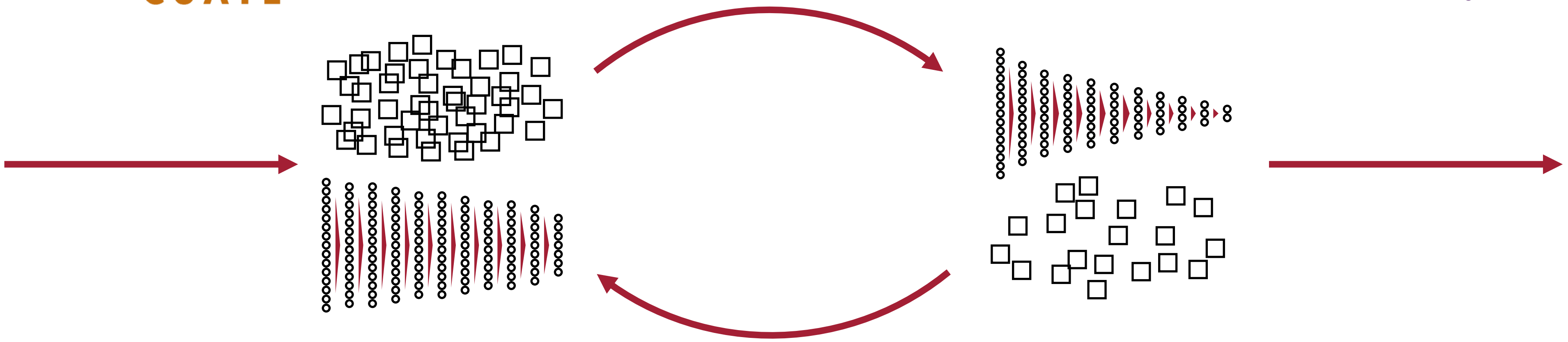
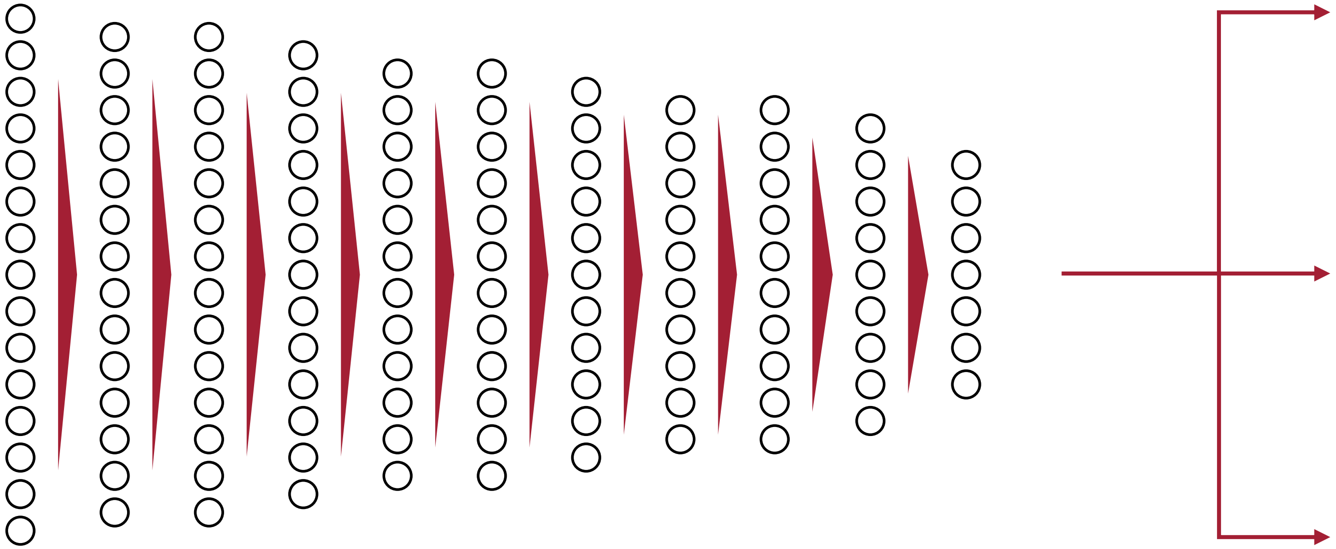


# Compressing Neural Networks: Towards Determining the Optimal Layer-wise Decomposition

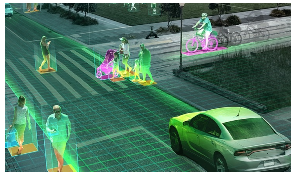
Lucas Liebenwein*, Alaa Maalouf*, Dan Feldman, Daniela Rus

