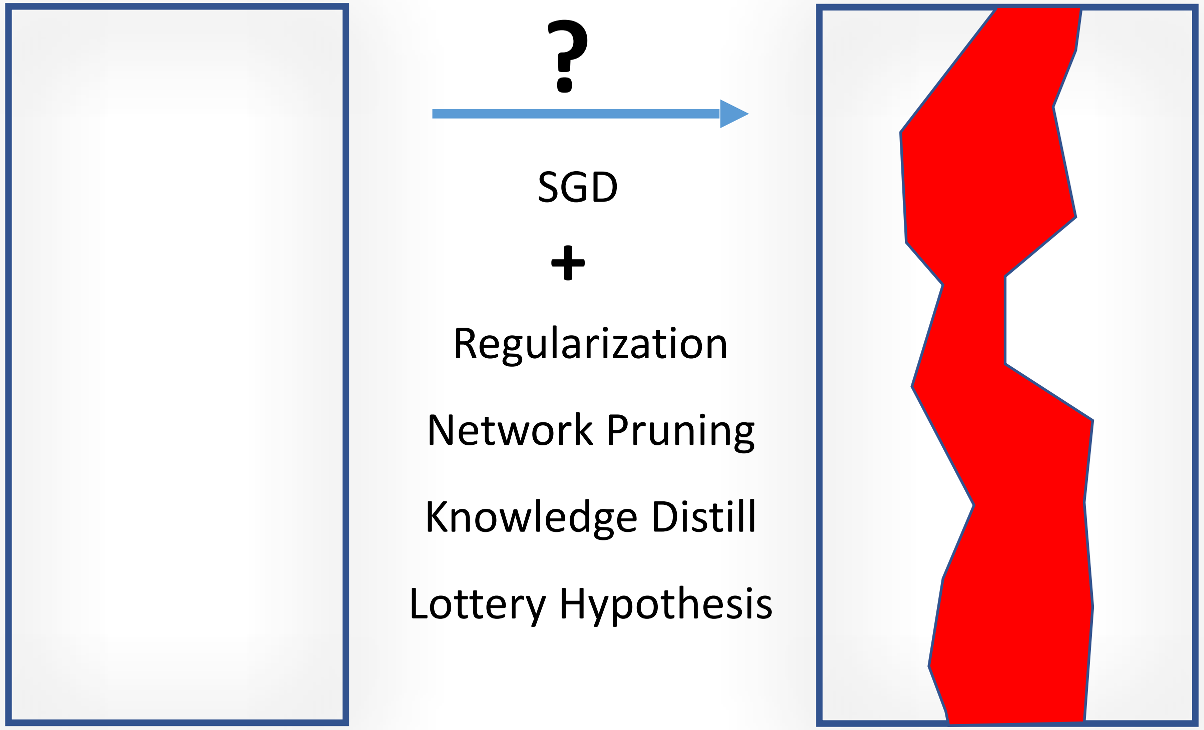
Fu et al. Exploring Structural Sparsity of Deep Networks via Inverse Scale Spaces. IEEE TPAMI accepted (2022)
Fu et al. DessiLBI: Exploring Structural Sparsity on Deep Network via Differential Inclusion Paths. ICML 2020

"OLD SCHOOL" For A Compromise



Fu et al. Exploring Structural Sparsity of Deep Networks via Inverse Scale Spaces. IEEE TPAMI accepted (2022)
Fu et al. DessiLBI: Exploring Structural Sparsity on Deep Network via Differential Inclusion Paths. ICML 2020

“OLD SCHOOL” For A Compromise



Over-parameterized Networks W

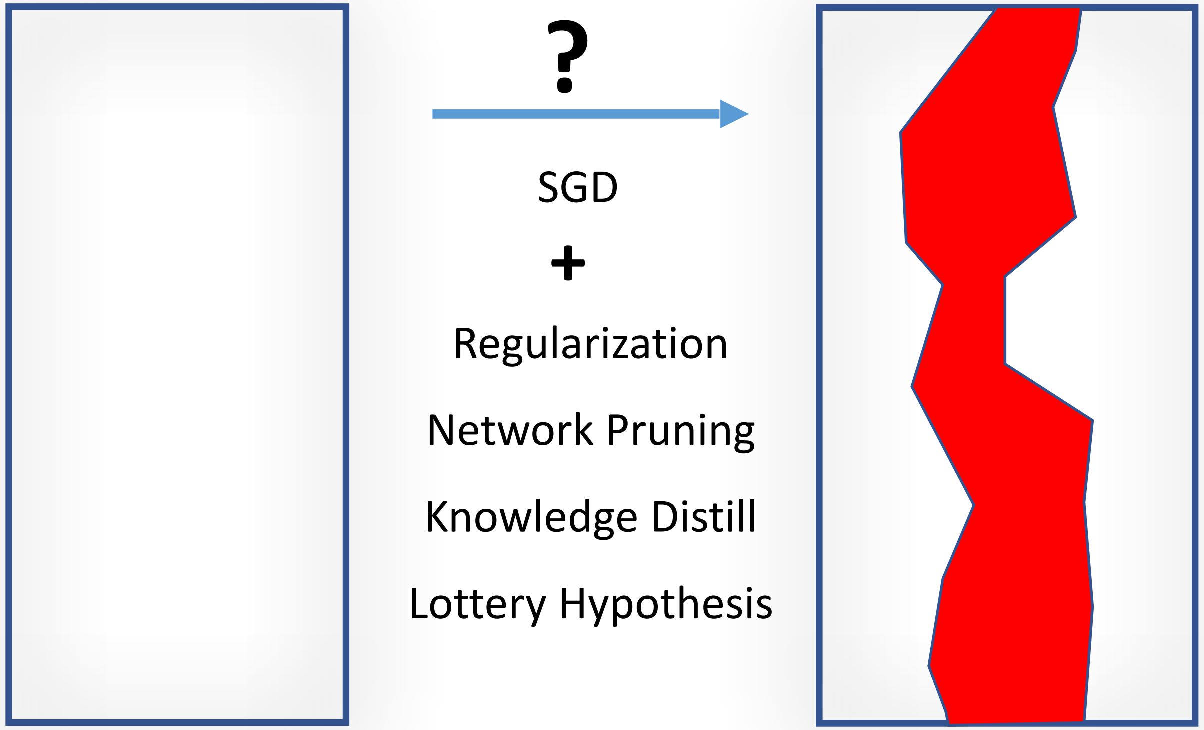
Compressive Networks W_s

Existing 2-Stage Approaches:

- Fully training dense network,

Fu et al. Exploring Structural Sparsity of Deep Networks via Inverse Scale Spaces. IEEE TPAMI accepted (2022)
Fu et al. DessiLBI: Exploring Structural Sparsity on Deep Network via Differential Inclusion Paths. ICML 2020

“OLD SCHOOL” For A Compromise



Over-parameterized Networks W

Compressive Networks W_s

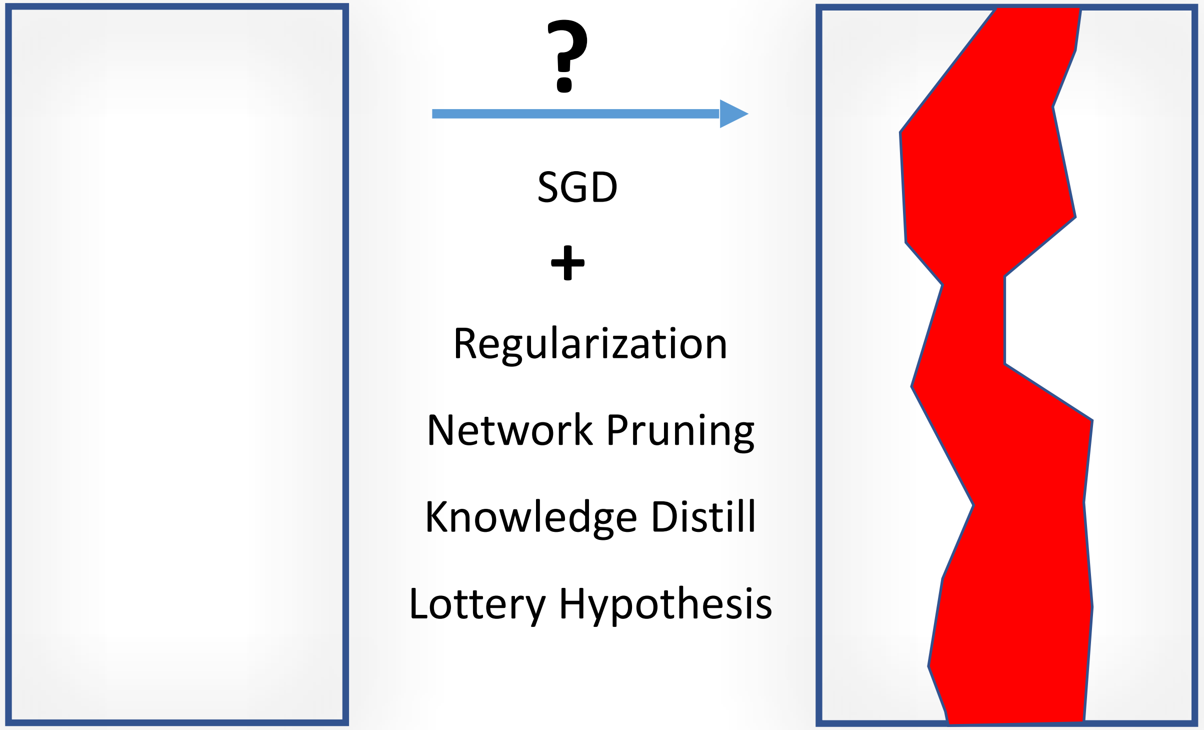
Existing 2-Stage Approaches:

- Fully training dense network,
- Finding good sparse subnets.

Fu et al. Exploring Structural Sparsity of Deep Networks via Inverse Scale Spaces. IEEE TPAMI accepted (2022)

Fu et al. DessiLBI: Exploring Structural Sparsity on Deep Network via Differential Inclusion Paths. ICML 2020

"OLD SCHOOL" For A Compromise



Over-parameterized Networks W

Compressive Networks W_s

Existing 2-Stage Approaches:

- Fully training dense network,
- Finding good sparse subnets.

Our method: 1-Stage Approach (end-to-end)
Without fully training a dense model.

Fu et al. Exploring Structural Sparsity of Deep Networks via Inverse Scale Spaces. IEEE TPAMI accepted (2022)
Fu et al. DessiLBI: Exploring Structural Sparsity on Deep Network via Differential Inclusion Paths. ICML 2020

Regularization and Overparameterization

Regularization and Overparameterization

- Regularization:
  - ℓ_1 regularization [Collins et al. 2014]: Spurious Correlation due to highly correlated neurons

Regularization and Overparameterization

- Regularization:
  - ℓ_1 regularization [Collins et al. 2014]: Spurious Correlation due to highly correlated neurons

Accuracy Loss due to trapping into local minima, without aid of *over-parameterize model* **!**

Regularization and Overparameterization

- Regularization:
  - ℓ_1 regularization [Collins et al. 2014]: Spurious Correlation due to highly correlated neurons

Accuracy Loss due to trapping into local minima, without aid of *over-parameterize model* **!**

Overparameterized Model:

- **Better Optimization** (*Linearization Approximation: find Global Optimal*) (Allen-Zhu, 2019)
- **Better Generalization** (*weight-size dependent complexities are controlled*) (Neyshabur, 2019)

Regularization and Overparameterization

- Regularization:
  - ℓ_1 regularization [Collins et al. 2014]: Spurious Correlation due to highly correlated neurons

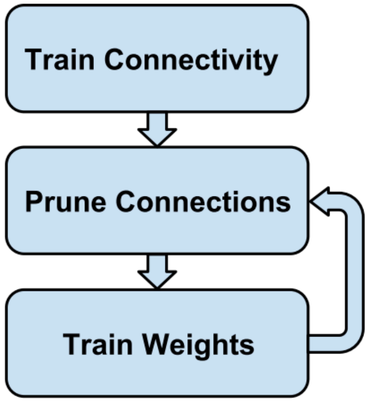
Accuracy Loss due to trapping into local minima, without aid of *over-parameterize model* **!**

Overparameterized Model:

- **Better Optimization** (*Linearization Approximation: find Global Optimal*) (Allen-Zhu, 2019)
- **Better Generalization** (*weight-size dependent complexities are controlled*) (Neyshabur, 2019)

However, Larger Inference Time and Memory Cost!

Network Pruning (1)



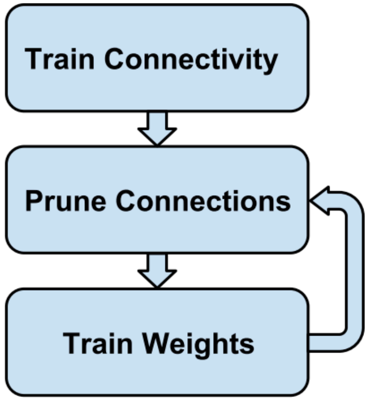
Three-Step Training Pipeline

Network Pruning:

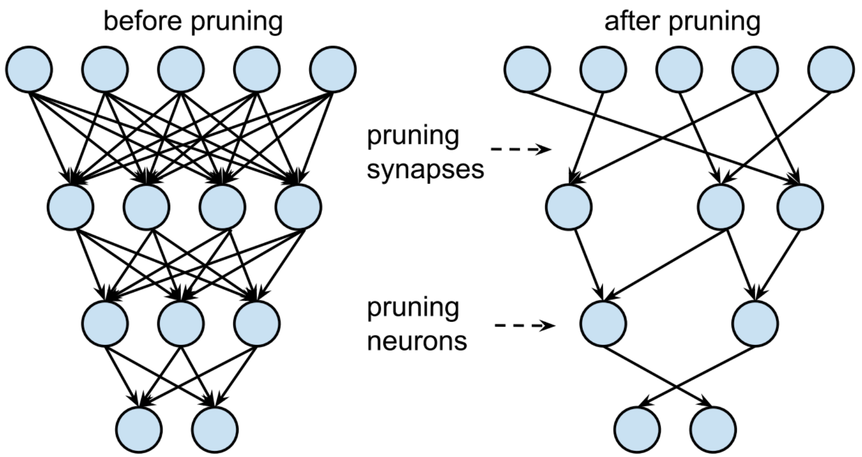
- Weight Pruning
- Filter Pruning

Han et al. Learning both Weights and Connections for Efficient Neural Networks. NeurPIS 2015
Han et al. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding. ICLR16 (Best Paper Award)
Song Han's talk: Hardware Efficiency Aware Neural Architecture Search. The 3rd Workshop on Energy Efficient Machine Learning and Cognitive Computing in CVPR 2019

Network Pruning (1)



Three-Step Training Pipeline



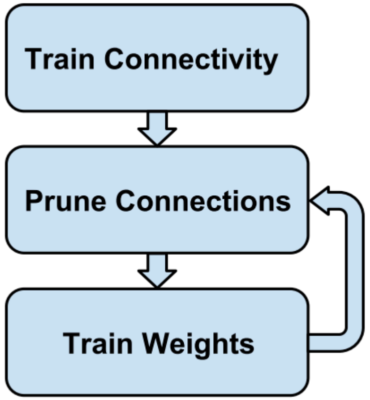
Network Pruning:

- Weight Pruning
- Filter Pruning

Pruning
[Han, NeurPIS15]

Han et al. Learning both Weights and Connections for Efficient Neural Networks. NeurPIS 2015
Han et al. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding. ICLR16 (Best Paper Award)
Song Han's talk: Hardware Efficiency Aware Neural Architecture Search. The 3rd Workshop on Energy Efficient Machine Learning and Cognitive Computing in CVPR 2019

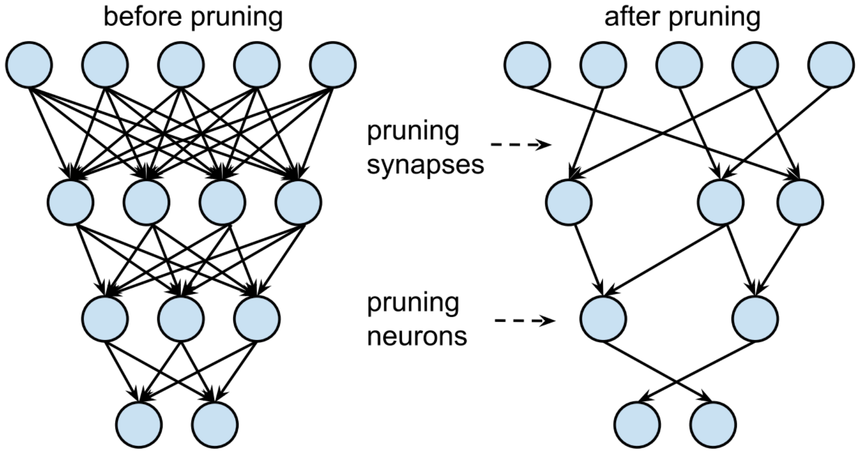
Network Pruning (1)



Three-Step Training Pipeline

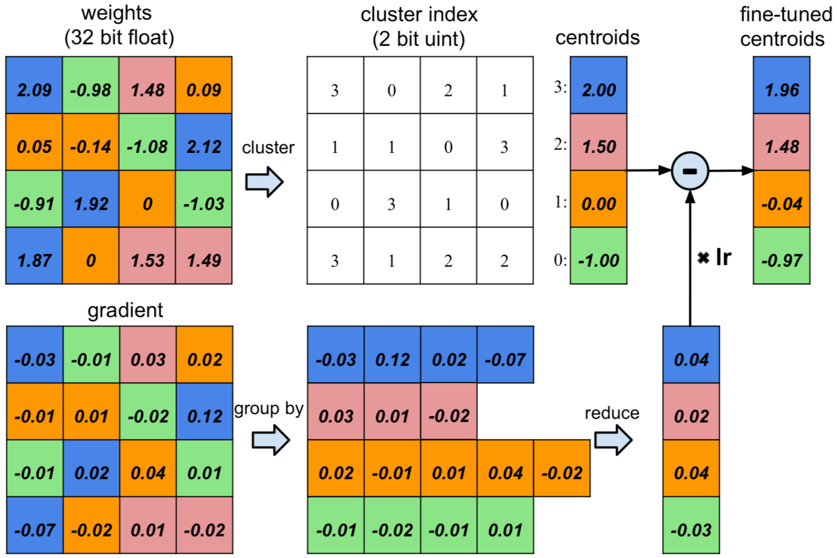
Network Pruning:

- Weight Pruning
- Filter Pruning



Pruning
[Han, NeurPIS15]

Hardware Acceleration

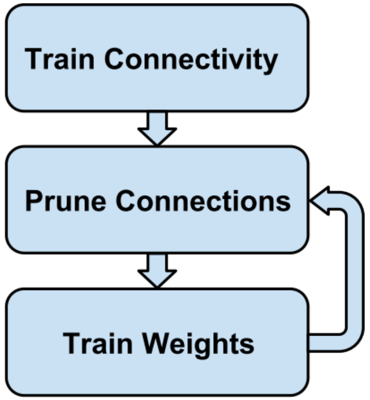


Quantization [Han, ICLR16]

Han et al. Learning both Weights and Connections for Efficient Neural Networks. NeurPIS 2015
Han et al. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding. ICLR16 (Best Paper Award)
Song Han’s talk: Hardware Efficiency Aware Neural Architecture Search. The 3rd Workshop on Energy Efficient Machine Learning and Cognitive Computing in CVPR 2019

大数据学院 School of Data Science

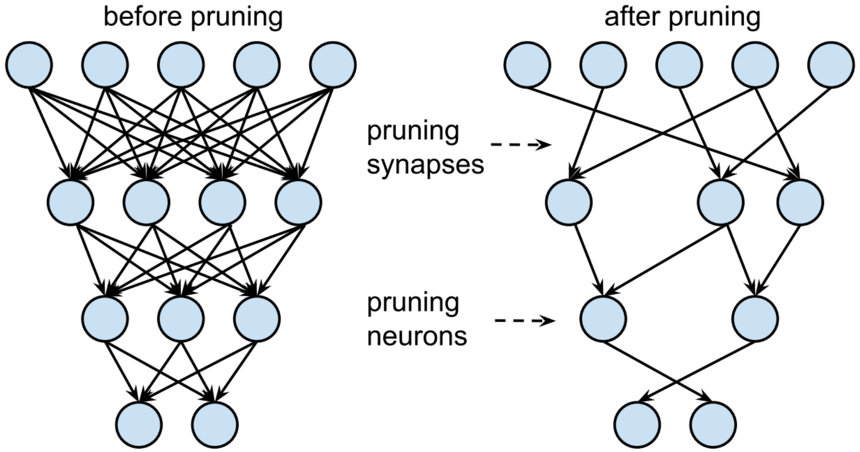
Network Pruning (1)



Three-Step Training Pipeline

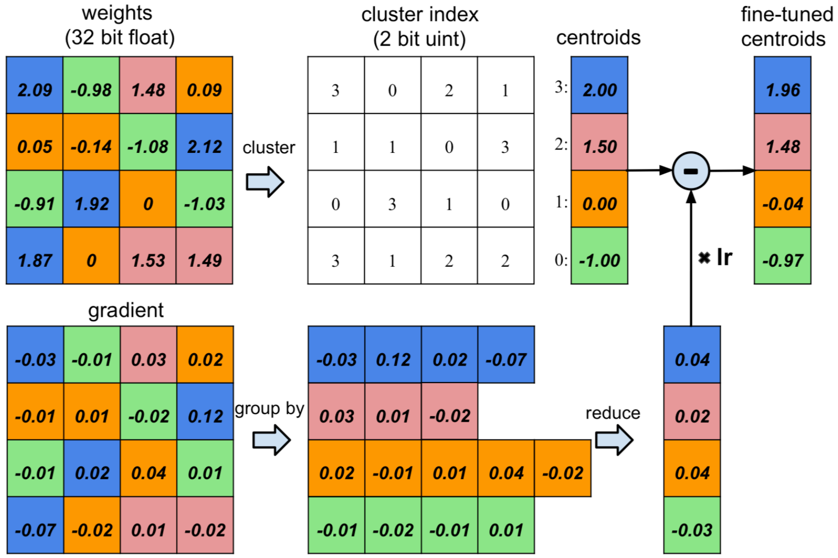
Network Pruning:

- Weight Pruning
- Filter Pruning



Pruning
[Han, NeurPIS15]

Hardware Acceleration



Quantization [Han, ICLR16]

Questions: Can we do sparsity in weight level, filter level, and even layer level with a unified 'algorithm'?