* Equal contribution

# Neural networks are SOTA



Natural Language Processing



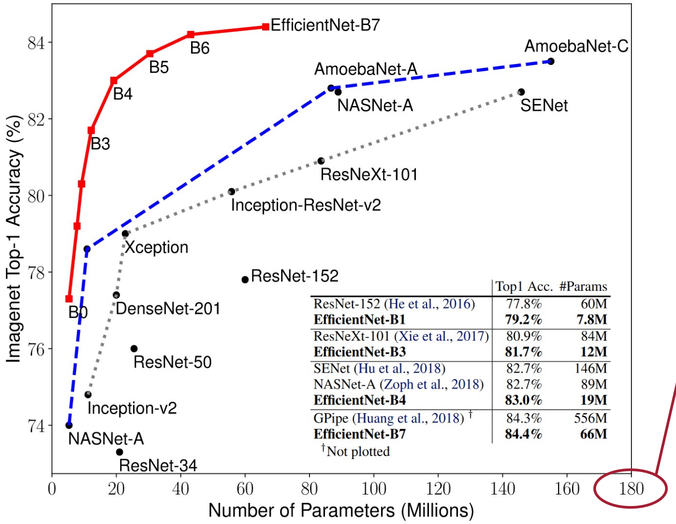
Computer Vision

Robotics

# The bigger, the better

180 m

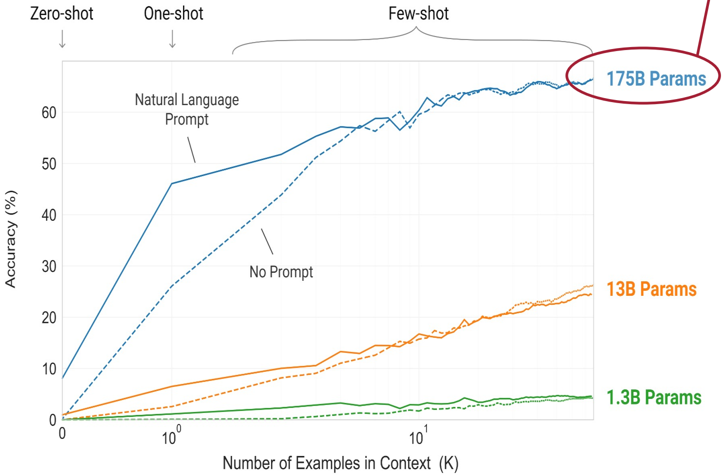
175B




| | Top1 Acc. | #Params |
|---|---|---|
| ResNet-152 (He et al., 2016) | 77.8% | 60M |
| **EfficientNet-B1** | **79.2%** | **7.8M** |
| ResNeXt-101 (Xie et al., 2017) | 80.9% | 84M |
| **EfficientNet-B3** | **81.7%** | **12M** |
| SENet (Hu et al., 2018) | 82.7% | 146M |
| NASNet-A (Zoph et al., 2018) | 82.7% | 89M |
| **EfficientNet-B4** | **83.0%** | **19M** |
| GPipe (Huang et al., 2018) † | 84.3% | 556M |
| **EfficientNet-B7** | **84.4%** | **66M** |

†Not plotted

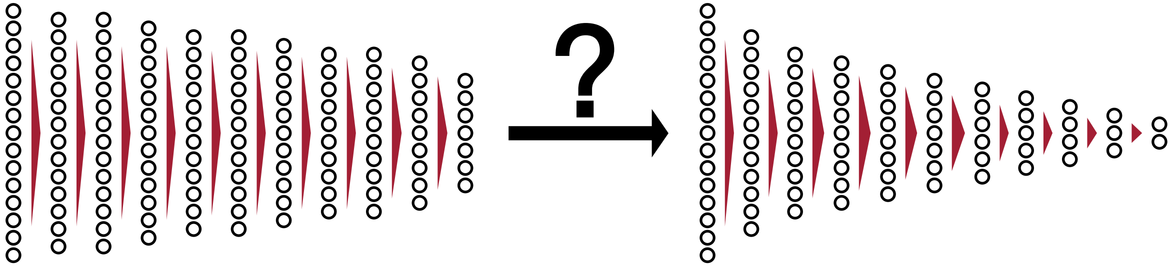
Tan, Mingxing, and Quoc Le. "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks." ICML. 2019.



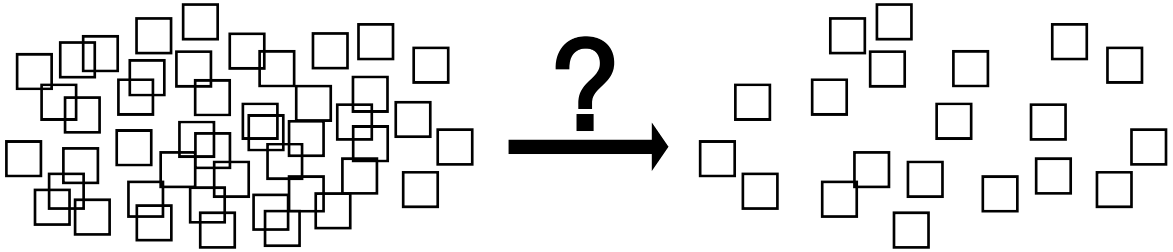

Brown, Tom, et al. "Language models are few-shot learners." arXiv preprint arXiv:2005.14165 (2020).