Han et al. Learning both Weights and Connections for Efficient Neural Networks. NeurPIS 2015

Han et al. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding. ICLR16 (Best Paper Award)

Song Han's talk: Hardware Efficiency Aware Neural Architecture Search. The 3rd Workshop on Energy Efficient Machine Learning and Cognitive Computing in CVPR 2019

Learning Sparsity on Models

Child et al. Generating Long Sequences with Sparse Transformers. arxiv 2019
Dettmers, et al. Sparse Networks from Scratch:Faster Training without Losing Performance. Arxiv 2019

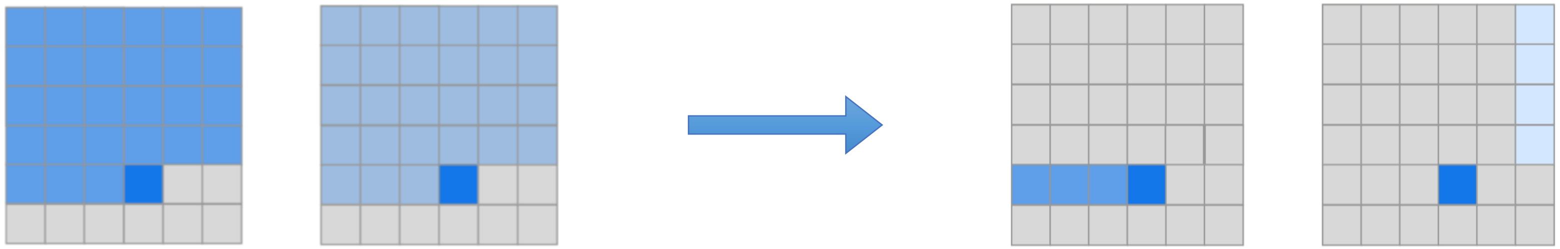
Learning Sparsity on Models

- Designing sparse transformer architectures (Child, 2019)

Child et al. Generating Long Sequences with Sparse Transformers. arxiv 2019
Dettmers, et al. Sparse Networks from Scratch:Faster Training without Losing Performance. Arxiv 2019

Learning Sparsity on Models

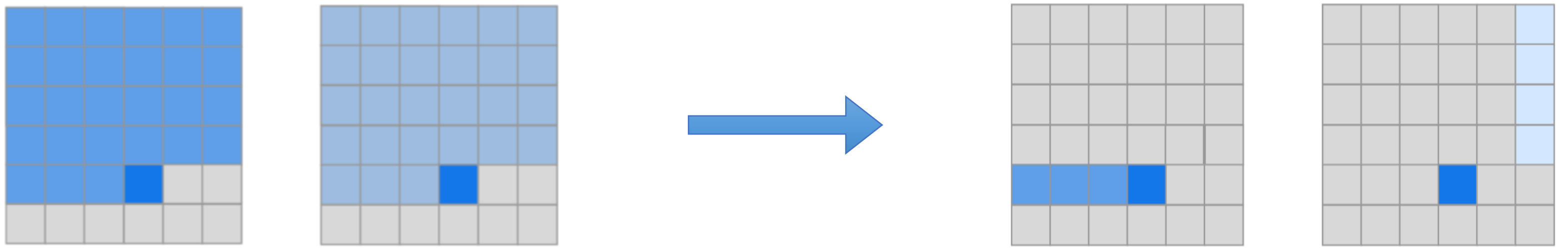
- Designing sparse transformer architectures (Child, 2019)



Child et al. Generating Long Sequences with Sparse Transformers. arxiv 2019
Dettmers, et al. Sparse Networks from Scratch:Faster Training without Losing Performance. Arxiv 2019

Learning Sparsity on Models

- Designing sparse transformer architectures (Child, 2019)

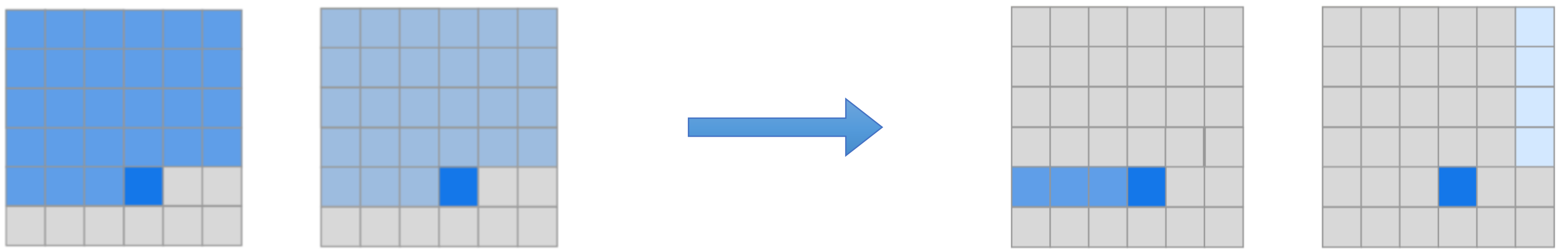


- Pruning over-parameterized structure with regrowing (Dettmers, 2019)

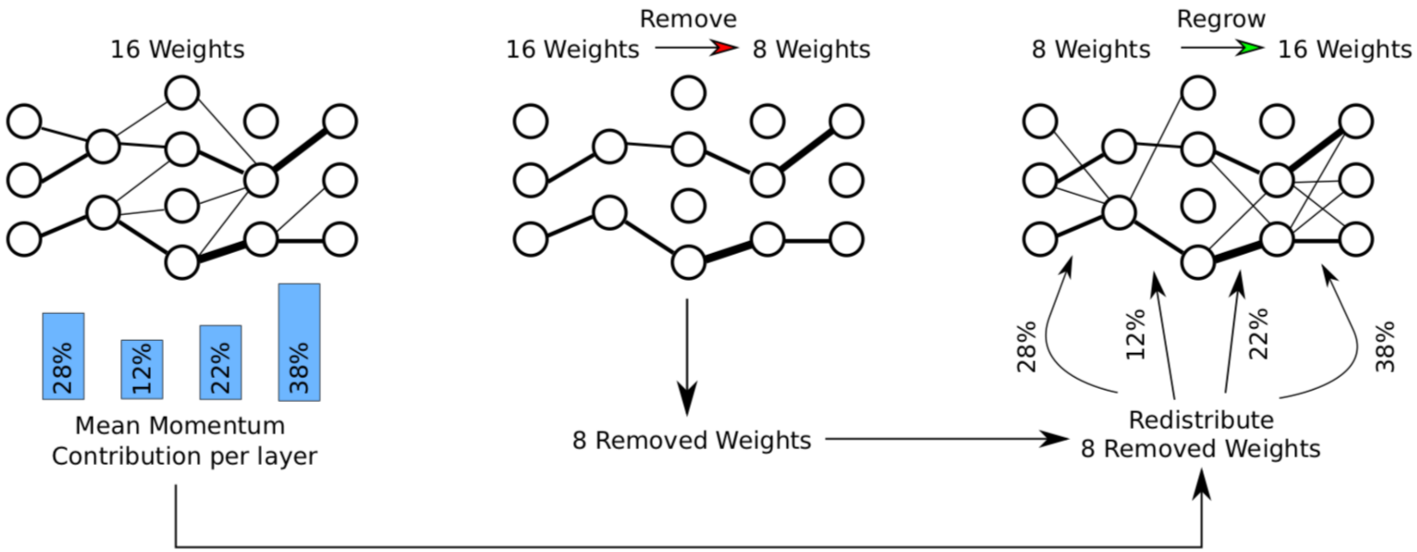
Child et al. Generating Long Sequences with Sparse Transformers. arxiv 2019
Dettmers, et al. Sparse Networks from Scratch:Faster Training without Losing Performance. Arxiv 2019

Learning Sparsity on Models

- Designing sparse transformer architectures (Child, 2019)



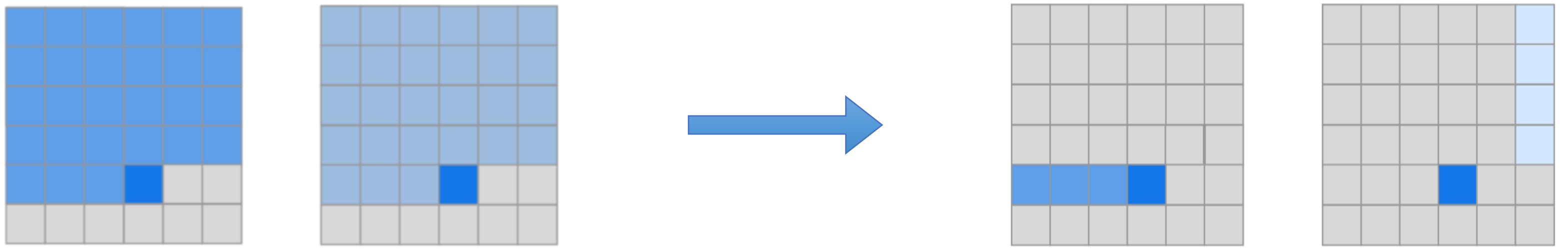
- Pruning over-parameterized structure with regrowing (Dettmers, 2019)



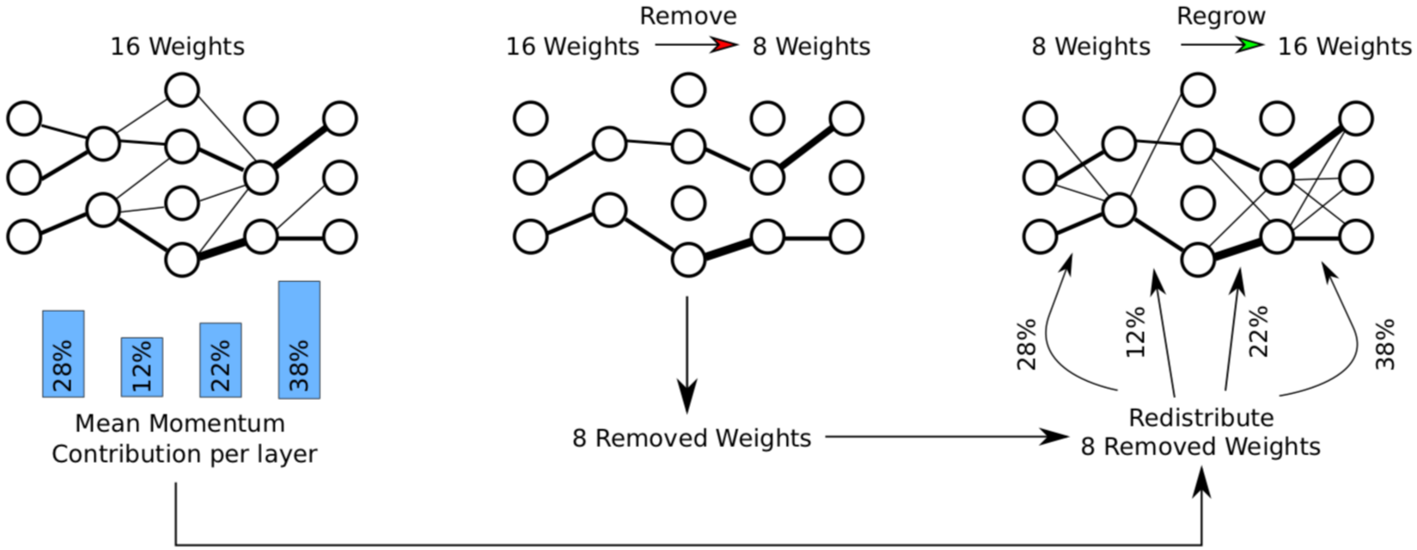
Child et al. Generating Long Sequences with Sparse Transformers. arxiv 2019
Dettmers, et al. Sparse Networks from Scratch:Faster Training without Losing Performance. Arxiv 2019

Learning Sparsity on Models

- Designing sparse transformer architectures (Child, 2019)



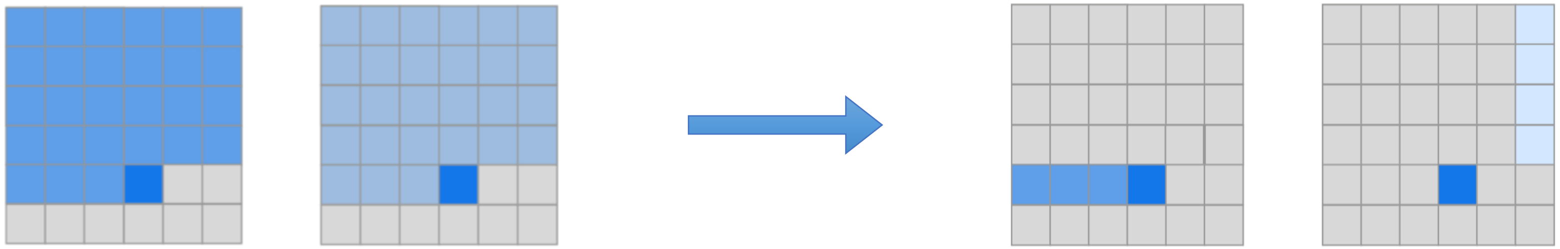
- Pruning over-parameterized structure with regrowing (Dettmers, 2019)



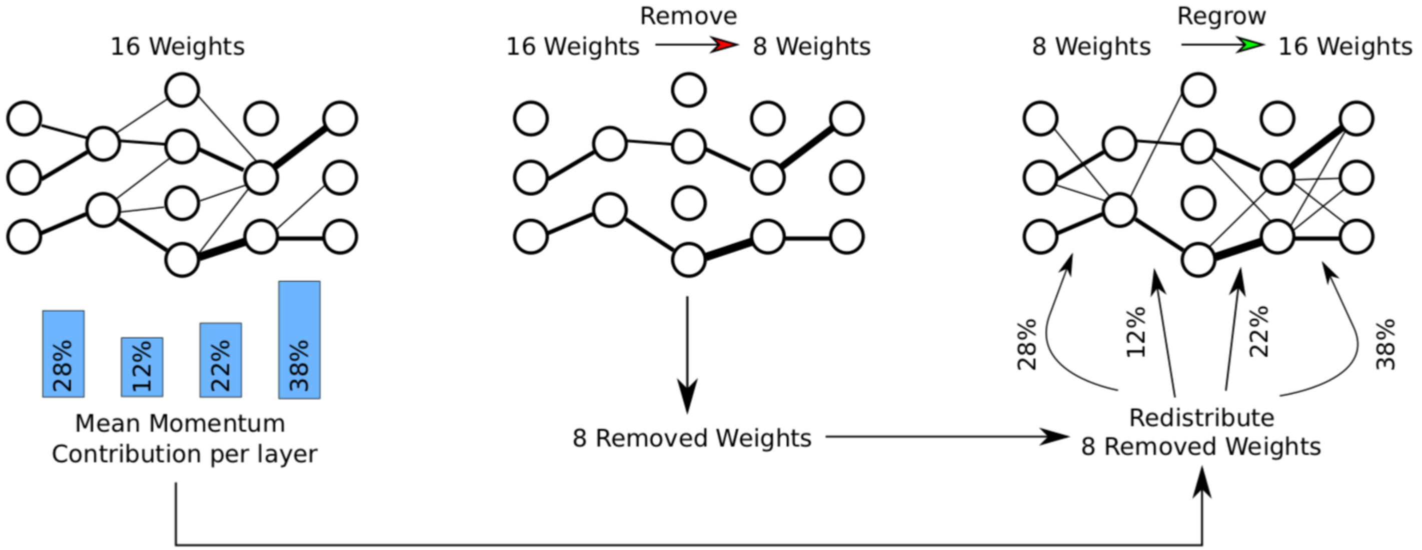
Child et al. Generating Long Sequences with Sparse Transformers. arxiv 2019
Dettmers, et al. Sparse Networks from Scratch:Faster Training without Losing Performance. Arxiv 2019

Learning Sparsity on Models

- Designing sparse transformer architectures (Child, 2019)



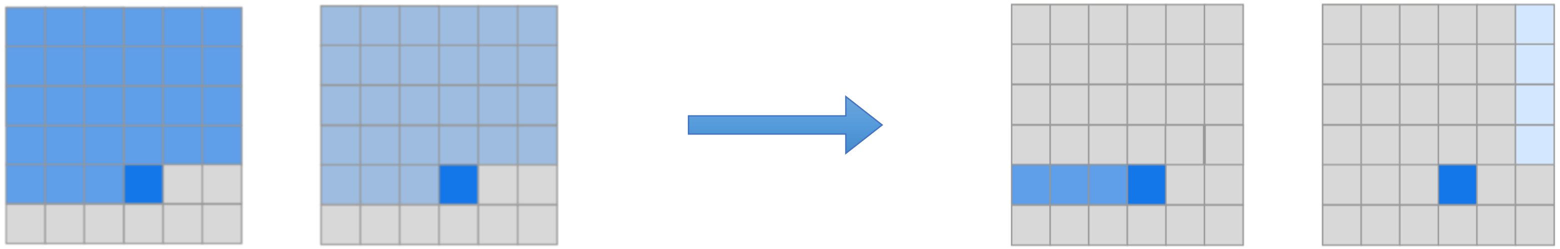
- Pruning over-parameterized structure with regrowing (Dettmers, 2019)



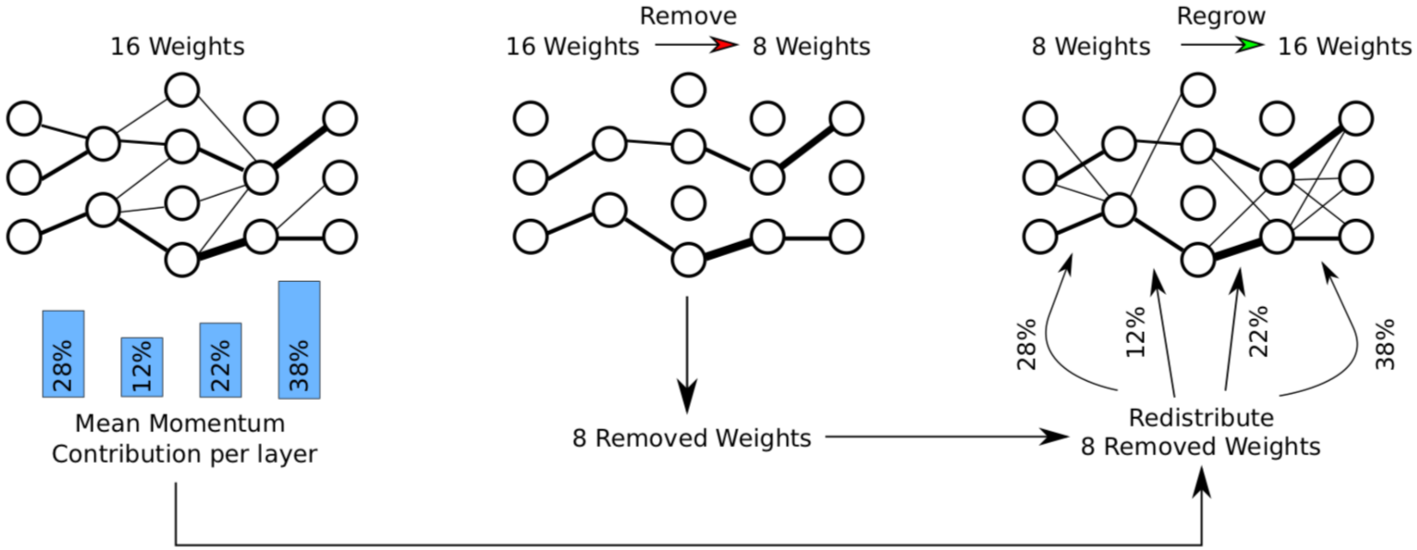
Child et al. Generating Long Sequences with Sparse Transformers. arxiv 2019
Dettmers, et al. Sparse Networks from Scratch:Faster Training without Losing Performance. Arxiv 2019

Learning Sparsity on Models

- Designing sparse transformer architectures (Child, 2019)

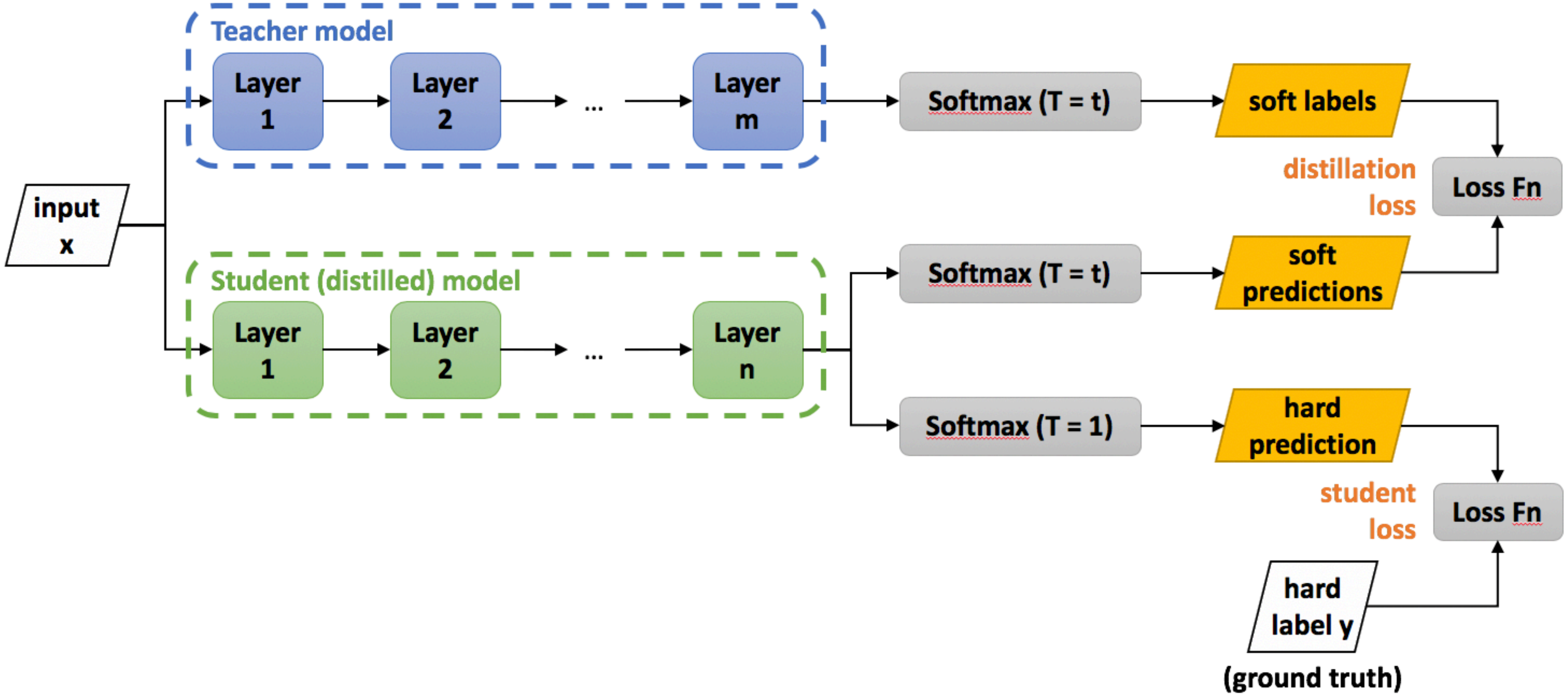


- Pruning over-parameterized structure with regrowing (Dettmers, 2019)



Child et al. Generating Long Sequences with Sparse Transformers. arxiv 2019
Dettmers, et al. Sparse Networks from Scratch:Faster Training without Losing Performance. Arxiv 2019

The Concept of Knowledge Distill

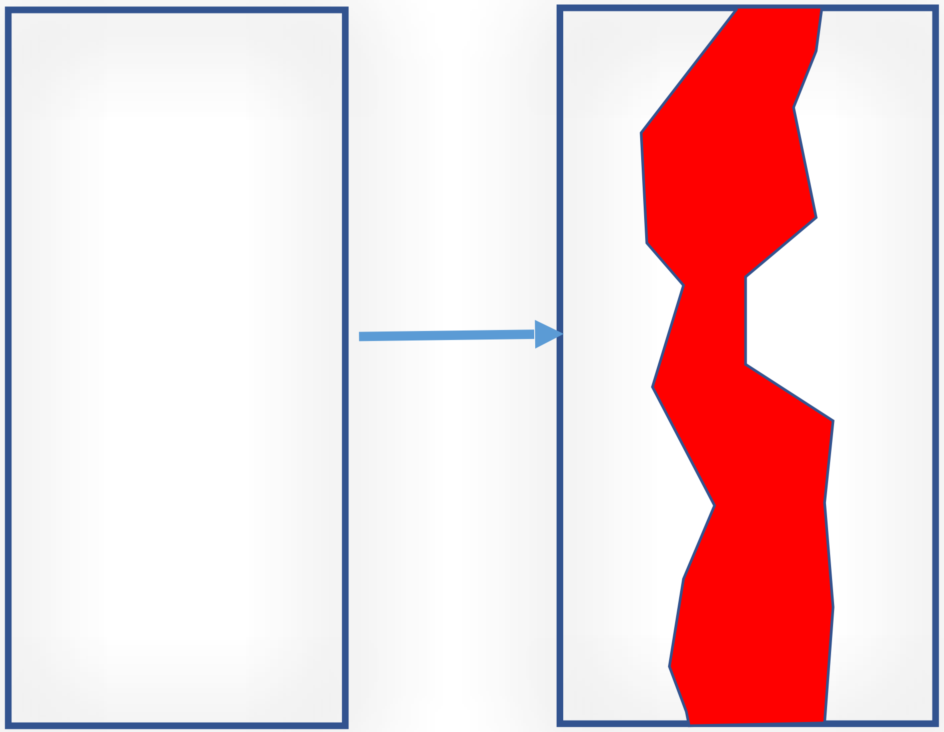


- Introducing the concept of "softmax temperature".
- Utilizing the “dark knowledge” of teacher model: which classes is more similar to the predicted class.

Hinton et al. Distilling the Knowledge in a Neural Network. NIPS 2014 Deep Learning Workshop

Figure from https://intellabs.github.io/distiller/knowledge_distillation.html

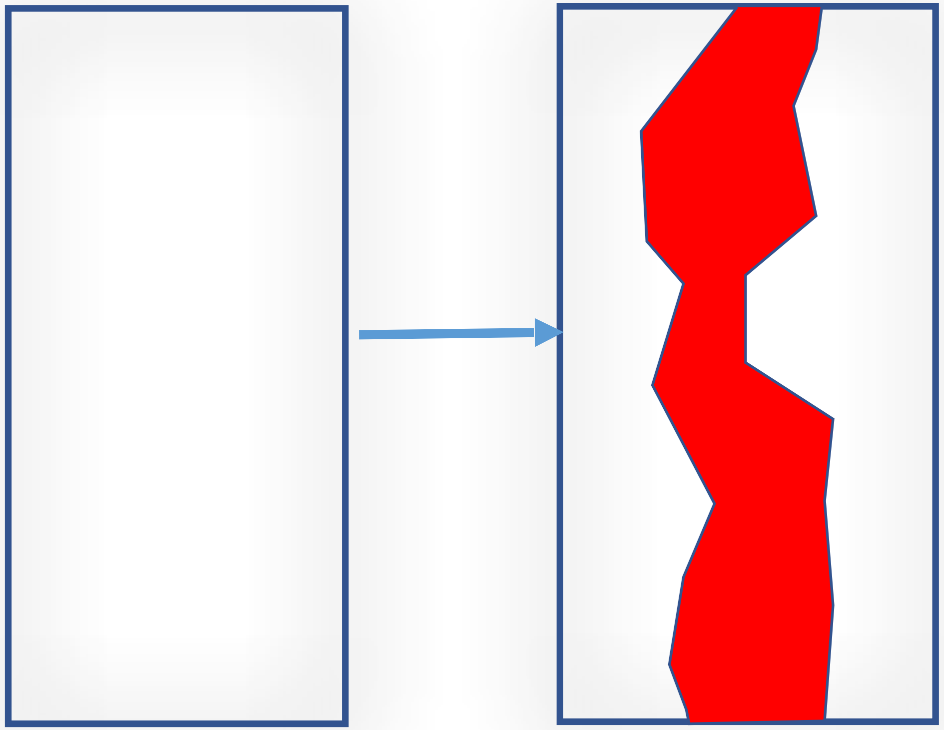
Lottery Hypothesis



Over-parameterized Networks W

Compressive Networks W_s

Lottery Hypothesis



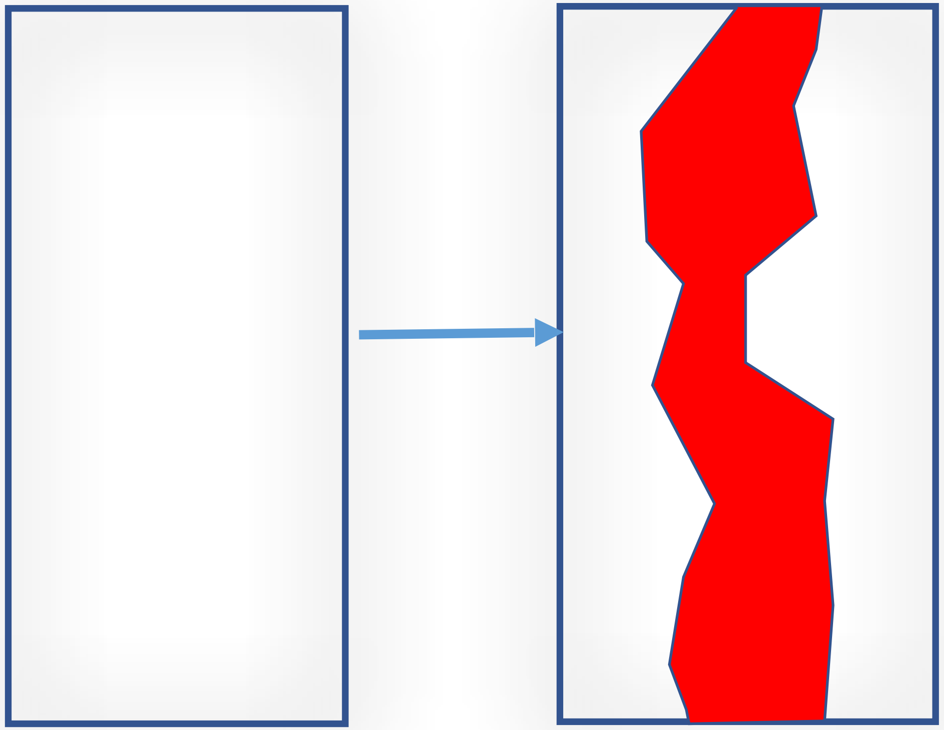
Over-parameterized Networks W

Compressive Networks W_s

Lottery Ticket Hypothesis

- *Dense, randomly-initialized, feed-forward networks contain subnetworks (winning tickets) that – when trained in isolation – reach test accuracy comparable to the original network in a similar number of iterations.* (Frankle & Carbin, 2019)

Lottery Hypothesis



Over-parameterized Networks W

Compressive Networks W_s

Lottery Ticket Hypothesis

- *Dense, randomly-initialized, feed-forward networks contain subnetworks (winning tickets) that – when trained in isolation – reach test accuracy comparable to the original network in a similar number of iterations.* (Frankle & Carbin, 2019)

Rewinding the network from the initialization, and find “winning ticket” subnet

The Key Idea of the Optimizer-DessiLBI

DessiLBI : **De**ep structurally **s**pl**i**tting **L**inearized **B**regman **I**teration

Over-parameterized Network in Space W

1, Lifting parameter space W to (W, Γ) coupling the inverse scale space.

2, Γ learns structural sparsity in inverse scale space.

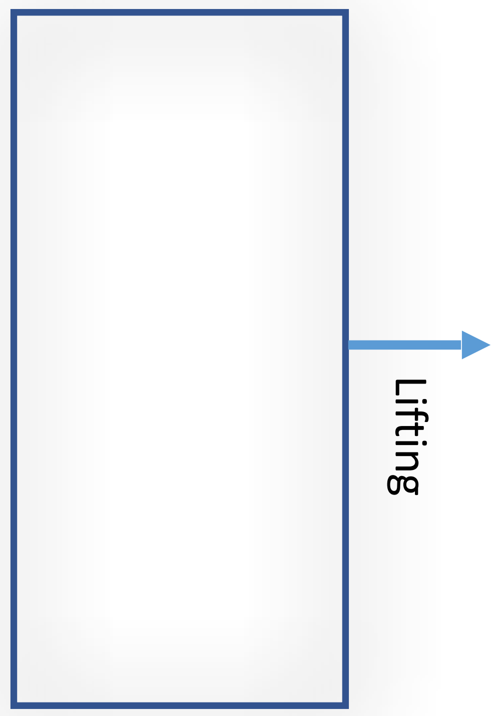
3, Network's solution path in (W, Γ) as the discretization of dynamics, and solved by LBI.

4, Our optimizer enjoys a provable global convergence guarantee.

We will give the mathematical and statistics introduction of **L**inearized **B**regman **I**teration (LBI) in the next talks.

The Key Idea of the Optimizer-DessiLBI

DessiLBI : **De**ep **s**tructurally **s**pl**i**tting **L**inearized **B**regman **I**teration



Over-parameterized
Network in Space W

1, Lifting parameter space W to (W, Γ) coupling the inverse scale space.

2, Γ learns structural sparsity in inverse scale space.

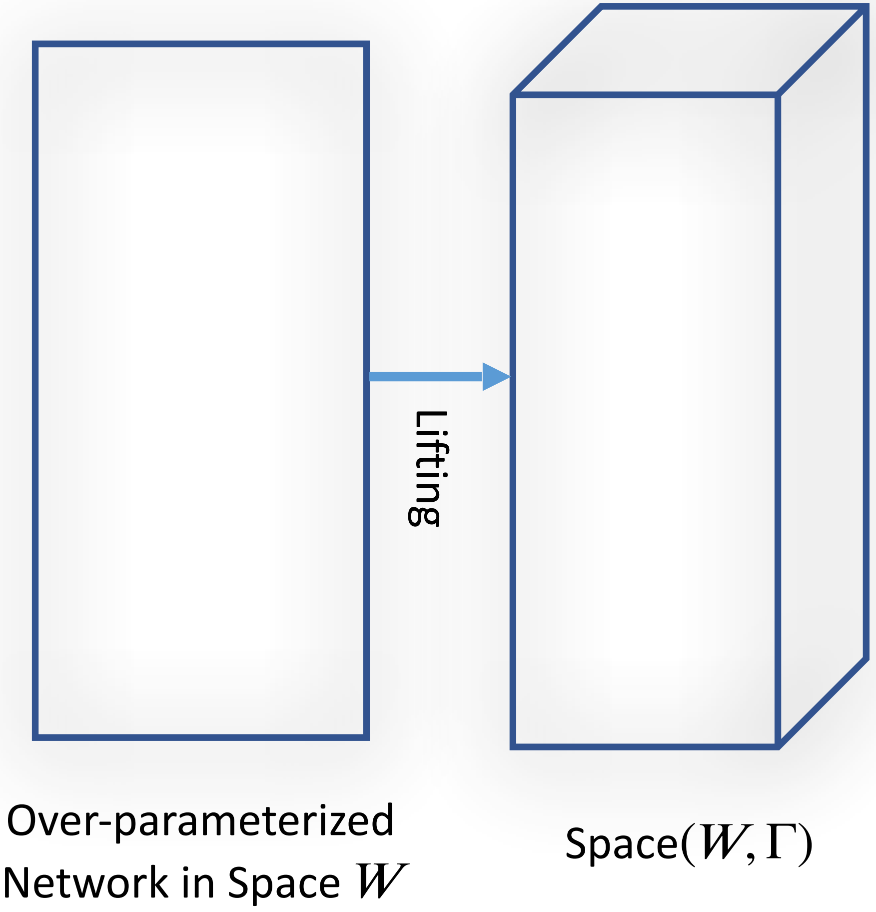
3, Network’s solution path in (W, Γ) as the discretization of dynamics, and solved by LBI.

4, Our optimizer enjoys a provable global convergence guarantee.

We will give the mathematical and statistics introduction of **L**inearized **B**regman **I**teration (LBI) in the next talks.

The Key Idea of the Optimizer-DessiLBI

DessiLBI : **De**ep **s**tructurally **s**pl**i**tting **L**inearized **B**regman **I**teration



1, Lifting parameter space W to (W, Γ) coupling the inverse scale space.

2, Γ learns structural sparsity in inverse scale space.

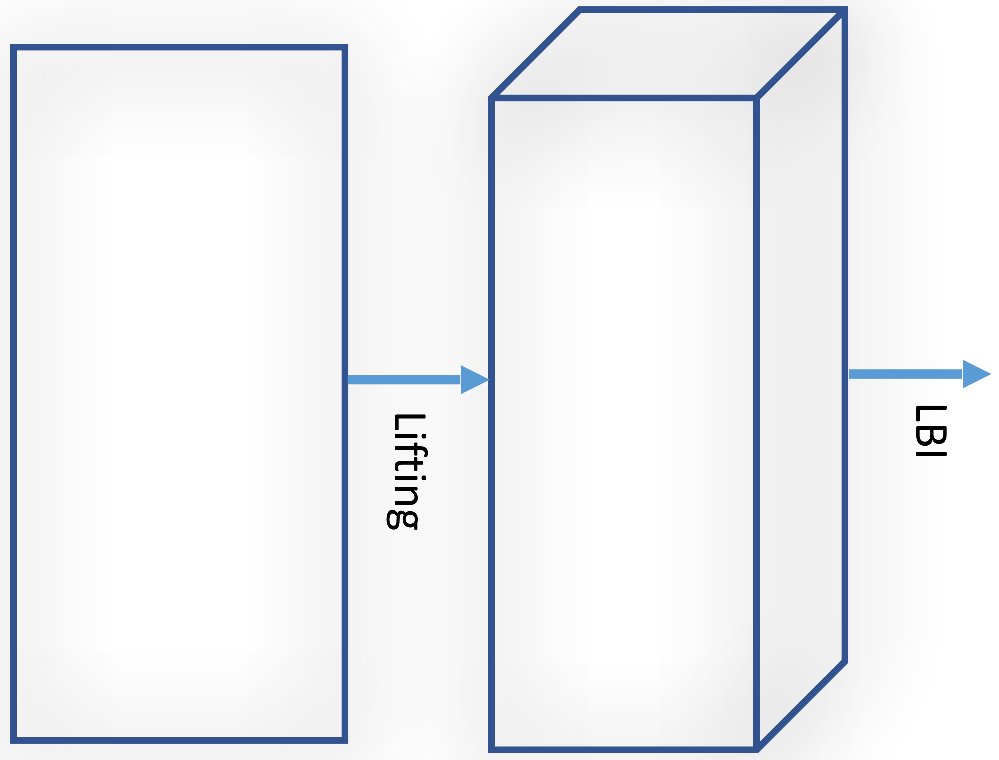
3, Network's solution path in (W, Γ) as the discretization of dynamics, and solved by LBI.

4, Our optimizer enjoys a provable global convergence guarantee.

We will give the mathematical and statistics introduction of **L**inearized **B**regman **I**teration (LBI) in the next talks.

The Key Idea of the Optimizer-DessiLBI

DessiLBI : **De**ep **s**tructurally **s**pl**i**tting **L**inearized **B**regman **I**teration



Over-parameterized Network in Space W

Space(W, Γ)

1, Lifting parameter space W to (W, Γ) coupling the inverse scale space.

2, Γ learns structural sparsity in inverse scale space.

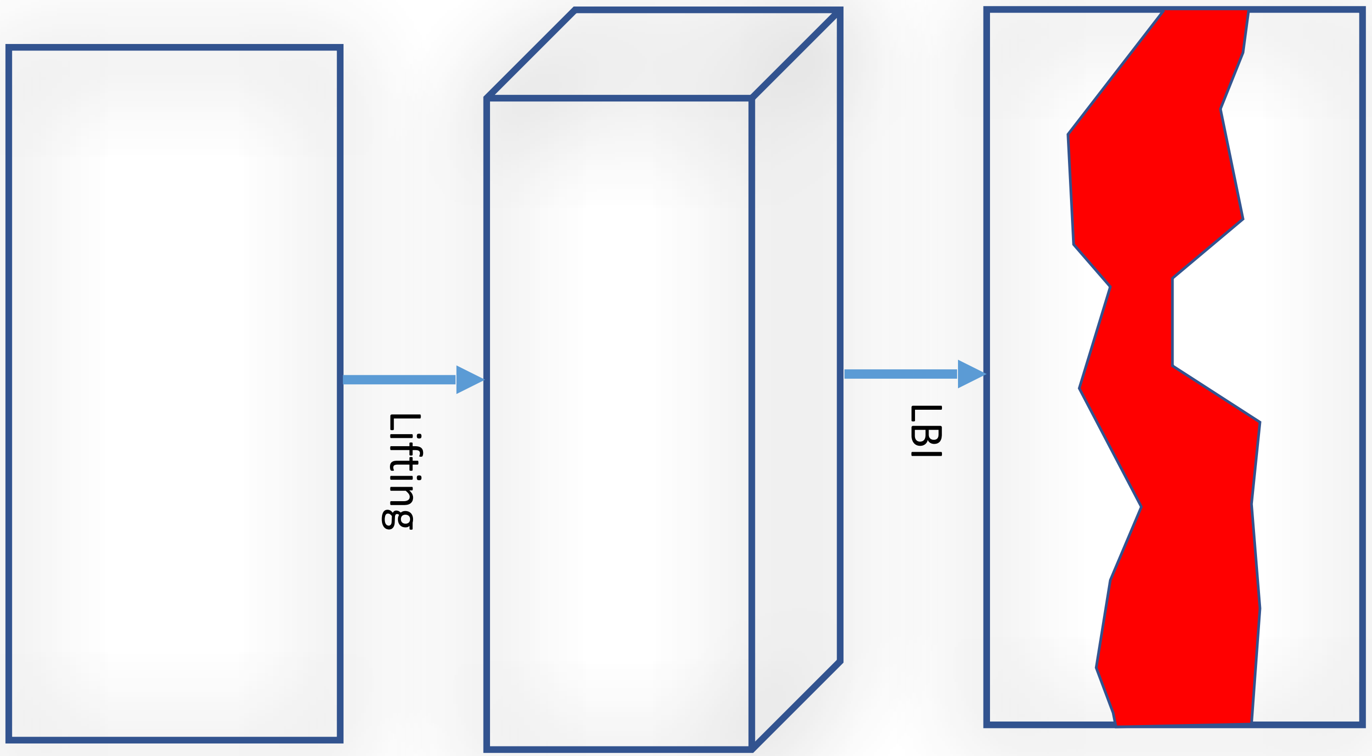
3, Network's solution path in (W, Γ) as the discretization of dynamics, and solved by LBI.