# Less resources, same performance?

Less Parameters




Effects on Generalization



# Outline of the talk

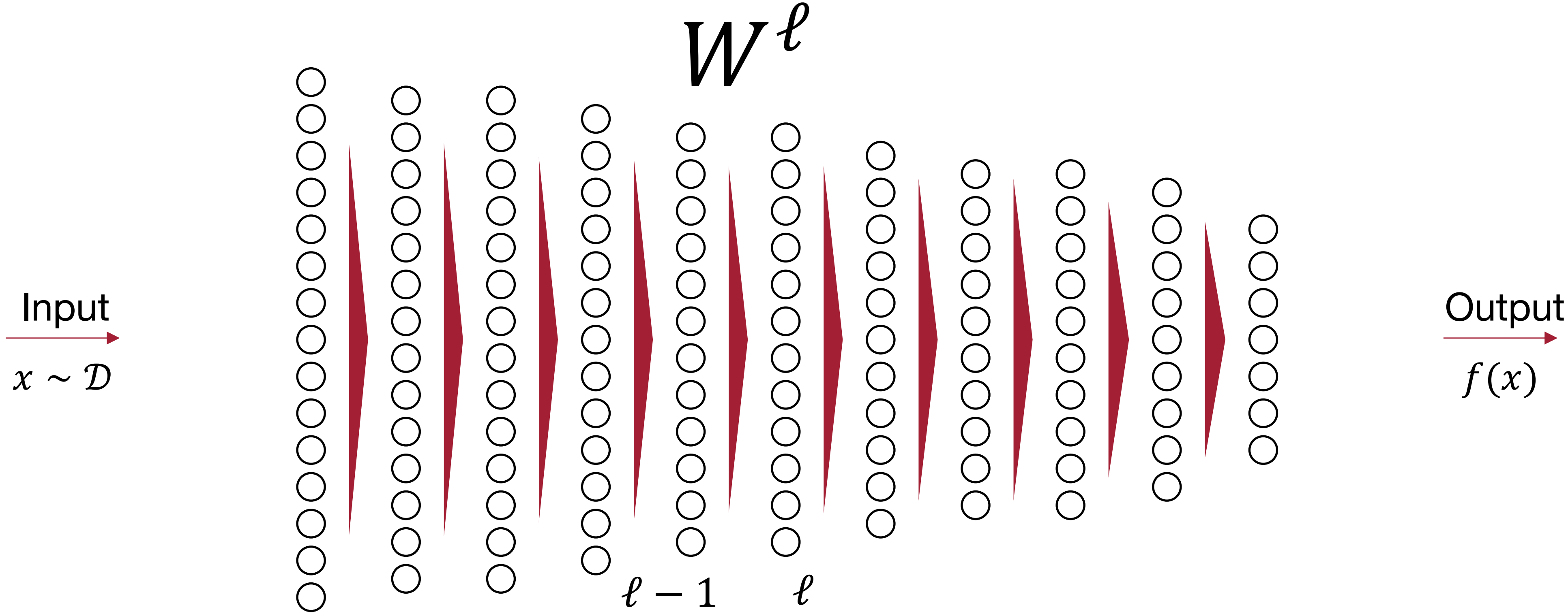
1. Introduce neural network compression via low-rank decomposition

2. Understand limitations of prior work and our contribution

3. Present our main algorithm **ALDS (Automatic Layer-wise Decomposition Selector)**

4. Discuss results, future work, and discussion

# Compression via low-rank decomposition

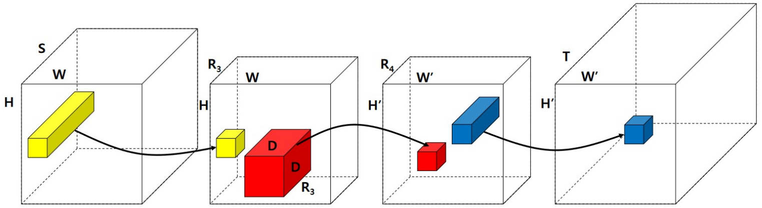


# Compression via low-rank decomposition

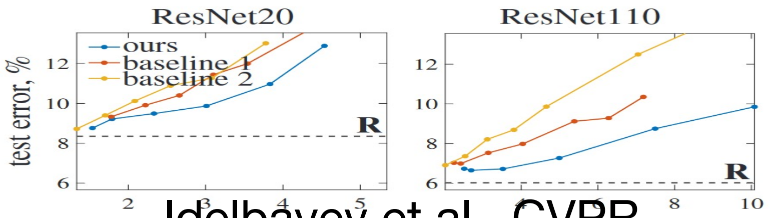
$$W^{\ell} = \overset{f\times c}{\left(\quad\right)} \approx \overset{f\times j}{\left(\quad\right)} * \overset{j\times c}{\left(\quad\right)} =: \hat{U}^{\ell} * \hat{V}^{\ell}$$

$\#\, params = fc \;\longrightarrow\; \#\, params = j(f+c)$

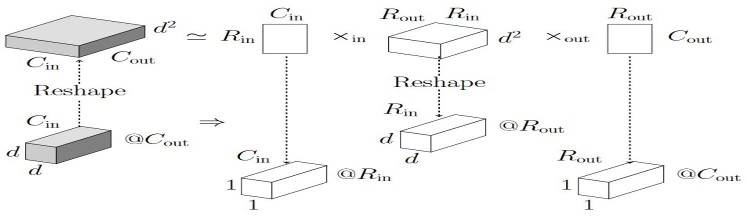
# Related work




Kim et al., ICLR 2016




Idelbayev et al., CVPR 2020




Gusak et al., ICCV 2019

✓ Per-layer low-rank decomposition:
**“local step”**

✓ Progressive decomposition + training:
**“retraining strategy”**

✓ Tuning of per-layer decomposition:
**“global step”**

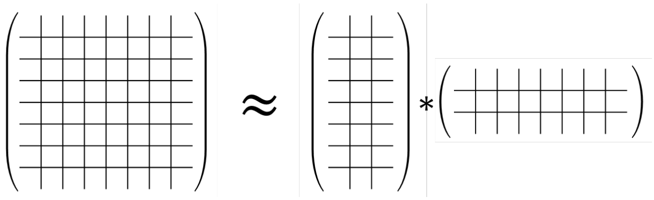
✗ **Unresolved**: combined **local + global step** for optimal network compression

**Our main contribution**

# A general approach to low-rank compression

1

"Local step"