4, Our optimizer enjoys a provable global convergence guarantee.

We will give the mathematical and statistics introduction of **L**inearized **B**regman **I**teration (LBI) in the next talks.

The Key Idea of the Optimizer-DessiLBI

DessiLBI : **De**ep **s**tructurally **s**pl**i**tting **L**inearized **B**regman **I**teration



Over-parameterized Network in Space W

Space (W, Γ)

Compressive Networks W_s

1, Lifting parameter space W to (W, Γ) coupling the inverse scale space.

2, Γ learns structural sparsity in inverse scale space.

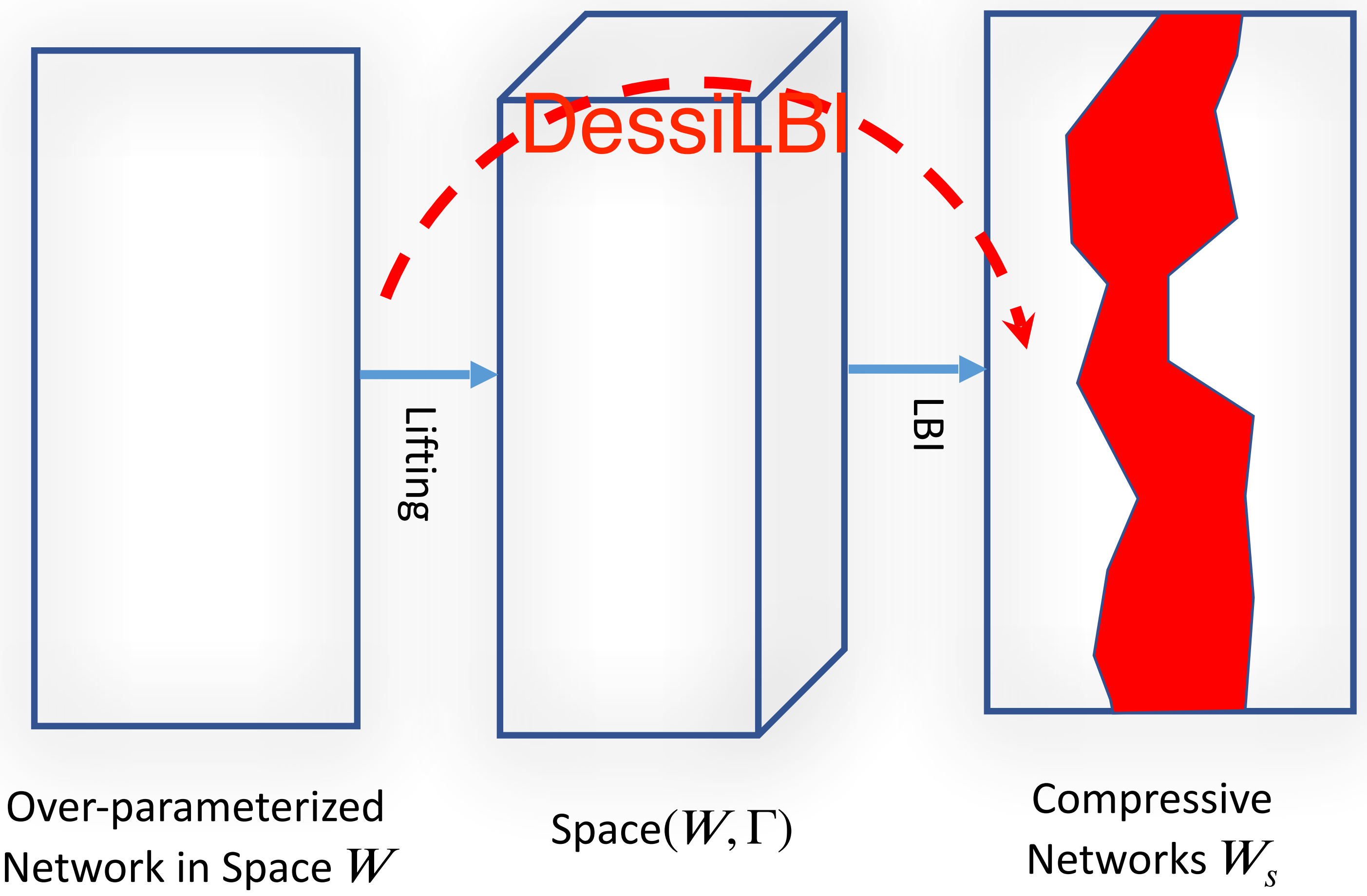
3, Network's solution path in (W, Γ) as the discretization of dynamics, and solved by LBI.

4, Our optimizer enjoys a provable global convergence guarantee.

We will give the mathematical and statistics introduction of **L**inearized **B**regman **I**teration (LBI) in the next talks.

The Key Idea of the Optimizer-DessiLBI

DessiLBI : **De**ep **s**tructurally **s**pl**i**tting **L**inearized **B**regman **I**teration



1, Lifting parameter space W to (W,Γ) coupling the inverse scale space.

2, Γ learns structural sparsity in inverse scale space.

3, Network's solution path in (W,Γ) as the discretization of dynamics, and solved by LBI.

4, Our optimizer enjoys a provable global convergence guarantee.

We will give the mathematical and statistics introduction of **L**inearized **B**regman **I**teration (LBI) in the next talks.

Any Better Solutions?

Two Stage Method, Training Dense Network → Produce sparse subnet

One Stage Method, An end-to-end method, directly producing structurally sparse Subnet.

Charles et al. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. CVPR 2017

Any Better Solutions?

Two Stage Method, Training Dense Network → Produce sparse subnet

One Stage Method, An end-to-end method, directly producing structurally sparse Subnet.

Inspired by

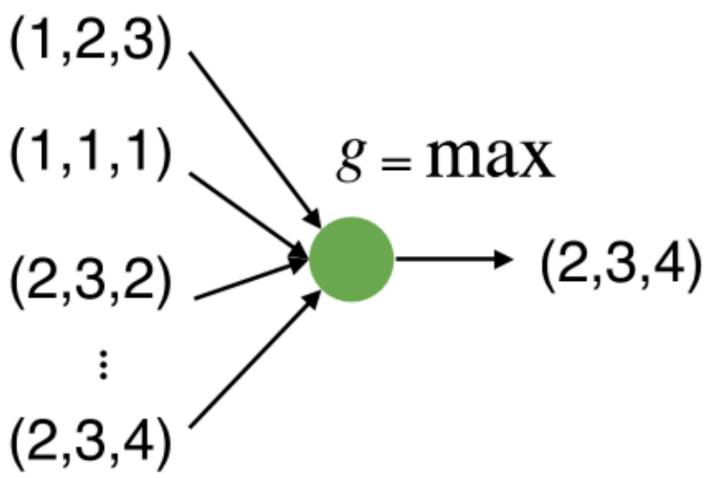
PointNet

Charles et al. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. CVPR 2017

Any Better Solutions?

Two Stage Method, Training Dense Network → Produce sparse subnet

One Stage Method, An end-to-end method, directly producing structurally sparse Subnet.



Discover naïve/extreme property of geometry

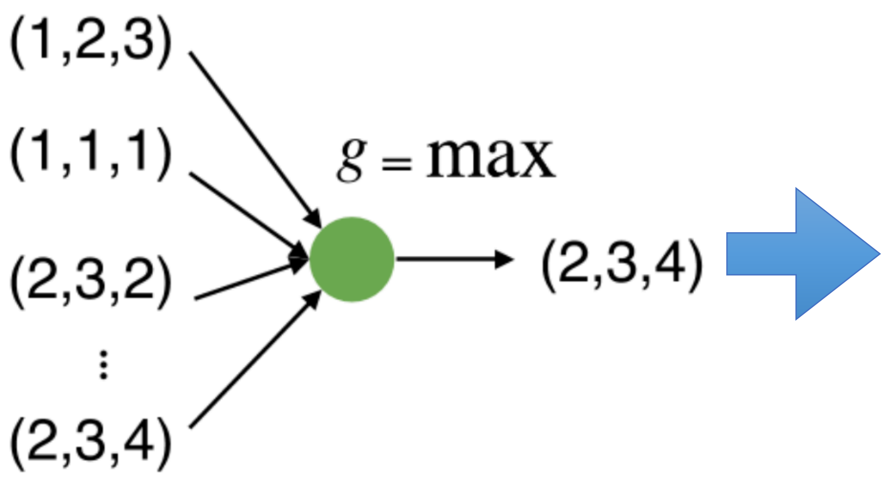
Inspired by PointNet

Charles et al. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. CVPR 2017

Any Better Solutions?

Two Stage Method, Training Dense Network → Produce sparse subnet

One Stage Method, An end-to-end method, directly producing structurally sparse Subnet.



Inspired by PointNet

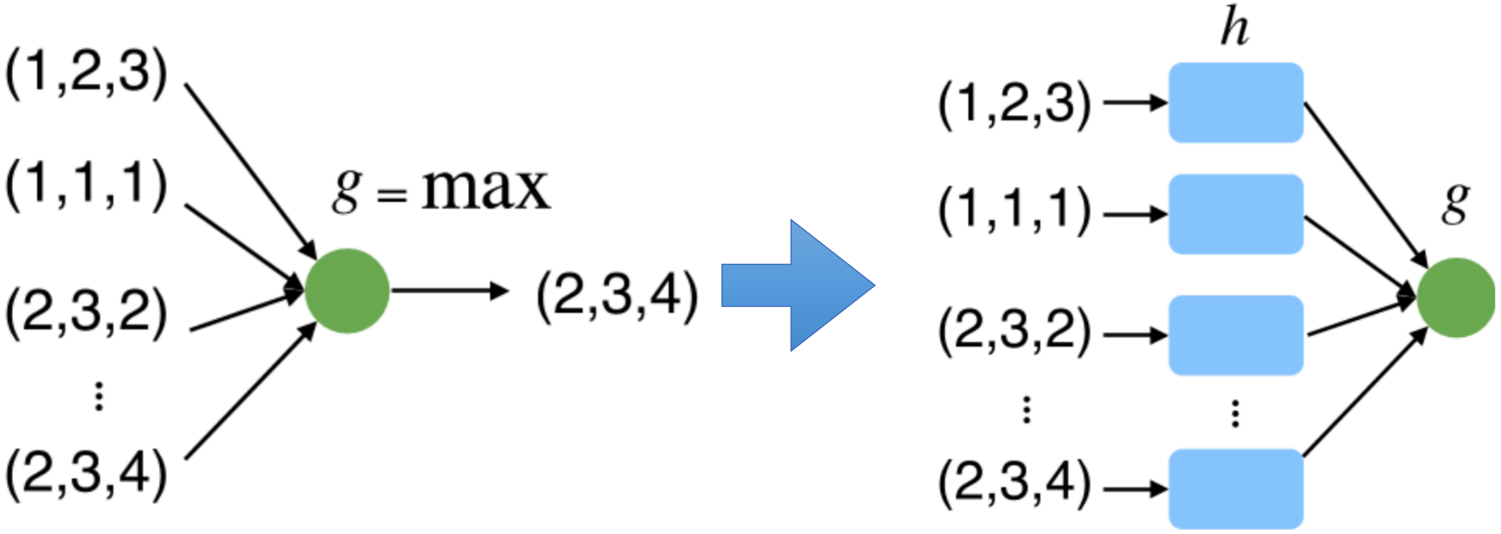
Discover naïve/extreme property of geometry

Charles et al. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. CVPR 2017

Any Better Solutions?

Two Stage Method, Training Dense Network → Produce sparse subnet

One Stage Method, An end-to-end method, directly producing structurally sparse Subnet.



Discover naïve/extreme property of geometry

Aggregation in high-dim space preserves interesting properties of the geometry

Inspired by PointNet

Charles et al. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. CVPR 2017

大数据学院 School of Data Science

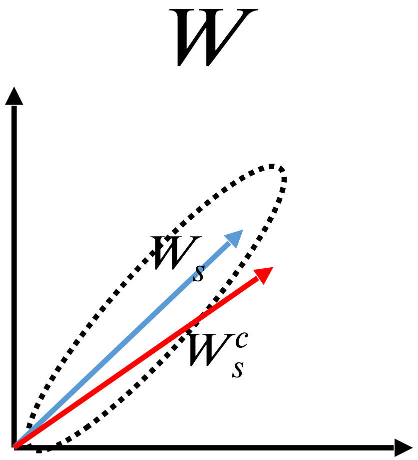
Insights for the Inverse Scale Space

W_s: wining ticket

W: dense parameters

$$W_s \cup W_{S^c} = W$$

W_s and W_{S^c} correlated



[Fu et al. ICML 2020/TPAMI2022]

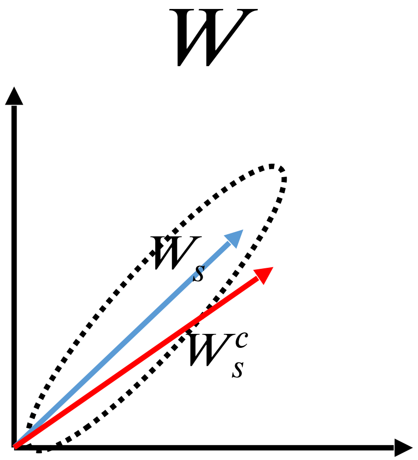
Insights for the Inverse Scale Space

W_s: wining ticket

W: dense parameters

$W_s \cup W_{S^c} = W$

W_s and W_{S^c} correlated



GD of Weight Space

$$\dot{W}_t = -\nabla_W \hat{\mathcal{L}}_n(W)$$

[Fu et al. ICML 2020/TPAMI2022]

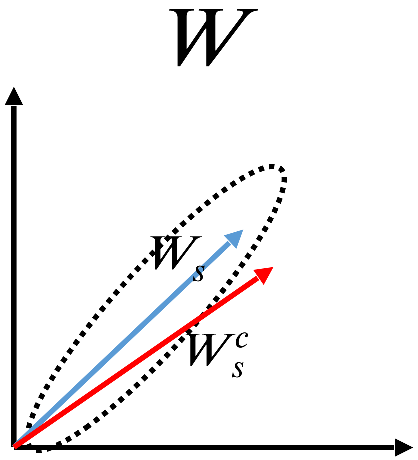
Insights for the Inverse Scale Space

W_s: wining ticket

W: dense parameters

$W_s \cup W_{S^c} = W$

W_s and W_{S^c} correlated



GD of Weight Space

$$\dot{W}_t = -\nabla_W \hat{\mathcal{L}}_n(W)$$

[Fu et al. ICML 2020/TPAMI2022]

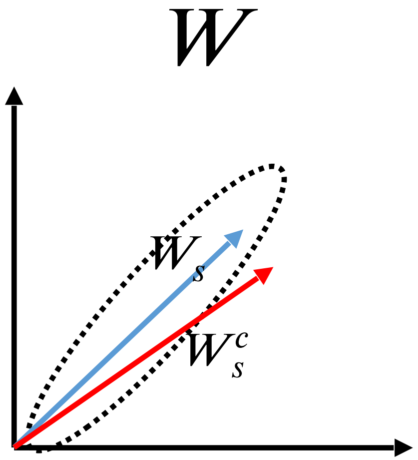
Insights for the Inverse Scale Space

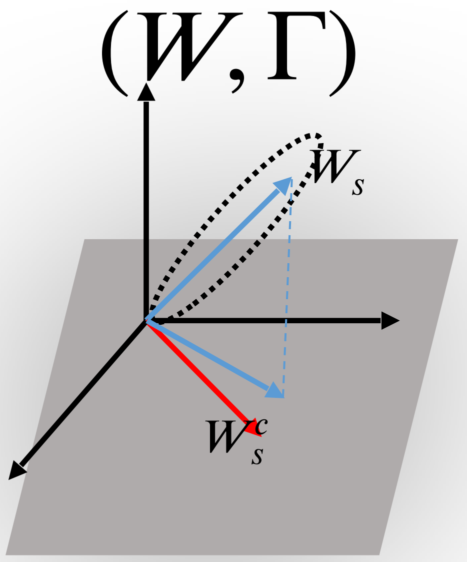
W_s: wining ticket

W: dense parameters

$W_s \cup W_{S^c} = W$

W_s and W_{S^c} correlated





$W_s \cup W_{S^c} = W$

W_s and W_s^c orthogonal

Differential Inclusion of Inverse Scale Space

GD of Weight Space

$$\dot{W}_t = -\nabla_W \hat{\mathcal{L}}_n(W)$$

[Fu et al. ICML 2020/TPAMI2022]

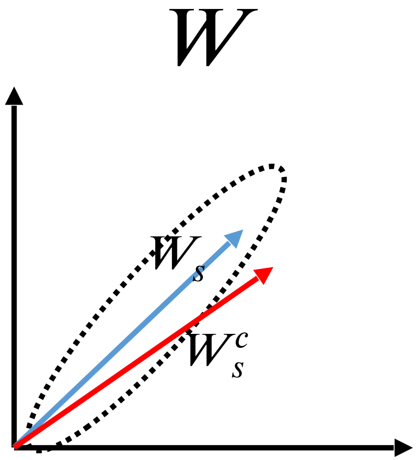
Insights for the Inverse Scale Space

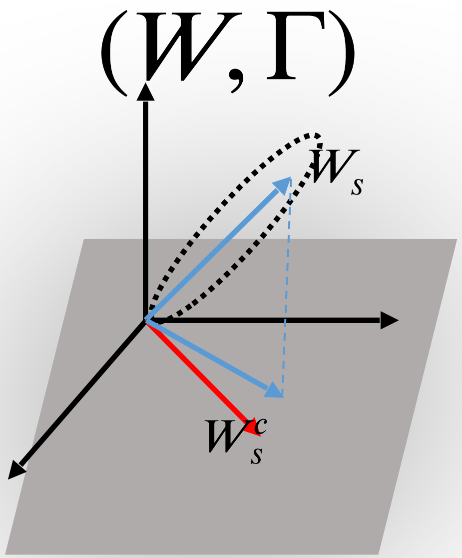
W_s: wining ticket

W: dense parameters

$W_s \cup W_{S^c} = W$

W_s and W_{S^c} correlated





$W_s \cup W_{S^c} = W$

W_s and W_s^c orthogonal

Differential Inclusion of Inverse Scale Space

GD of Weight Space

$$\dot{W}_t = -\nabla_W \hat{\mathcal{L}}_n(W)$$

$$\bar{\mathcal{L}}(W, \Gamma) = \hat{\mathcal{L}}_n(W) + \frac{1}{2\nu}\|W - \Gamma\|_F^2$$

[Fu et al. ICML 2020/TPAMI2022]

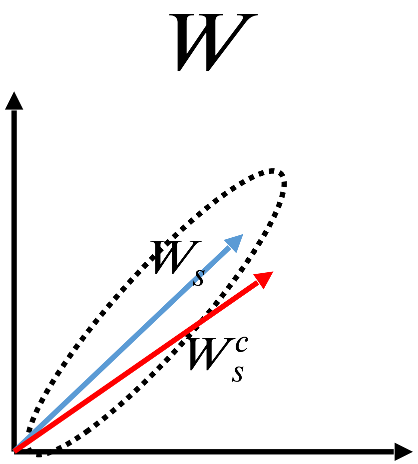
Insights for the Inverse Scale Space

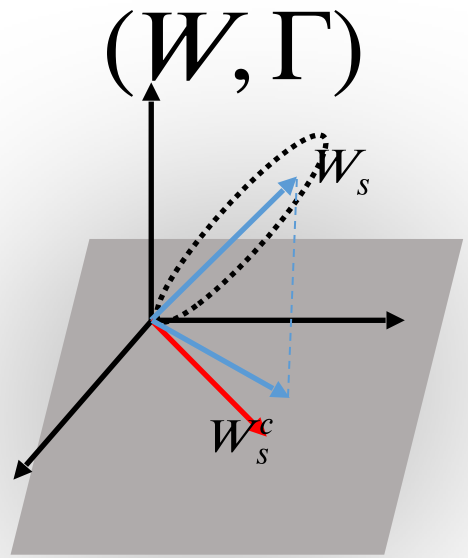
W_s: wining ticket

W: dense parameters

$W_s \cup W_{S^c} = W$

W_s and W_{S^c} correlated





$W_s \cup W_{S^c} = W$

W_s and W_s^c orthogonal

Differential Inclusion of Inverse Scale Space

GD of Weight Space

$$\dot{W}_t = -\nabla_W \hat{\mathcal{L}}_n (W)$$

$$\bar{\mathcal{L}}(W, \Gamma) = \hat{\mathcal{L}}_n (W) + \boxed{\frac{1}{2\nu}\|W - \Gamma\|_F^2}$$

[Fu et al. ICML 2020/TPAMI2022]

Formulations

$$\frac{\dot{W}_t}{\kappa} = -\nabla_W \bar{\mathcal{L}}\left(W_t, \Gamma_t\right)$$

$$\dot{V}_t = -\nabla_\Gamma \bar{\mathcal{L}}\left(W_t, \Gamma_t\right)$$

$$\boxed{V_t \in \partial\left(\Omega(\Gamma) + \frac{1}{2\kappa}\|\Gamma\|^2\right)}$$

$$\bar{\mathcal{L}}\left(W, \Gamma\right) = \hat{\mathcal{L}}_n\left(W\right) + \frac{1}{2\nu}\|W - \Gamma\|_F^2$$

Differential Inclusion of Inverse Scale Space

$$\boxed{\begin{aligned} W_{k+1} &= W_k - \kappa\alpha_k \cdot \nabla_W \bar{\mathcal{L}}\left(W_k, \Gamma_k\right) \\ V_{k+1} &= V_k - \alpha_k \cdot \nabla_\Gamma \bar{\mathcal{L}}\left(W_k, \Gamma_k\right), \\ \Gamma_{k+1} &= \kappa \cdot \mathrm{Prox}_{\Omega_\lambda}\left(V_{k+1}\right) \end{aligned}}$$

The Simple Discretization - DessLBI

V is the sub-gradient for some sparsity-enforced, often non-differentiable regularization $\Omega_\lambda\left(\Gamma\right) = \lambda\Omega_1\left(\Gamma\right), (\lambda \in \mathbb{R}_+)$ such as Lasso or group Lasso penalties for $\Omega_1\left(\Gamma\right)$

[Fu et al. ICML 2020/TPAMI2022]

Formulations

$$\frac{\dot{W}_t}{\kappa} = -\nabla_W \bar{\mathcal{L}}\left(W_t, \Gamma_t\right)$$

$$\dot{V}_t = -\nabla_\Gamma \bar{\mathcal{L}}\left(W_t, \Gamma_t\right)$$

$$\boxed{V_t \in \partial\left(\Omega(\Gamma) + \frac{1}{2\kappa}\|\Gamma\|^2\right)}$$

$$\bar{\mathcal{L}}\left(W, \Gamma\right) = \hat{\mathcal{L}}_n\left(W\right) + \frac{1}{2\nu}\|W - \Gamma\|_F^2$$

Differential Inclusion of Inverse Scale Space

$$\boxed{\begin{aligned} W_{k+1} &= W_k - \kappa\alpha_k \cdot \nabla_W \bar{\mathcal{L}}\left(W_k, \Gamma_k\right) \\ V_{k+1} &= V_k - \alpha_k \cdot \nabla_\Gamma \bar{\mathcal{L}}\left(W_k, \Gamma_k\right), \\ \Gamma_{k+1} &= \kappa \cdot \mathrm{Prox}_{\Omega_\lambda}\left(V_{k+1}\right) \end{aligned}}$$

The Simple Discretization - DessLBI

V is the sub-gradient for some sparsity-enforced, often non-differentiable regularization $\Omega_\lambda\left(\Gamma\right) = \lambda\Omega_1\left(\Gamma\right), \left(\lambda \in \mathbb{R}_+\right)$ such as Lasso or group Lasso penalties for $\Omega_1\left(\Gamma\right)$

- W_t follows the gradient descent with ℓ_2-regularization

[Fu et al. ICML 2020/TPAMI2022]

Formulations

$$\frac{\dot{W}_t}{\kappa} = -\nabla_W \bar{\mathcal{L}}\left(W_t, \Gamma_t\right)$$

$$\dot{V}_t = -\nabla_\Gamma \bar{\mathcal{L}}\left(W_t, \Gamma_t\right)$$

$$\boxed{V_t \in \partial\left(\Omega(\Gamma) + \frac{1}{2\kappa}\|\Gamma\|^2\right)}$$

$$\bar{\mathcal{L}}\left(W, \Gamma\right) = \hat{\mathcal{L}}_n\left(W\right) + \frac{1}{2\nu}\|W - \Gamma\|_F^2$$

Differential Inclusion of Inverse Scale Space

$$\boxed{\begin{aligned} W_{k+1} &= W_k - \kappa\alpha_k \cdot \nabla_W \bar{\mathcal{L}}\left(W_k, \Gamma_k\right) \\ V_{k+1} &= V_k - \alpha_k \cdot \nabla_\Gamma \bar{\mathcal{L}}\left(W_k, \Gamma_k\right), \\ \Gamma_{k+1} &= \kappa \cdot \mathrm{Prox}_{\Omega_\lambda}\left(V_{k+1}\right) \end{aligned}}$$

The Simple Discretization - DessLBI

V is the sub-gradient for some sparsity-enforced, often non-differentiable regularization $\Omega_\lambda\left(\Gamma\right) = \lambda\Omega_1\left(\Gamma\right), (\lambda \in \mathbb{R}_+)$ such as Lasso or group Lasso penalties for $\Omega_1\left(\Gamma\right)$

- W_t follows the gradient descent with ℓ_2-regularization
- Important Features of Γ_t are first selected: *Inverse Scale Space*

[Fu et al. ICML 2020/TPAMI2022]

大数据学院 School of Data Science

Formulations

$$\frac{\dot{W}_t}{\kappa} = -\nabla_W \bar{\mathcal{L}}(W_t, \Gamma_t)$$

$$\dot{V}_t = -\nabla_\Gamma \bar{\mathcal{L}}(W_t, \Gamma_t)$$

$$\boxed{V_t \in \partial\left(\Omega(\Gamma) + \frac{1}{2\kappa}\|\Gamma\|^2\right)}$$

$$\bar{\mathcal{L}}(W, \Gamma) = \hat{\mathcal{L}}_n(W) + \frac{1}{2\nu}\|W - \Gamma\|_F^2$$

Differential Inclusion of Inverse Scale Space

$$\boxed{\begin{aligned} W_{k+1} &= W_k - \kappa\alpha_k \cdot \nabla_W \bar{\mathcal{L}}(W_k, \Gamma_k) \\ V_{k+1} &= V_k - \alpha_k \cdot \nabla_\Gamma \bar{\mathcal{L}}(W_k, \Gamma_k), \\ \Gamma_{k+1} &= \kappa \cdot \mathrm{Prox}_{\Omega_\lambda}(V_{k+1}) \end{aligned}}$$

The Simple Discretization - DessLBI

$$\boxed{\begin{aligned} W_{k+1} &= W_k - \kappa\alpha_k \cdot \nabla_W \bar{\mathcal{L}}(W_k, \Gamma_k), \\ V_{k+1} &= V_k - \alpha_k \cdot \nabla_\Gamma \bar{\mathcal{L}}(W_k, \Gamma_k) \end{aligned}}$$

Gradient Descent

V is the sub-gradient for some sparsity-enforced, often non-differentiable regularization $\Omega_\lambda(\Gamma) = \lambda\Omega_1(\Gamma), (\lambda \in \mathbb{R}_+)$ such as Lasso or group Lasso penalties for $\Omega_1(\Gamma)$

- W_t follows the gradient descent with ℓ_2-regularization
- Important Features of Γ_t are first selected: *Inverse Scale Space*

[Fu et al. ICML 2020/TPAMI2022]

大数据学院 School of Data Science

Formulations

$$\frac{\dot{W}_t}{\kappa} = -\nabla_W \bar{\mathcal{L}}(W_t, \Gamma_t)$$

$$\dot{V}_t = -\nabla_\Gamma \bar{\mathcal{L}}(W_t, \Gamma_t)$$

$$\boxed{V_t \in \partial\left(\Omega(\Gamma) + \frac{1}{2\kappa}\|\Gamma\|^2\right)}$$

$$\bar{\mathcal{L}}(W, \Gamma) = \hat{\mathcal{L}}_n(W) + \frac{1}{2\nu}\|W - \Gamma\|_F^2$$

Differential Inclusion of Inverse Scale Space

$$\boxed{\begin{aligned} W_{k+1} &= W_k - \kappa\alpha_k \cdot \nabla_W \bar{\mathcal{L}}(W_k, \Gamma_k) \\ V_{k+1} &= V_k - \alpha_k \cdot \nabla_\Gamma \bar{\mathcal{L}}(W_k, \Gamma_k), \\ \Gamma_{k+1} &= \kappa \cdot \mathrm{Prox}_{\Omega_\lambda}(V_{k+1}) \end{aligned}}$$

$$\Rightarrow$$

$$\boxed{\begin{aligned} W_{k+1} &= W_k - \kappa\alpha_k \cdot \nabla_W \bar{\mathcal{L}}(W_k, \Gamma_k), \\ V_{k+1} &= V_k - \alpha_k \cdot \nabla_\Gamma \bar{\mathcal{L}}(W_k, \Gamma_k) \end{aligned}}$$

Gradient Descent

$$\boxed{\begin{aligned} \Gamma_{k+1} &= \kappa \cdot \mathrm{Prox}_\Omega(V_{k+1}) \\ \mathrm{Prox}_\Omega(V) &= \arg\min_\Gamma \left\{\frac{1}{2}\|\Gamma - V\|_2^2 + \Omega(\Gamma)\right\} \end{aligned}}$$

Proximal Mapping

The Simple Discretization - DessLBI

V is the sub-gradient for some sparsity-enforced, often non-differentiable regularization $\Omega_\lambda(\Gamma) = \lambda\Omega_1(\Gamma), (\lambda \in \mathbb{R}_+)$ such as Lasso or group Lasso penalties for $\Omega_1(\Gamma)$

- W_t follows the gradient descent with ℓ_2-regularization
- Important Features of Γ_t are first selected: *Inverse Scale Space*

[Fu et al. ICML 2020/TPAMI2022]

Proximal Mapping Controls the Sparsity

$$\mathrm{Prox}_{\Omega}(V) = \arg\min_{\Gamma} \left\{ \frac{1}{2}\|\Gamma - V\|_2^2 + \Omega\left(\Gamma\right) \right\}$$

[Fu et al. ICML 2020/TPAMI2022]

Proximal Mapping Controls the Sparsity

$$\mathrm{Prox}_{\Omega}(V) = \arg\min_{\Gamma} \left\{ \frac{1}{2}\|\Gamma - V\|_2^2 + \Omega(\Gamma) \right\}$$

DessLBI enforce structural sparsity by Group lasso penalty,

$$\Omega(\Gamma) = \sum_g \|\Gamma^g\|_2 = \sum_g \sqrt{\sum_{i=1}^{|\Gamma^g|} (\Gamma_i^g)^2}$$

A close form solution:

$$\Gamma^g = \kappa \cdot \max\left(0, 1 - 1/\|V^g\|_2\right) V^g$$

[Fu et al. ICML 2020/TPAMI2022]

Proximal Mapping Controls the Sparsity

$$\mathrm{Prox}_{\Omega}(V) = \arg\min_{\Gamma} \left\{ \frac{1}{2}\|\Gamma - V\|_2^2 + \Omega\left(\Gamma\right) \right\}$$

DessLBI enforce structural sparsity by Group lasso penalty,

$$\Omega(\Gamma) = \sum_g \|\Gamma^g\|_2 = \sum_g \sqrt{\sum_{i=1}^{|\Gamma^g|} \left(\Gamma_i^g\right)^2}$$

A close form solution:

$$\Gamma^g = \kappa \cdot \max\left(0, 1 - 1/\|V^g\|_2\right) V^g$$

Convolutional layer, $\Gamma^g = \Gamma^g(c_{in}, c_{out}, \texttt{size})$

Fully connected layer $\Gamma = \Gamma(c_{in}, c_{out})$

c_{in} :No. of input channel

c_{out} :No. of output channel

size: kernel size

[Fu et al. ICML 2020/TPAMI2022]

Proximal Mapping Controls the Sparsity

$$\mathrm{Prox}_{\Omega}(V) = \arg\min_{\Gamma} \left\{ \frac{1}{2}\|\Gamma - V\|_2^2 + \Omega(\Gamma) \right\}$$

DessLBI enforce structural sparsity by Group lasso penalty,

$$\Omega(\Gamma) = \sum_g \|\Gamma^g\|_2 = \sum_g \sqrt{\sum_{i=1}^{|\Gamma^g|} (\Gamma_i^g)^2}$$

A close form solution:

$$\Gamma^g = \kappa \cdot \max(0, 1 - 1/\|V^g\|_2)\, V^g$$

Convolutional layer, $\Gamma^g = \Gamma^g(c_{in}, c_{out}, \texttt{size})$

Fully connected layer $\Gamma = \Gamma(c_{in}, c_{out})$

c_{in} :No. of input channel

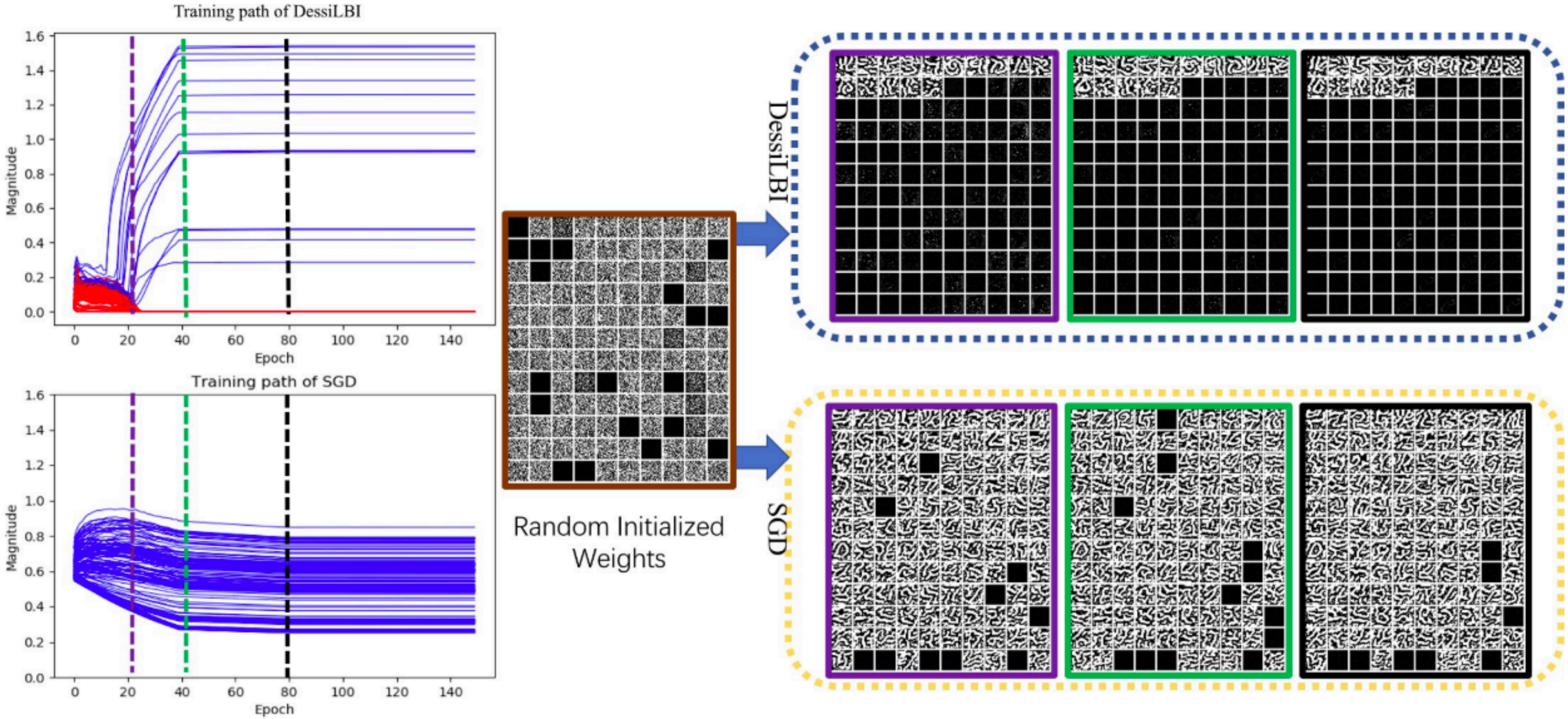
c_{out} :No. of output channel

size: kernel size

- (Batch) DessLBI w./w.o. Momentum and Weight-decay (Mom-Wd)
- We have a theorem that guarantees the global convergence of DessiLBI: from any initialization, DessiLBI sequence converges to a critical point.

[Fu et al. ICML 2020/TPAMI2022]

Visualization of Sparse Filters

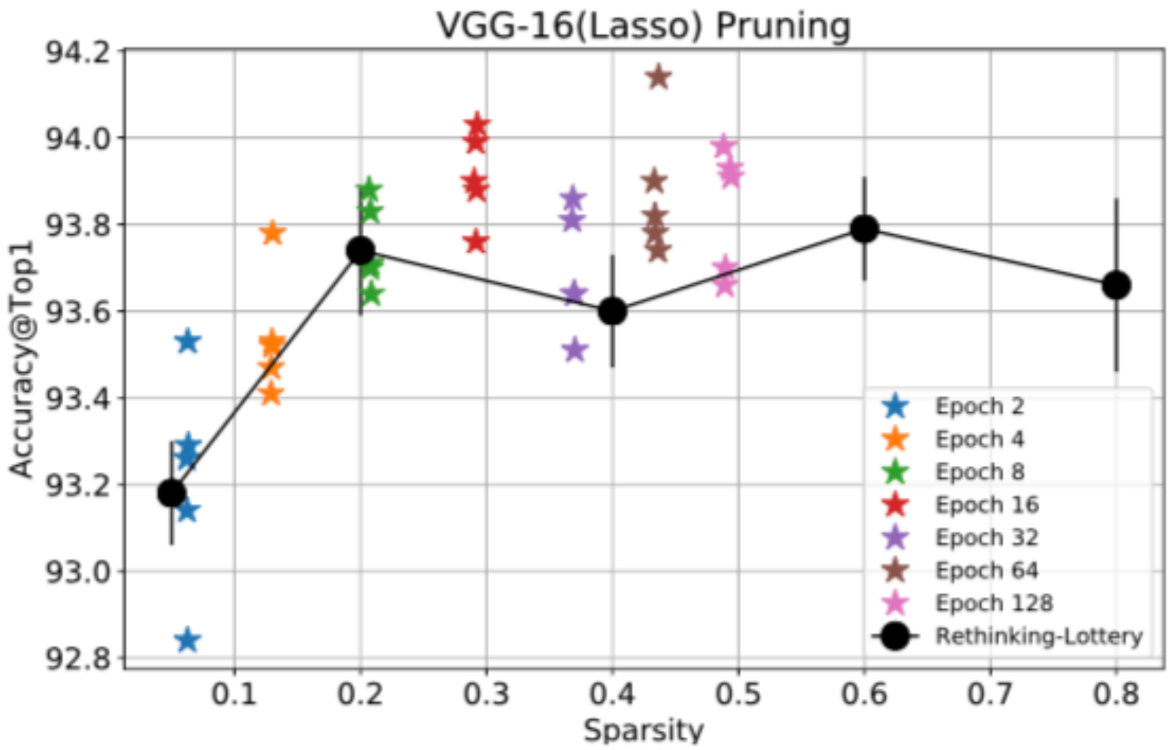


Solution path and filter patterns of the 3rd conv. layer of LetNet-5 on MNIST

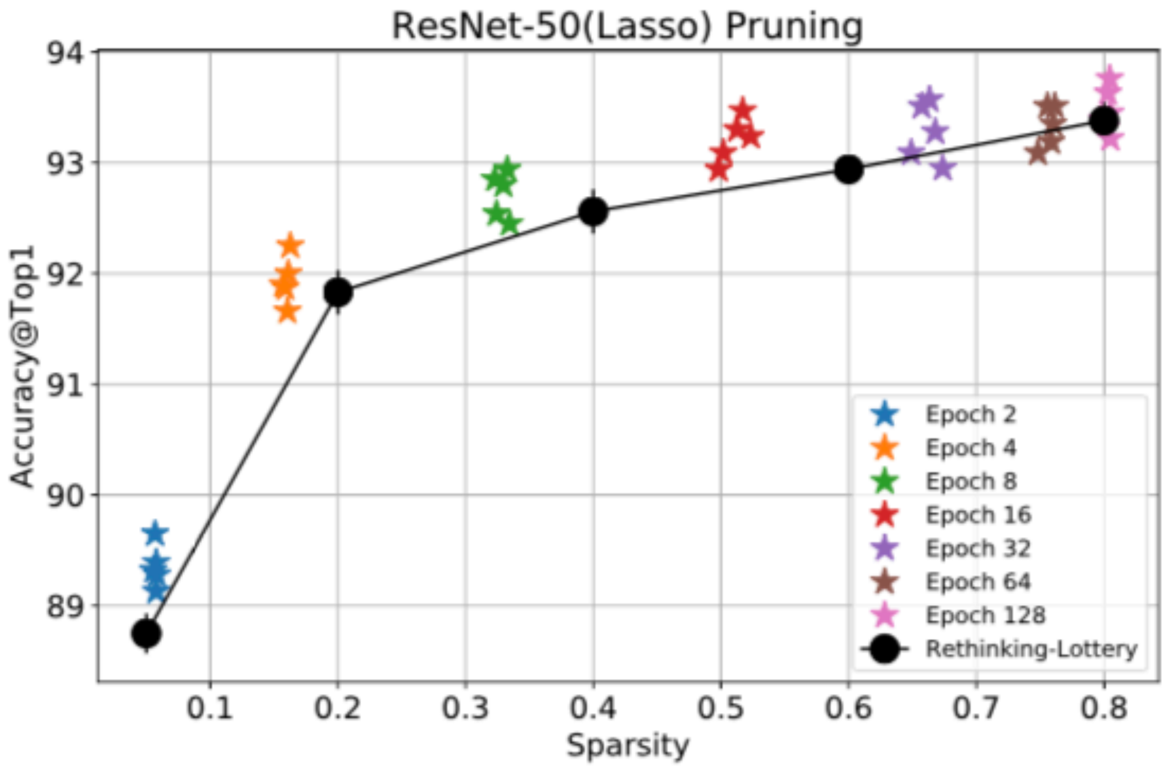
[Fu et al. ICML 2020/TPAMI2022]