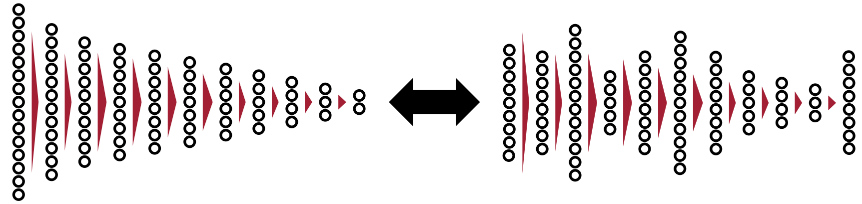


$\varepsilon^{\ell} = \varepsilon^{\ell}(\text{compression\_ratio})$

Efficiently implementable and easy to evaluate

2

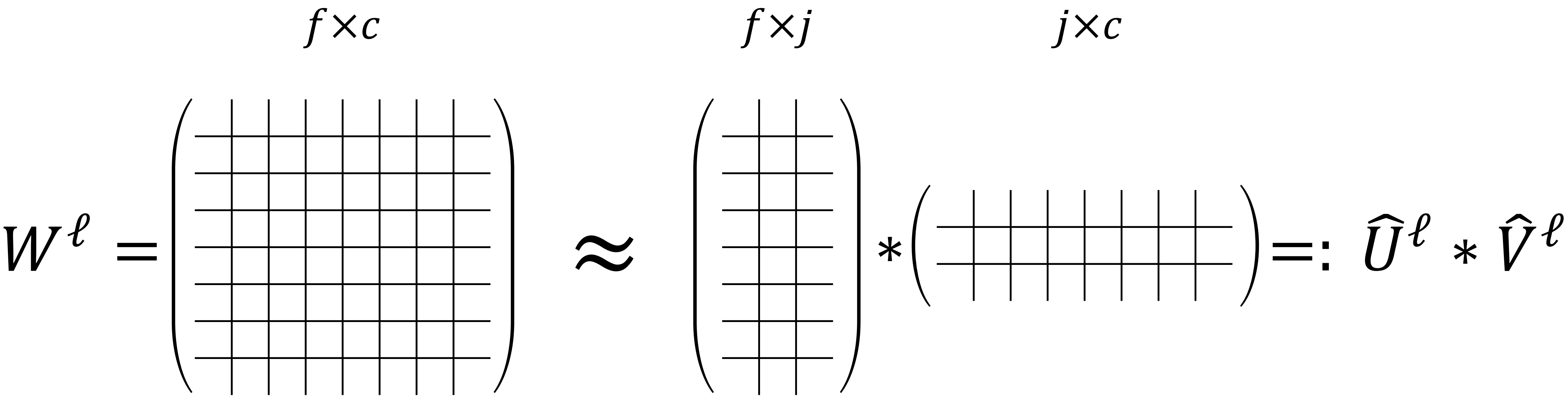
"Global step"



$\|f - \hat{f}\| \leq \varepsilon \, \|f\|$ where $\varepsilon = \varepsilon(\varepsilon^1, \dots, \varepsilon^L)$

$\text{minimize}\; cost(\varepsilon^1, \dots, \varepsilon^L) \;\; s.t. \;\; size(\hat{\theta}) \leq \mathcal{B}$

# Local step: per-layer low-rank compression



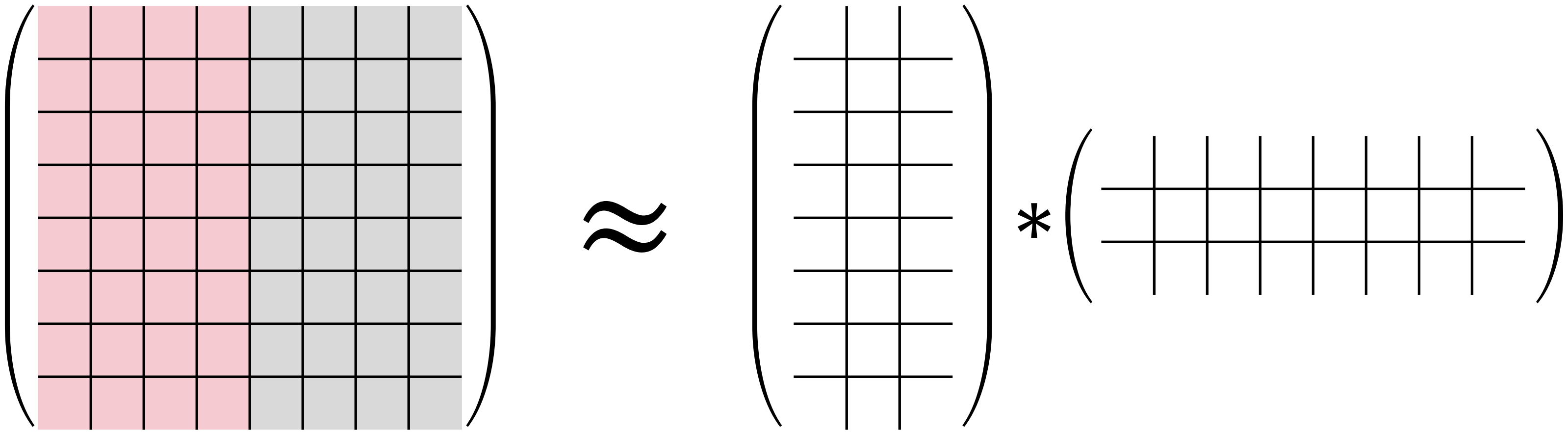

$\#\, params = fc \quad \longrightarrow \quad \#\, params = j(f+c)$

$$\boxed{\varepsilon^\ell = \varepsilon^\ell(j) = \varepsilon^\ell(\text{"compression\_ratio"})}$$

# Local step: per-layer low-rank compression

$f \times \frac{c}{k} k$ $\qquad$ $f \times j$ $\qquad$ $j \times c$

$W^{\ell} =$



$=: \hat{U}^{\ell} * \hat{V}^{\ell}$

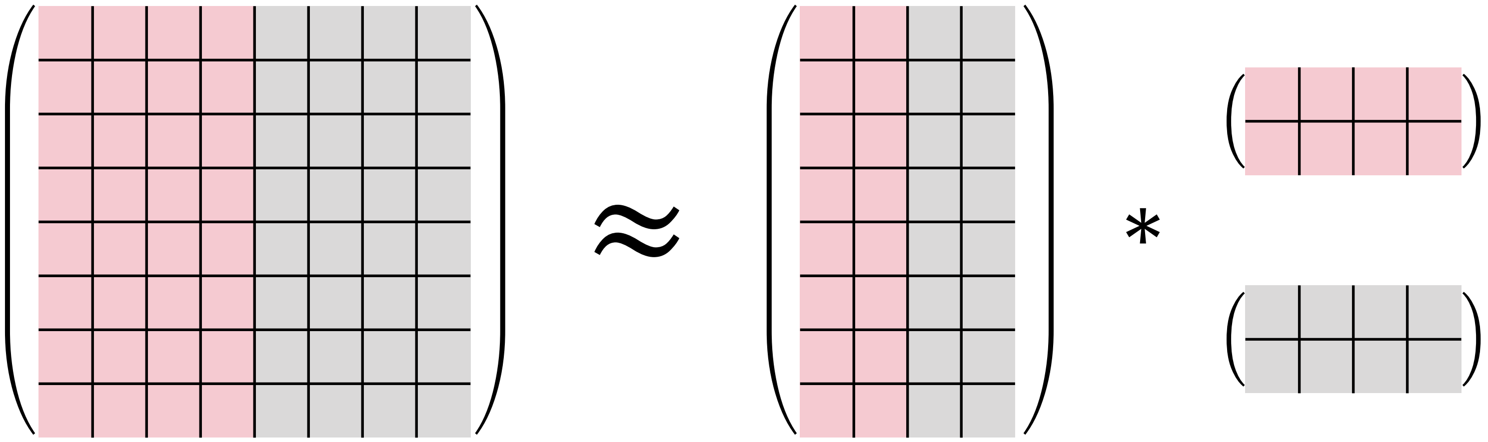
$\# \, params = fc$ ➡ $\# \, params = j(f + c)$

$$\varepsilon^{\ell} = \varepsilon^{\ell}(j) = \varepsilon^{\ell}(\text{"compression\_ratio"})$$

# Local step: per-layer low-rank compression

$f \times \frac{c}{k} k$ $\quad$ $f \times jk$ $\quad$ $k * \left(j \times \frac{c}{k}\right)$

$W^{\ell} =$



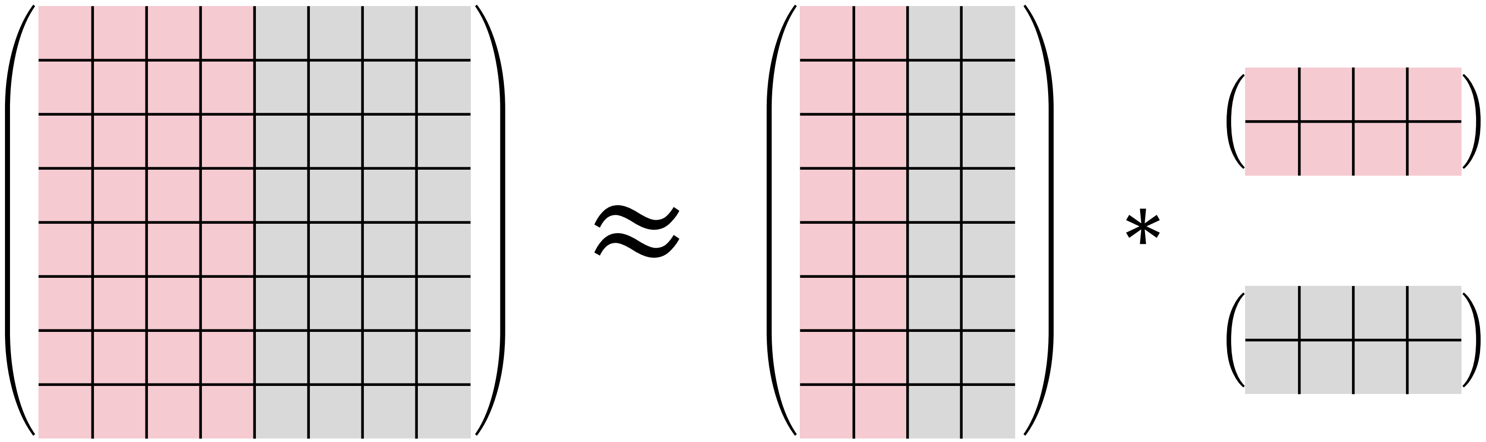
$=: \hat{U}^{\ell} * \hat{V}^{\ell}$

$\# \, params = fc$ ➡ $\# \, params = j(f + c)$

$$\boxed{\varepsilon^{\ell} = \varepsilon^{\ell}(j) = \varepsilon^{\ell}(\text{"compression\_ratio"})}$$