大数据学院 School of Data Science

Winning Tickets by DessiLBI



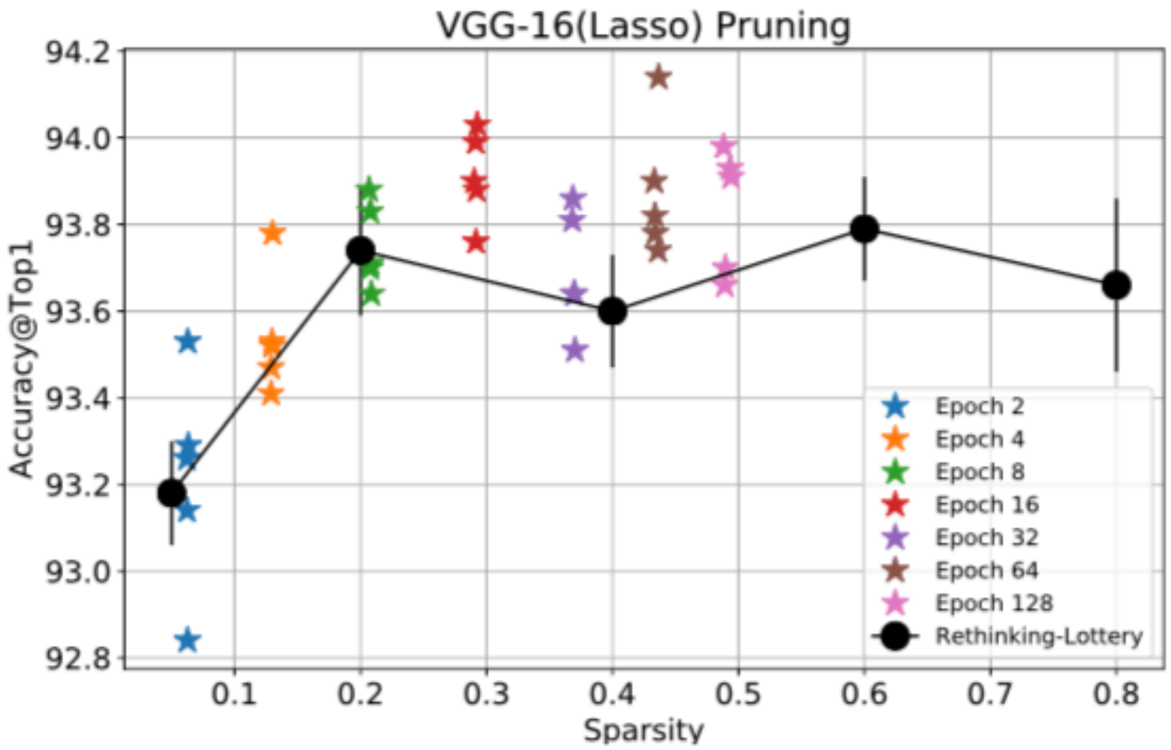
(c) VGG-16 (Lasso)



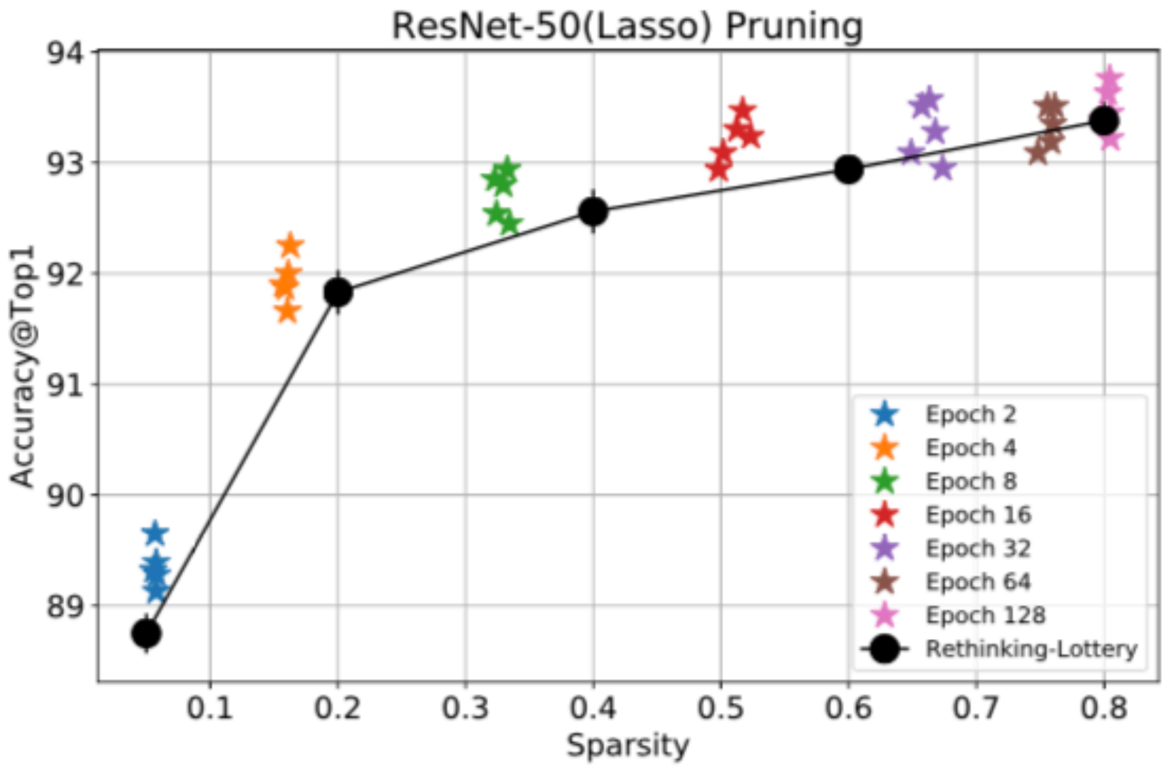
(d) ResNet-50 (Lasso)

[Fu et al. ICML 2020/TPAMI2022]

Winning Tickets by DessiLBI



(c) VGG-16 (Lasso)

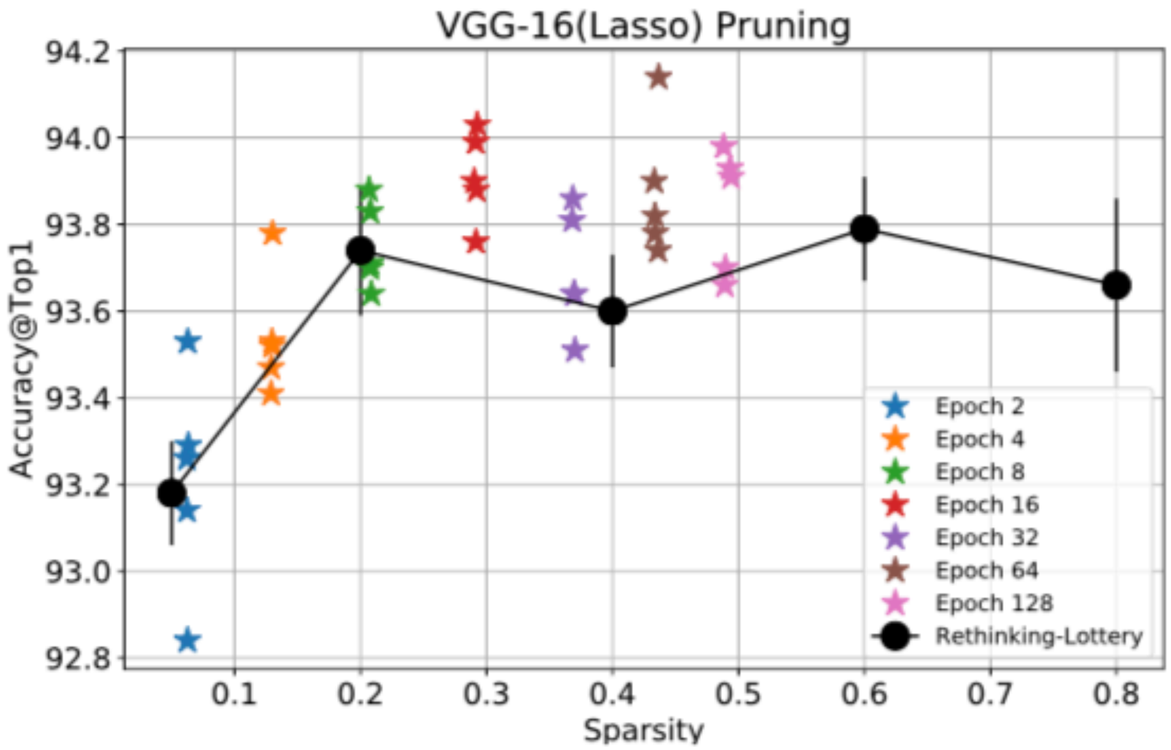


(d) ResNet-50 (Lasso)

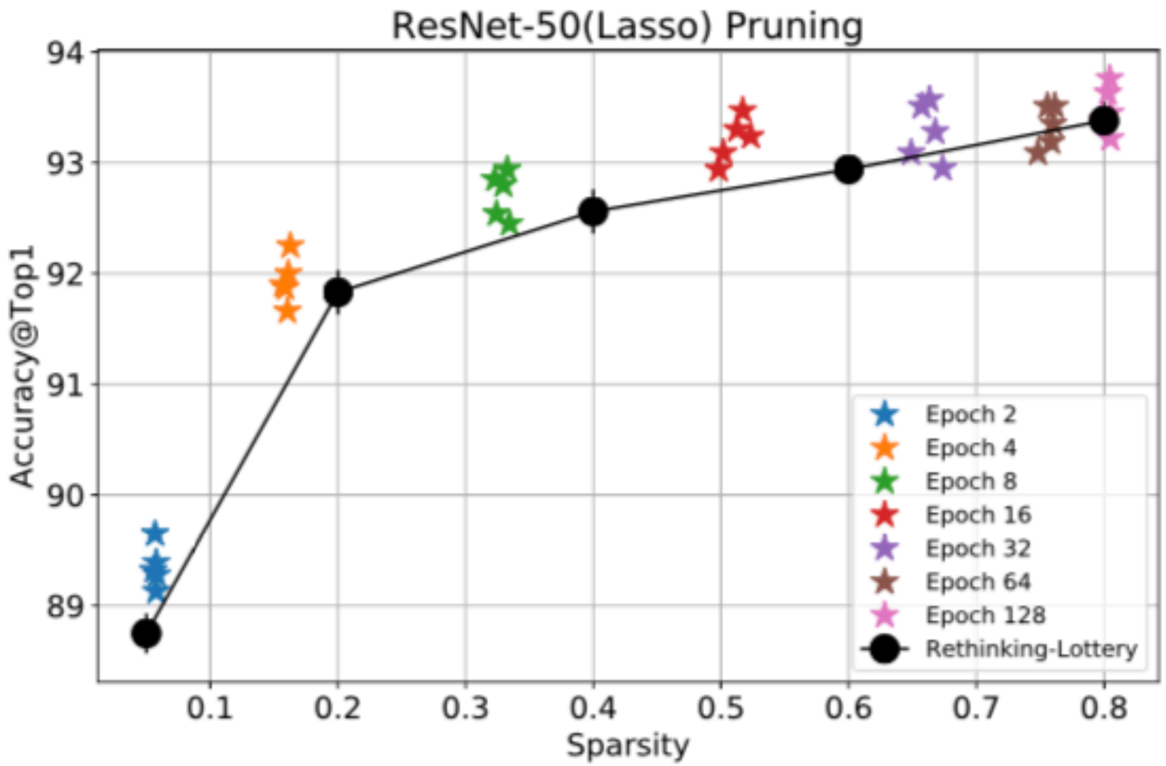
Train DessiLBI with Early Stopping

[Fu et al. ICML 2020/TPAMI2022]

Winning Tickets by DessiLBI



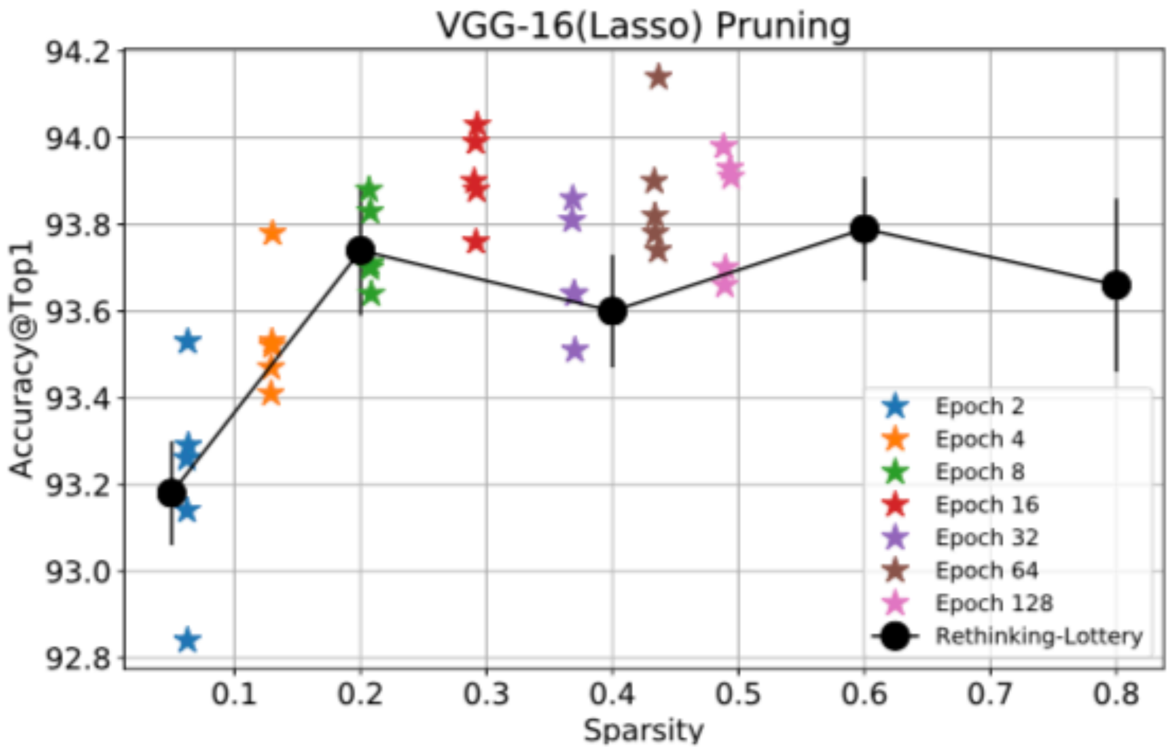
(c) VGG-16 (Lasso)



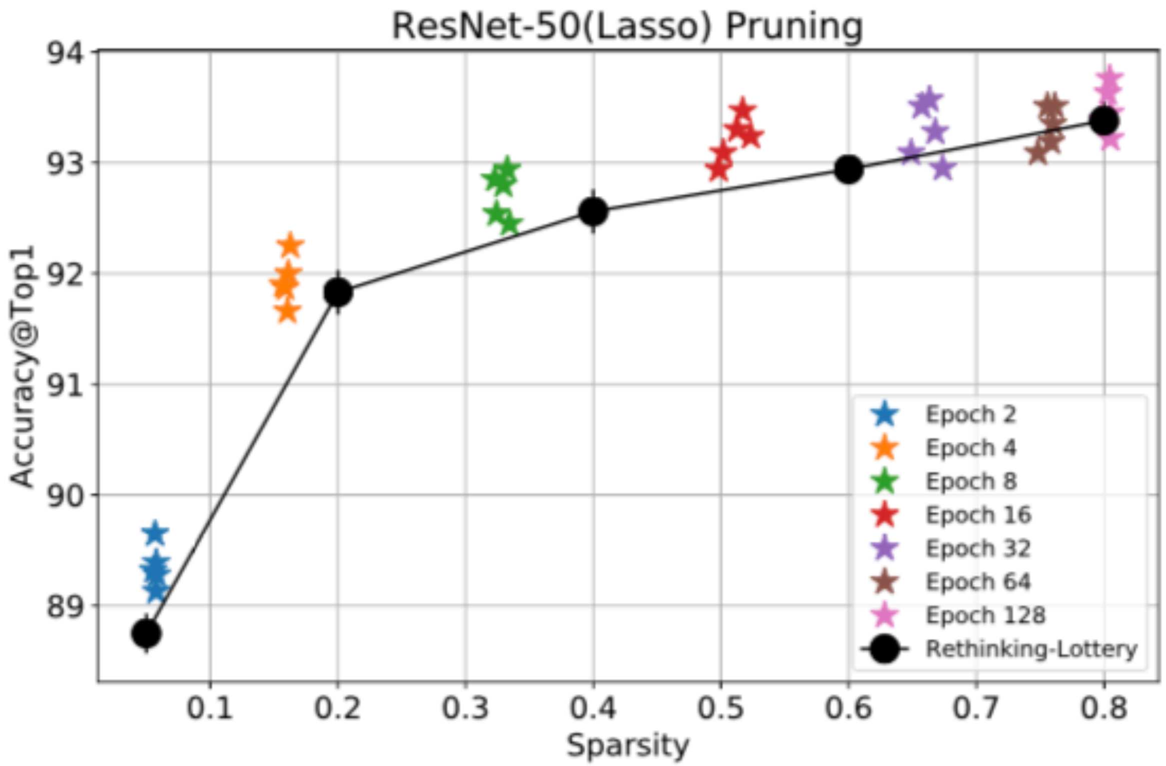
(d) ResNet-50 (Lasso)

[Fu et al. ICML 2020/TPAMI2022]

Winning Tickets by DessiLBI



(c) VGG-16 (Lasso)



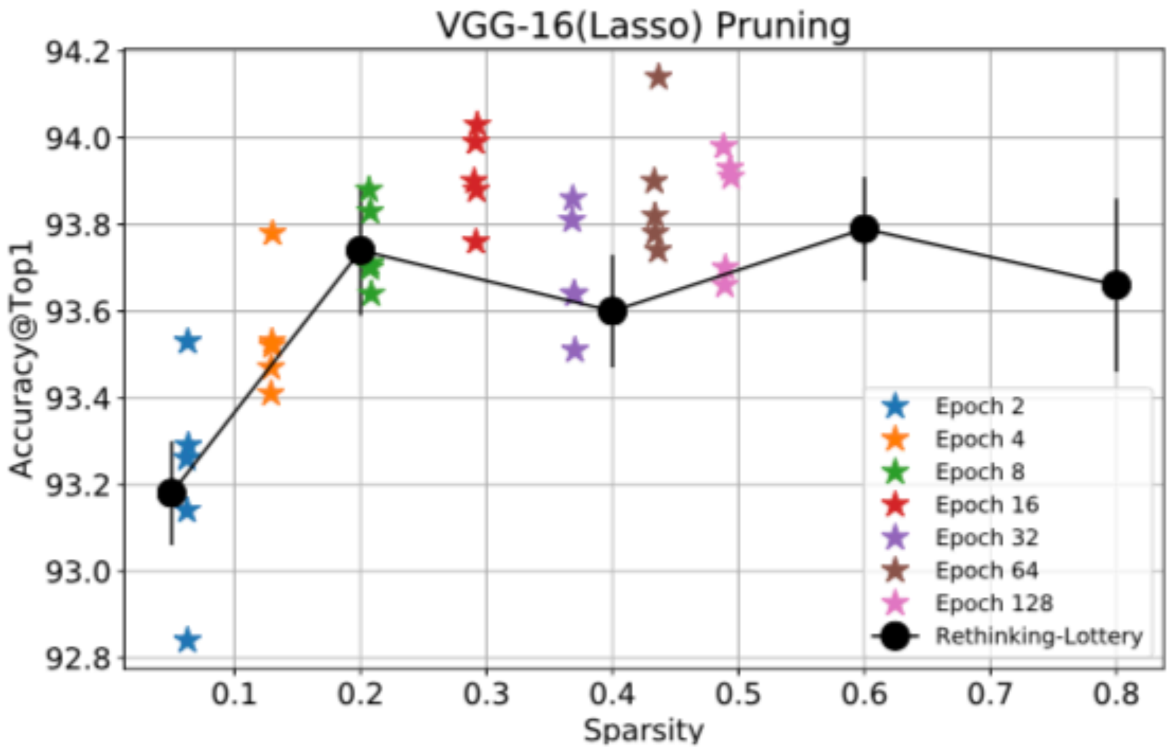
(d) ResNet-50 (Lasso)