# Local step: per-layer low-rank compression

$f \times \frac{c}{k} k$ $\qquad$ $f \times jk$ $\qquad$ $k * \left( j \times \frac{c}{k} \right)$



$$W^{\ell} = (\ldots) \approx (\ldots) * (\ldots) =: \hat{U}^{\ell} * \hat{V}^{\ell}$$

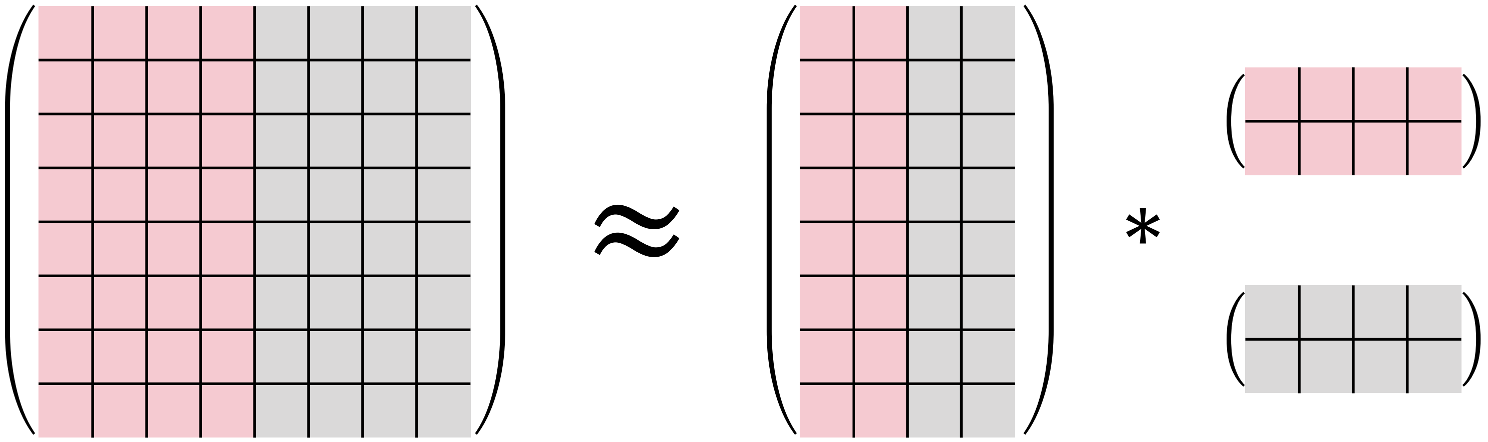
$\#\, params = fc$ $\longrightarrow$ $\#\, params = j(fk + c)$

$$\boxed{\varepsilon^{\ell} = \varepsilon^{\ell}(j) = \varepsilon^{\ell}(\text{"compression\_ratio"})}$$

# Local step: per-layer low-rank compression

$f \times \frac{c}{k} k \qquad f \times jk \qquad k * \left(j \times \frac{c}{k}\right)$

$W^{\ell} =$



$=: \hat{U}^{\ell} * \hat{V}^{\ell}$

$\# \, params = fc \quad \longrightarrow \quad \# \, params = j(fk + c)$

$$\boxed{\varepsilon^{\ell} = \varepsilon^{\ell}(j,k) = \varepsilon^{\ell}(\text{"compression\_ratio"})}$$

# ALDS: Automatic Layer-wise Decomposition Selector

Per-layer budget
$b^1, \dots, b^L$

Randomly initialize
$k^1, \dots, k^L$

$$\min_{j^1,\dots,j^L} \max_{l \in [L]} \varepsilon^\ell(k^\ell, j^\ell)$$

$$s.t.\ size(\hat{\theta}) \leq \mathcal{B}$$

Iterate

Solve for all layers:

$$\min_{j^\ell, k^\ell} \varepsilon^\ell(j^\ell, k^\ell)$$

$$s.t.\ size(\hat{\theta}^\ell) \leq b^\ell$$

Optimal network decomposition

\# subspaces
$k^1, \dots, k^L$

# ALDS: Automatic Layer-wise Decomposition Selector

Randomly initialize
$k^1, \dots, k^L$

$$\min_{j^1,\dots,j^L} \max_{l \in [L]} \varepsilon^\ell(k^\ell, j^\ell)$$
$$s.t.\ size(\hat{\theta}) \leq \mathcal{B}$$

Per-layer budget
$b^1, \dots, b^L$

Iterate

Solve for all layers:

$$\min_{j^\ell, k^\ell} \varepsilon^\ell(j^\ell, k^\ell)$$
$$s.t.\ size(\hat{\theta}^\ell) \leq b^\ell$$

\# subspaces
$k^1, \dots, k^L$

Optimal network decomposition

Global step

# ALDS: Automatic Layer-wise Decomposition Selector

Per-layer budget
$b^1, \dots, b^L$

Randomly initialize
$k^1, \dots, k^L$

$$\min_{j^1,\dots,j^L} \max_{l \in [L]} \varepsilon^\ell(k^\ell, j^\ell)$$
$$s.t.\ size(\hat{\theta}) \leq \mathcal{B}$$

Iterate

Solve for all layers:

$$\min_{j^\ell, k^\ell} \varepsilon^\ell(j^\ell, k^\ell)$$
$$s.t.\ size(\hat{\theta}^\ell) \leq b^\ell$$

Optimal network decomposition

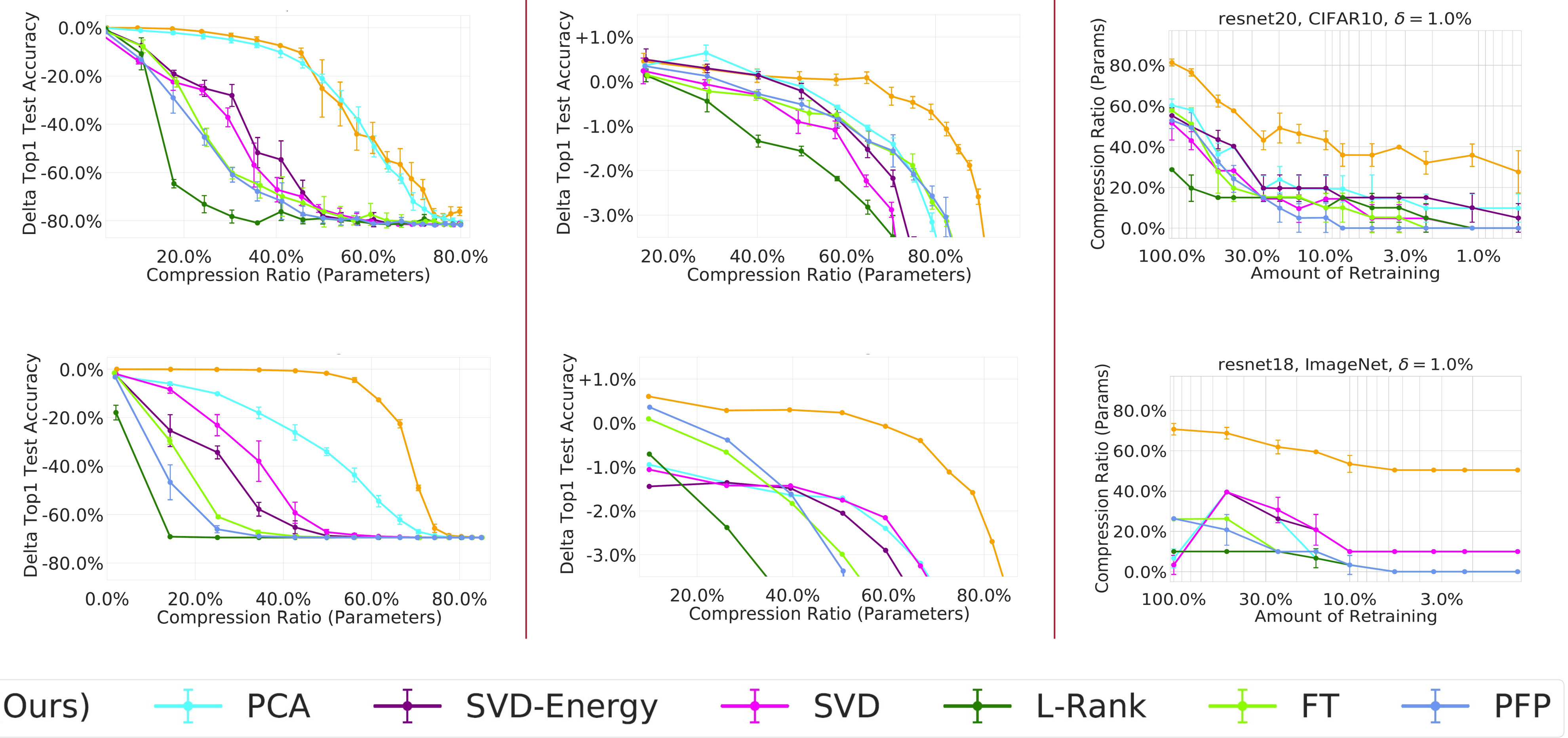
# subspaces
$k^1, \dots, k^L$

Global step

Local step

# Results: one-shot (+ retraining) with baselines

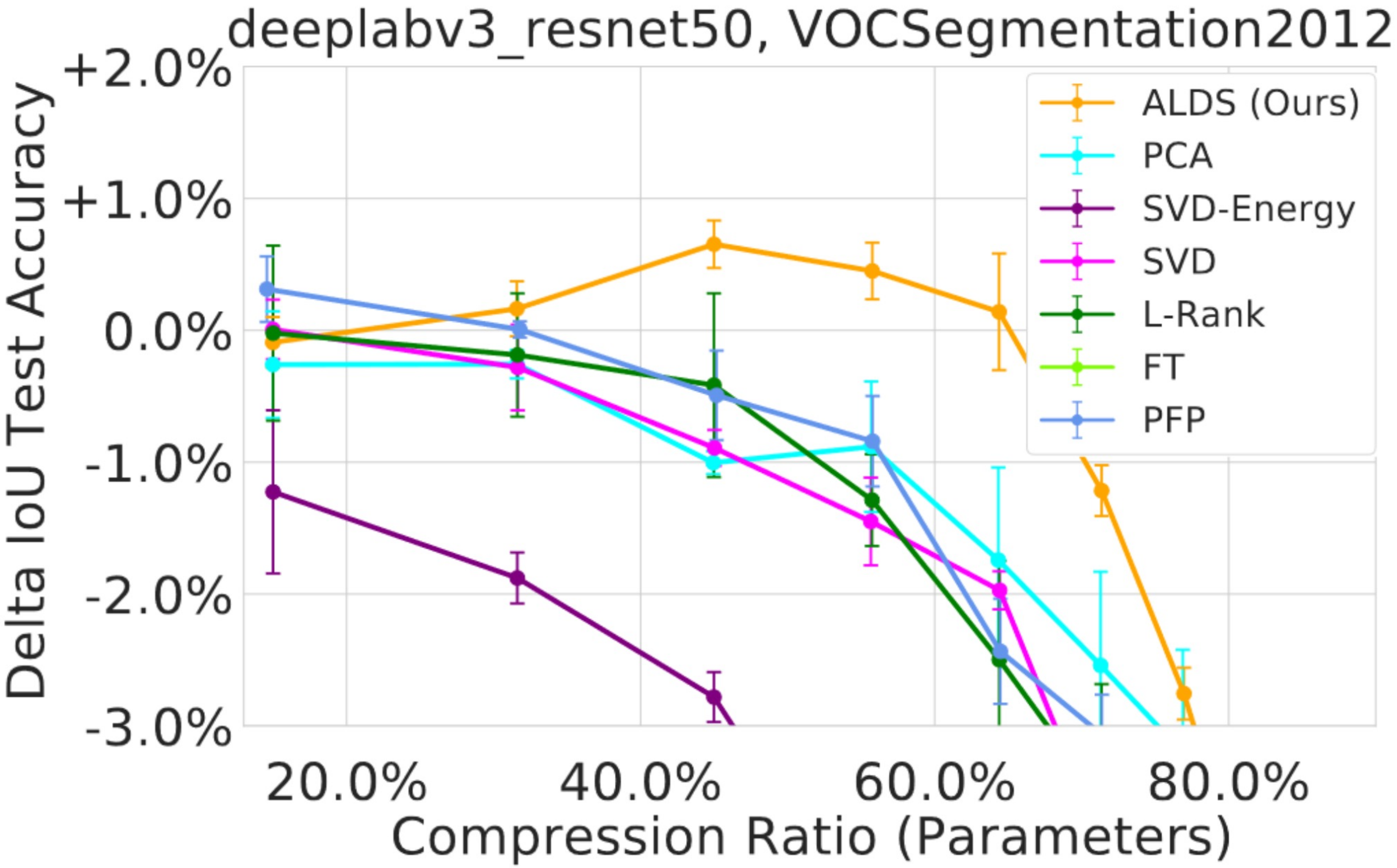
| | Compress-only | Retrain | Sweep |
|---|---|---|---|
| ResNet20 CIFAR10 | | | |
| ResNet18 ImageNet | | | |






lucas@csail.mit.edu

# Results: one-shot (+ retraining) with baselines



# Results: ImageNet benchmarks

| | Method | Δ-Top1 | Δ-Top5 | CR-F (%) |
|---|---|---|---|---|
| ResNet18, Top1, 5: 69.64%, 88.98% | ALDS (Ours) | **-0.38** | **+0.04** | **64.5** |
| | ALDS (Ours) | -1.37 | -0.56 | 76.3 |
| | MUSCO (Gusak et al., 2019) | -0.37 | -0.20 | 58.67 |
| | TRP1 (Xu et al., 2020) | -4.18 | -2.5 | 44.70 |
| | TRP1+Nu (Xu et al., 2020) | -4.25 | -2.61 | 55.15 |
| | TRP2+Nu (Xu et al., 2020) | -4.3 | -2.37 | 68.55 |
| | PCA (Zhang et al., 2015b) | -6.54 | -4.54 | 29.07 |
| | Expand (Jaderberg et al., 2014) | -6.84 | -5.26 | 50.00 |
| | PFP (Liebenwein et al., 2020) | -2.26 | -1.07 | 29.30 |
| | SoftNet (He et al., 2018) | -2.54 | -1.2 | 41.80 |
| | Median (He et al., 2019) | -1.23 | -0.5 | 41.80 |
| | Slimming (Liu et al., 2017) | -1.77 | -1.19 | 28.05 |
| | Low-cost (Dong et al., 2017) | -3.55 | -2.2 | 34.64 |
| | Gating (Hua et al., 2018) | -1.52 | -0.93 | 37.88 |
| | FT (He et al., 2017) | -3.08 | -1.75 | 41.86 |