Extract Γ as subnetwork structure

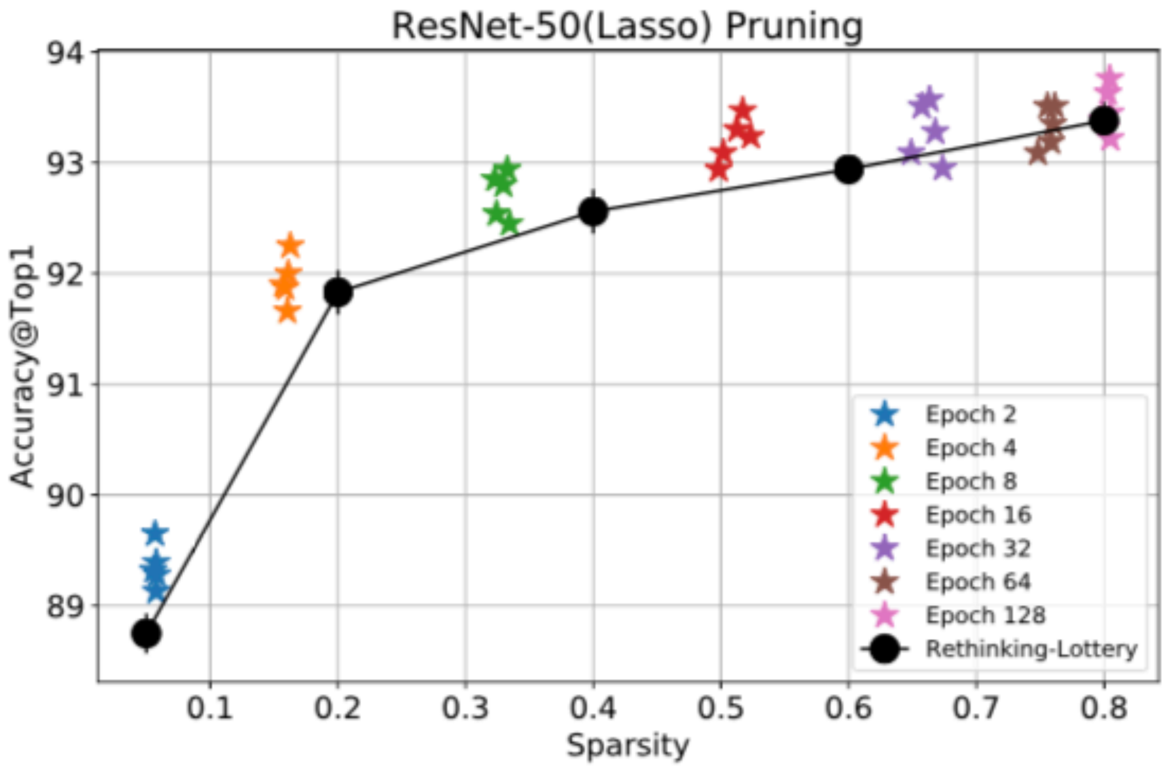
[Fu et al. ICML 2020/TPAMI2022]

大数据学院 School of Data Science

Winning Tickets by DessiLBI



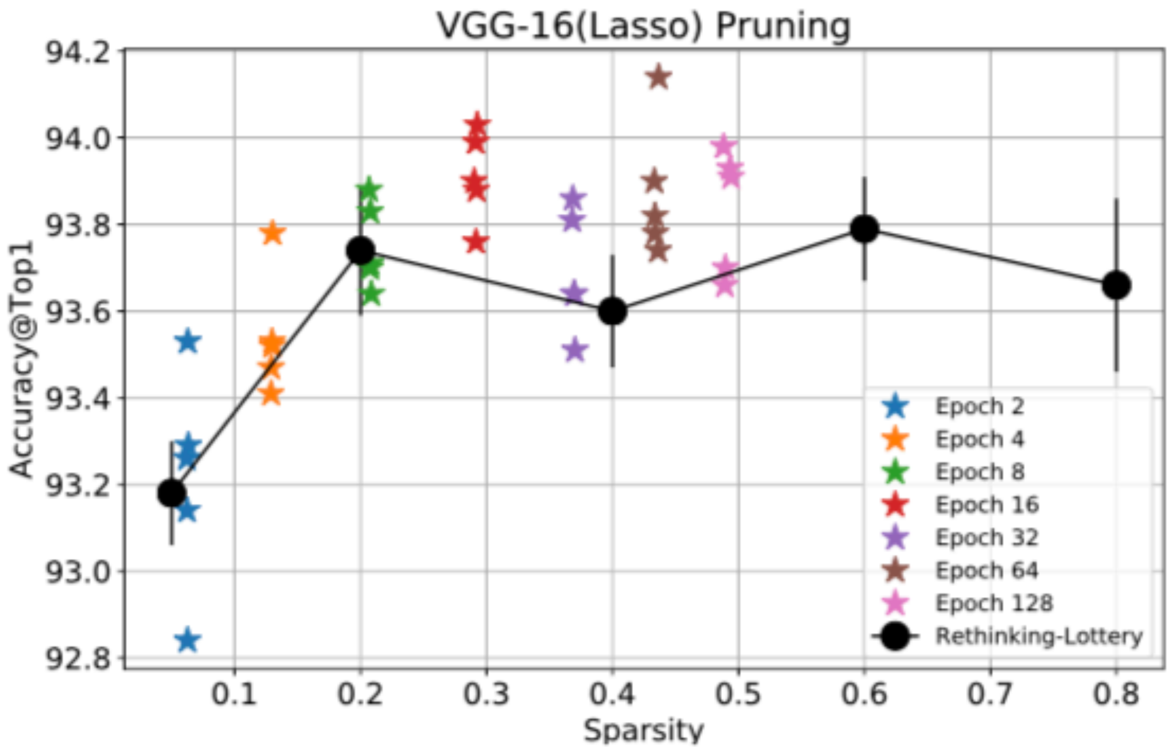
(c) VGG-16 (Lasso)



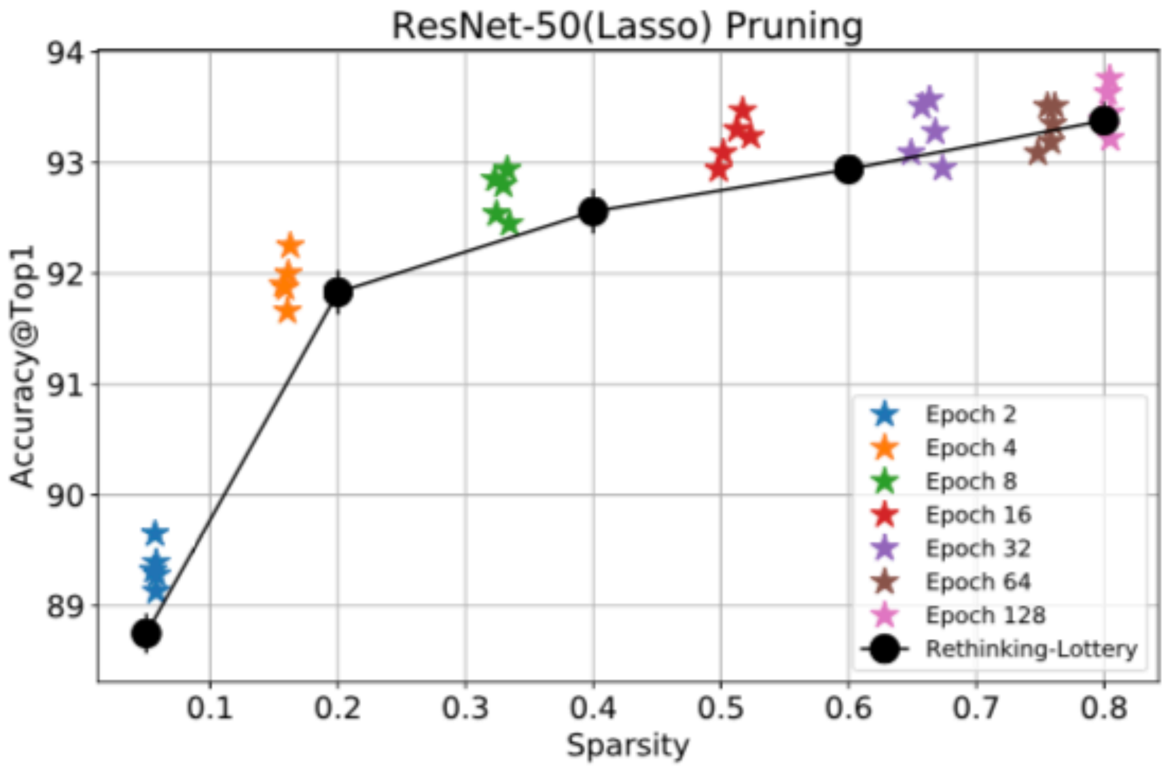
(d) ResNet-50 (Lasso)

[Fu et al. ICML 2020/TPAMI2022]

Winning Tickets by DessiLBI



(c) VGG-16 (Lasso)

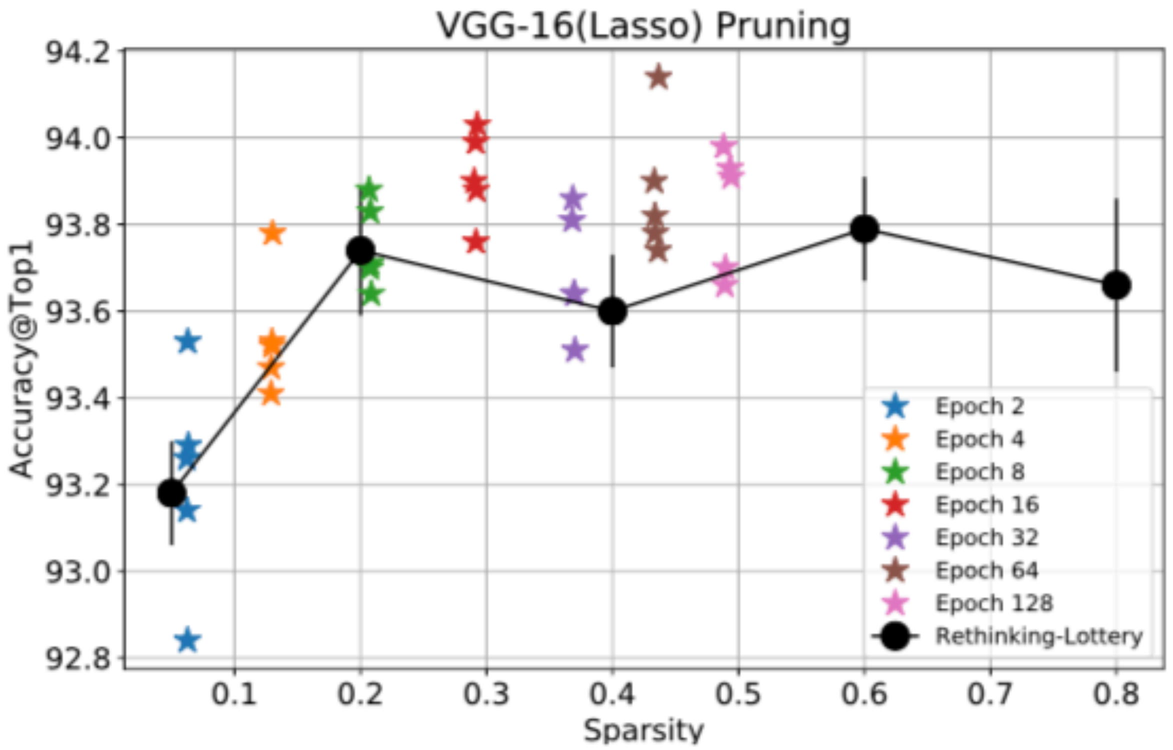


(d) ResNet-50 (Lasso)

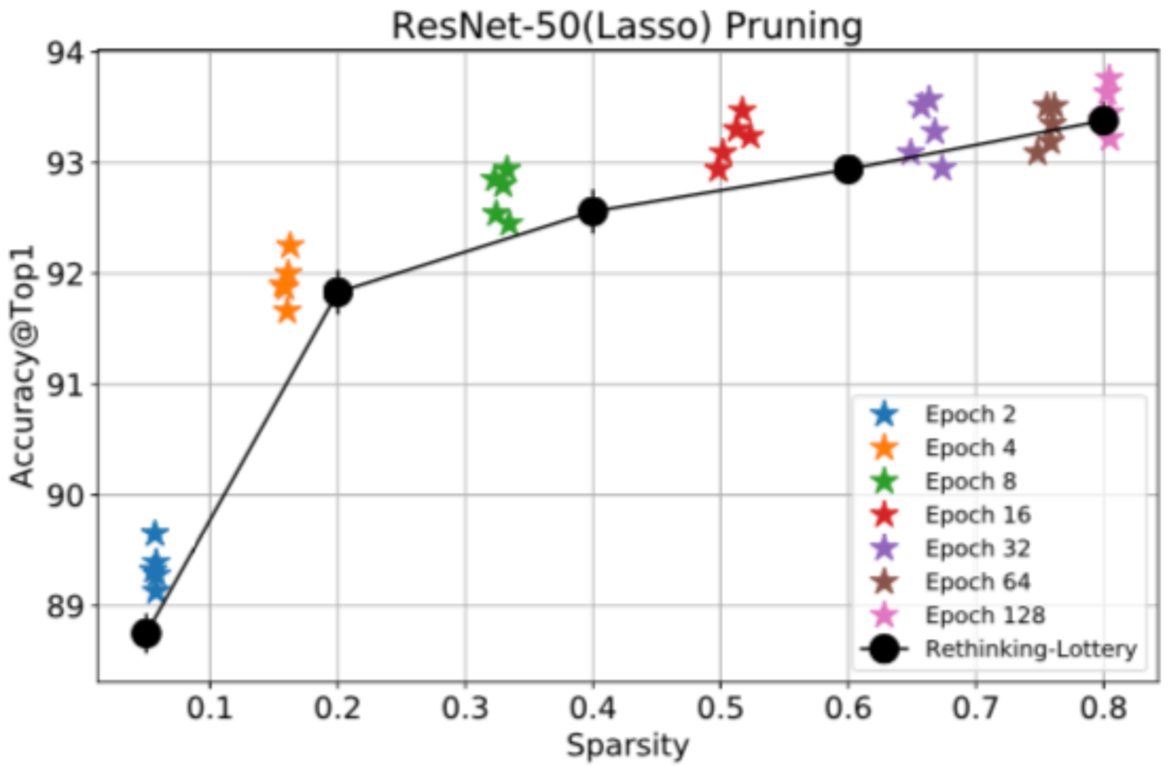
Retrain subnetwork

[Fu et al. ICML 2020/TPAMI2022]

Winning Tickets by DessiLBI



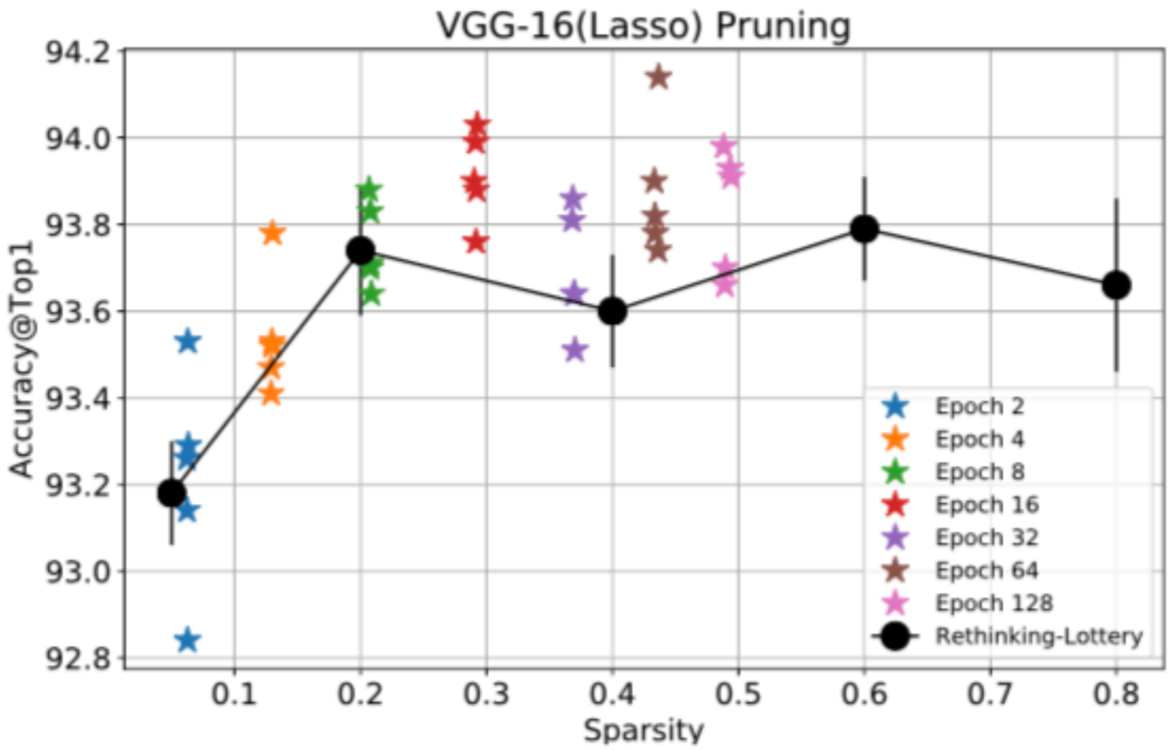
(c) VGG-16 (Lasso)



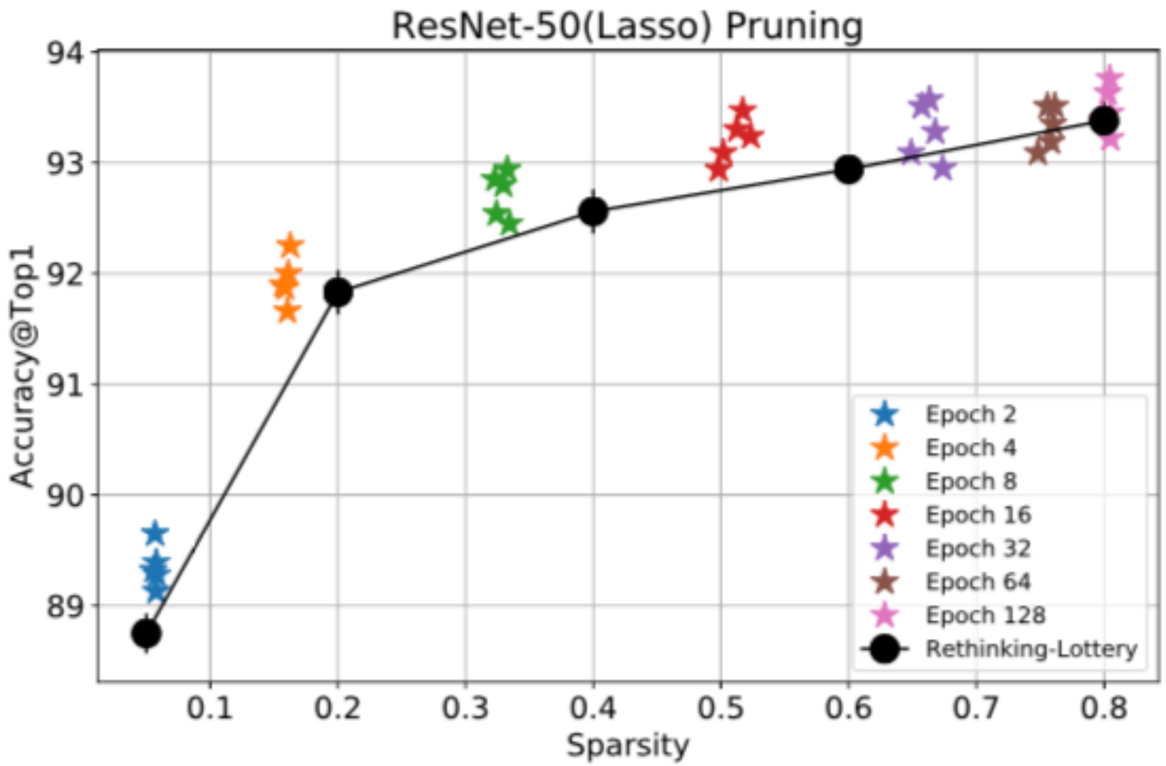
(d) ResNet-50 (Lasso)

[Fu et al. ICML 2020/TPAMI2022]

Winning Tickets by DessiLBI



(c) VGG-16 (Lasso)

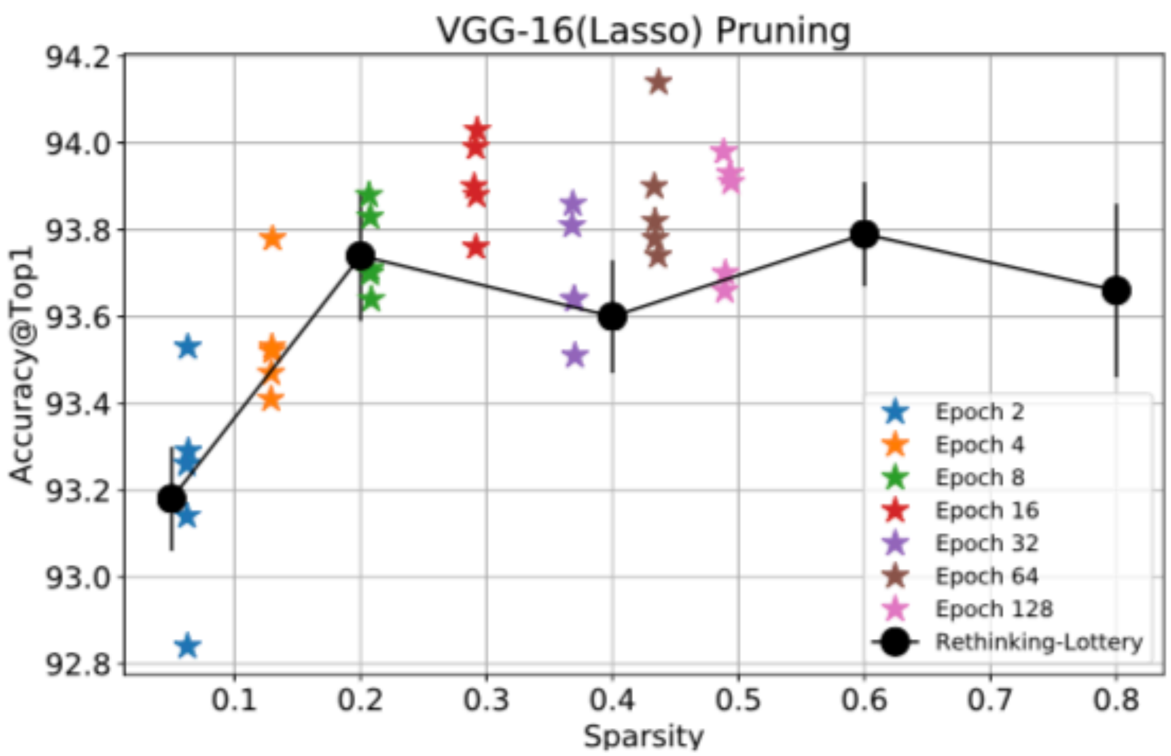


(d) ResNet-50 (Lasso)

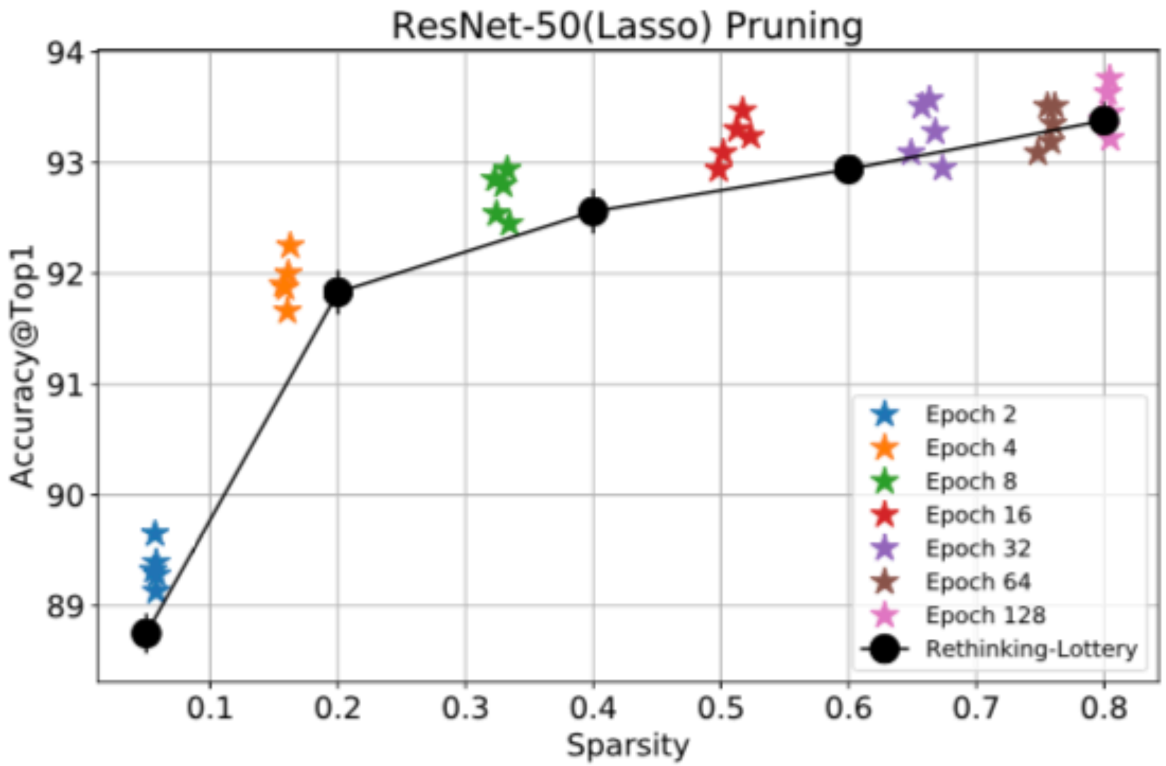
Winning Tickets

[Fu et al. ICML 2020/TPAMI2022]

Winning Tickets by DessiLBI



(c) VGG-16 (Lasso)



(d) ResNet-50 (Lasso)

[Fu et al. ICML 2020/TPAMI2022]

大数据学院
School of Data Science

Toolbox: Very Easy to Use

https://github.com/DessiLBI2020/DessiLBI

It is install-free, put slbi_opt.py and slbi_toolbox.py into the project folder and import them.

Quick Example to Start with,

python ./example/train/train_lenet.py

To initialize the toolbox, the following codes are needed.

```
from slbi_toolbox import SLBI_ToolBox
import torch
optimizer = SLBI_ToolBox(model.parameters(), lr=args.lr, kappa=args.kappa, mu=args.mu, weight_decay=0)
optimizer.assign_name(name_list)
optimizer.initialize_slbi(layer_list)
```

For training a neural network, the process is similar to one that uses built-in optimizer

```
optimizer.zero_grad()
loss.backward()
optimizer.step()
```

Training Neural Network

The training process is the same as original Pytorch Optimizer

ImageNet Training Example

This part of code is included in example/imagenet. To do this demo, run

```
python train_imagenet_slbi.py
```

```python
for ep in range(args.epoch):
    model.train()
    descent_lr(args.lr, ep, optimizer, args.interval)
    loss_val = 0
    correct = num = 0
    for iter, pack in enumerate(train_loader):
        data, target = pack[0].to(device), pack[1].to(device)
        logits = model(data)
        loss = F.nll_loss(logits, target)
        optimizer.zero_grad()
        loss.backward()
        optimizer.step()
        _, pred = logits.max(1)
        loss_val += loss.item()
        correct += pred.eq(target).sum().item()
        num += data.shape[0]
```

```python
if  'z_buffer' in param_state:
        new_grad = d_p * lr_kappa + (p.data - param_state['gamma_buffer']) * lr_kappa / mu
        last_p = copy.deepcopy(p.data)
        p.data.add_(-new_grad)
        param_state['z_buffer'].add_(-lr_gamma, param_state['gamma_buffer'] - last_p)
        if len(p.data.size()) == 2:
                param_state['gamma_buffer'] = kappa * self.shrink(param_state['z_buffer'], 1)
        elif len(p.data.size()) == 4:
                param_state['gamma_buffer'] = kappa * self.shrink_group(param_state['z_buffer'])
        else:
                pass
else:
        p.data.add_(-lr_kappa, d_p)#for bias update as vanilla sgd
```

We record the path via two buffer during training

https://github.com/DessiLBI2020/DessiLBI

Training Neural Network

The training process is the same as original Pytorch Optimizer

ImageNet Training Example

This part of code is included in example/imagenet. To do this demo, run

```
python train_imagenet_slbi.py
```

It is a network optimizer with

- finding important structural sparsity in model learning,
- Shorter training time,
- Exploring regularization path,
- Nice theoretical properties,
- Good interpretation of important parameters

https://github.com/DessiLBI2020/DessiLBI

```
for ep in range(args.epoch):
    model.train()
    descent_lr(args.lr, ep, optimizer, args.interval)
    loss_val = 0
    correct = num = 0
    for iter, pack in enumerate(train_loader):
        data, target = pack[0].to(device), pack[1].to(device)
        logits = model(data)
        loss = F.nll_loss(logits, target)
        optimizer.zero_grad()
        loss.backward()
        optimizer.step()
        _, pred = logits.max(1)
        loss_val += loss.item()
        correct += pred.eq(target).sum().item()
        num += data.shape[0]
```

```
if  'z_buffer' in param_state:
        new_grad = d_p * lr_kappa + (p.data - param_state['gamma_buffer']) * lr_kappa / mu
        last_p = copy.deepcopy(p.data)
        p.data.add_(-new_grad)
        param_state['z_buffer'].add_(-lr_gamma, param_state['gamma_buffer'] - last_p)
        if len(p.data.size()) == 2:
                param_state['gamma_buffer'] = kappa * self.shrink(param_state['z_buffer'], 1)
        elif len(p.data.size()) == 4:
                param_state['gamma_buffer'] = kappa * self.shrink_group(param_state['z_buffer'])
        else:
                pass
else:
        p.data.add_(-lr_kappa, d_p)#for bias update as vanilla sgd
```

We record the path via two buffer during training

Pruning Neural Network

We can prune the network according to the information of augmented variable Γ

For pruning a neural network, the code is as follows.

```
optimizer.update_prune_order(epoch)
optimizer.prune_layer_by_order_by_list(percent, layer_name)
```

Filter Pruning

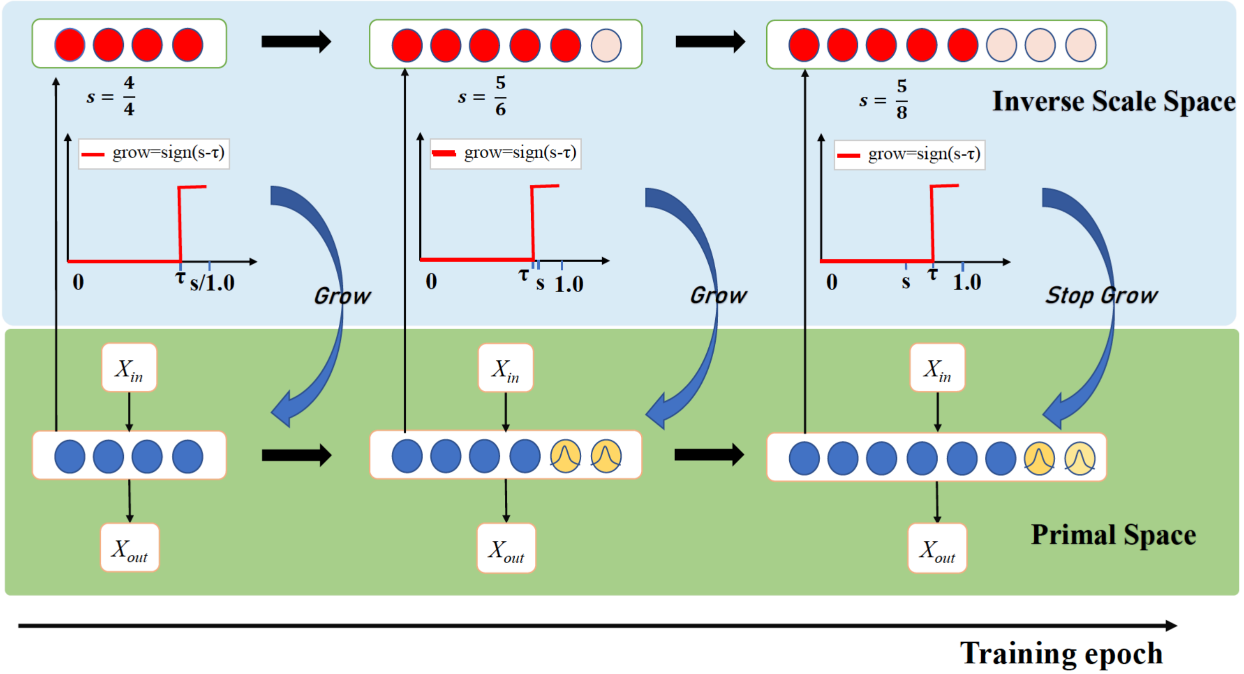
```
ts_reshape = torch.reshape(param_state['w_star'], (param_state['w_star'].shape[0], -1))
ts_norm = torch.norm(ts_reshape, 2, 1)
num_selected_filters = torch.sum(ts_norm != 0).item()
param_state['original_params'] = copy.deepcopy(p.data)
p.data = param_state['w_star']
```

Weight Pruning

```
num_selected_units = (param_state['w_star'] > 0.0).sum().item()
param_state['original_params'] = copy.deepcopy(p.data)
p.data = param_state['w_star']
```

Growing Neural Network

We add new filters according to the support set of augmented Γ, to enlarge the model capacity.



Dataset	Method	Params.	Acc(%)
CIFAR10	AutoGrow [124]	4.06 M	94.27
	Ours	2.69 M	94.82
CIFAR100	AutoGrow [124]	5.13 M	74.72
	Ours	3.37 M	76.86

```python
def grow_filter(model, new_arc, NET, args, logger, topk_dict=None):
    # new_arc: [basic_block, [block_num list], [filter_num list]]
    # layer_name: the layer to be growed
    old_params = {}
    for n, p in model.named_parameters():
        if 'module' in n:
            n = '.'.join(n.split('.')[1:])
        old_params[n] = p.data

    new_net = NET(new_arc[0], new_arc[1], new_arc[2], num_classes=new_arc[3], resolution=new_arc[4])

    for n, p in new_net.named_parameters():
        if n in old_params.keys():
            if p.data.size() != old_params[n].size(): #this layer grown
                old_size = old_params[n].size()
                if len(old_size) == 4:
                    try:
                        filter_idx = topk_dict[n]
                        n_out, n_in, k1, k2 = old_size
                        for idx in filter_idx:
                            p.data[idx, :n_in, :k1, :k2] = old_params[idx, :, :, :]
                    except: #shortcut weight
                        n_out, n_in, k1, k2 = old_size
                        p.data[:n_out, :n_in, :k1, :k2] = old_params[n]

                elif len(old_size) == 2:
                    num_out, num_in = old_size
                    p.data[:num_out, :num_in] = old_params[n]
                elif len(old_size) == 1:
                    a, = old_size
                    p.data[:a] = old_params[n]
            else:  #this layer did not grow
                p.data = old_params[n]
            #logger.info('{} has succeed parameters from last model!'.format(n))

        else:
            pass

    return new_net
```

Different from ADMM

$$W_{k+1} = \underset{W}{\operatorname{argmax}} \mathcal{L}\left(W\right) + \frac{\rho}{2}\left\|W - \Gamma_k + U_k\right\|^2$$
$$\Gamma_{k+1} = \underset{W}{\operatorname{argmax}} \Omega\left(\Gamma\right) + \frac{\rho}{2}\left\|W_{k+1} - \Gamma + U_k\right\|^2$$
$$U_{k+1} = U_k + W_{k+1} - \Gamma_{k+1}$$

- Different from ours
    - ADMM targets on convergence result, with objective function: $\mathcal{L}\left(W\right) + \lambda \cdot \Omega\left(W\right)$
        - DessiLBI is discretization of Differential Inclusion
    - DessiLBI cares the regularized solution path; it returns a sequence of models from simple to complex, corresponding to different regularization parameters;

[Fu et al. ICML 2020/TPAMI2022]

THANKS

Yanwei Fu
School of Data Science, Fudan University
yanweifu@fudan.edu.cn

大数据学院
School of Data Science

References: Our Works

- Fu et al. Interestingness Prediction by Robust Learning to Rank. ECCV 2014
- Fu et al. Robust Subjective Visual Property Prediction from Crowdsourced Pairwise Labels. IEEE TPAMI 2016
- Wang et al. Instance Credibility Inference for Few-Shot Learning. CVPR 2020
- Wang et al. How to Trust Unlabeled Data? Instance Credibility Inference for Few-Shot Learning. IEEE TPAMI 2021
- Wang et al. Scalable Penalized Regression for Noise Detection in Learning with Noisy Labels. CVPR 2022
- Zhao et al. MSplit LBI: Realizing Feature Selection and Dense Estimation Simultaneously in Few-shot and Zero-shot Learning. ICML 2018
- Fu et al. DessiLBI: Exploring Structural Sparsity on Deep Network via Differential Inclusion Paths. ICML 2020
- Fu et al. Exploring Structural Sparsity of Deep Networks via Inverse Scale Spaces. IEEE TPAMI accepted (2022)

References

- Tutorial on Sparse and Low-rank Modeling, European Conference on Computer Vision, Firenze, Italy, October 2012
- Yiyuan She and Art B Owen. Outlier detection using nonconvex penalized regression. Journal of the American Statistical Association, 2011.
- Sparse Networks from Scratch: Faster Training without Losing Performance. Tim Dettmers and Luke Zettlemoyer. 2019.
- Bommasani et al. On the Opportunities and Risks of Foundation Models. arx:108.07258, 2021
- CLIP: Learning Transferable Visual Models From Natural Language Supervision, arXiv Feb. 24, ICML2021, OpenAI
- ALIGN : Scaling Up Visual and Vision-Language Representation Learning With Noisy Text Supervision, ICML2021, Google Research
- ALBEF : Align before Fuse: Vision and Language Representation Learning with Momentum Distillation, NeurIPS 2021, Salesforce Research
- Florence: A New Foundation Model for Computer Vision, arXiv, Nov. 22, 2021, Microsoft Cloud and AI, Microsoft Research Redmond
- NUWA: Visual Synthesis Pre-training for Neural visUal World creAtion, arXiv Nov. 24, 2021, MSRA, Peking Unv.
- INTERN: A New Learning Paradigm Towards General Vision, arXiv Nov. 16, 2021 Shanghai AI Laboratory, SenseTime, CUKH, SJTU
- Gopher: Scaling Language Models: Methods, Analysis & Insights from Training Gopher, arXiv Dec. 8, 2021, DeepMind
- FLAVA: A Foundational Language And Vision Alignment Model, CVPR 2022, FAIR
- OPT: Open Pre-trained Transformer Language Models. Meta AI 2022

References

- Collins, Maxwell D., and Pushmeet Kohli. "Memory bounded deep convolutional networks." *arXiv preprint arXiv:1412.1442* (2014).
- Loshchilov, Ilya, and Frank Hutter. "Decoupled weight decay regularization." *arXiv preprint arXiv:1711.05101* (2017).
- Han, Song, et al. "Learning both weights and connections for efficient neural network." *Advances in neural information processing systems*. 2015.
- Arora, Sanjeev, et al. "Stronger generalization bounds for deep nets via a compression approach." *arXiv preprint arXiv:1802.05296* (2018).
- Generating Long Sequences with Sparse Transformers. Rewon Child, Scott Gray, Alec Radford, Ilya Sutskever. https://arxiv.org/pdf/1904.10509.pdf. 2019
- Dettmers, et al. Sparse Networks from Scratch:Faster Training without Losing Performance. Arxiv 2019
- Fabian et al., A Comparative Framework for Preconditioned Lasso Algorithms, NIPS 2013.
- Wright et al. Sharp Thresholds for High-Dimensional and Noisy Sparsity Recovery Using l1-Constrained Quadratic Programming (Lasso). IEEE TIT 2009
- Fan,et al. Partial consistency with sparse incidental parameters. Statistica Sinica, 2018
- Geirhos et al. IMAGENET-TRAINED CNNS ARE BIASED TOWARDS TEXTURE; INCREASING SHAPE BIAS IMPROVES ACCURACY AND ROBUSTNESS. ICLR 2019

References

- Neyshabur, B., Li, Z., Bhojanapalli, S., LeCun, Y., and Srebro, N. The role of over-parametrization in generalization of neural networks. In International Conference on Learning Representations (ICLR), New Orleans, Louisiana, USA. 2019.
- Allen-Zhu, Z., Li, Y., and Song, Z. A convergence theory for deep learning via over-parameterization. ICML 2019

Learning for Sparse Optimization

Prof. Yuan Yao: Inverse Scale Space Method and Statistical Properties

Outlines

1. Inverse scale space method, differential inclusions, linearized bregman iterations and mirror descent
2. Structural sparsity and splitting method
3. Statistical regularization path and model selection consistency
4. Huber's robust statistics and outlier detection
5. False discovery rate control and (split) Knockoffs

Prof. Wotao Yin: Learning to Optimize (L2O)

Outlines

1. L2O idea and typical work flow.
2. Unrolling a classic algorithm and learning its parameters
3. Generalization and convergence safeguard
4. Learning regularization and plug-and-play
5. Fixed-point network and Jacobian-free back propagation