| | DCP (Zhuang et al., 2018) | -2.19 | -1.28 | 47.08 |
| | FBS (Gao et al., 2018) | -2.44 | -1.36 | 49.49 |

| | Method | Δ-Top1 | Δ-Top5 | CR-F (%) |
|---|---|---|---|---|
| AlexNet, Top1, 5: 57.30%, 80.20% | ALDS (Ours) | -0.21 | -0.36 | 77.9 |
| | ALDS (Ours) | **-0.41** | **-0.54** | **81.4** |
| | Tucker (Kim et al., 2015a) | N/A | -1.87 | 62.40 |
| | Regularize (Tai et al., 2015) | N/A | -0.54 | 74.35 |
| | Coordinate (Wen et al., 2017) | N/A | -0.34 | 62.82 |
| | Efficient (Kim et al., 2019) | -0.7 | -0.3 | 62.40 |
| | L-Rank (Idelbayev et al., 2020) | -0.13 | -0.13 | 66.77 |
| | NISP (Yu et al., 2018) | -1.43 | N/A | 67.94 |
| | OICSR (Li et al., 2019a) | -0.47 | N/A | 53.70 |
| | Oracle (Ding et al., 2019) | -1.13 | -0.67 | 31.97 |

# Discussion and future work

✓ ALDS leads to novel **state-of-the-art results** in low-rank compression

✓ Combining the **local + global** decomposition step leads to a more flexible approach

✓ **Error bounds** lead to better global insights about compression

? **Future:** ALDS as a modular compression framework for any per-layer compression

# Compressing Neural Networks: Towards Determining the Optimal Layer-wise Decomposition

**Lucas Liebenwein*, Alaa Maalouf***, Dan Feldman, Daniela Rus

* Equal contribution

## Thank you

Alaa Maalouf

Dan Feldman

Daniela Rus

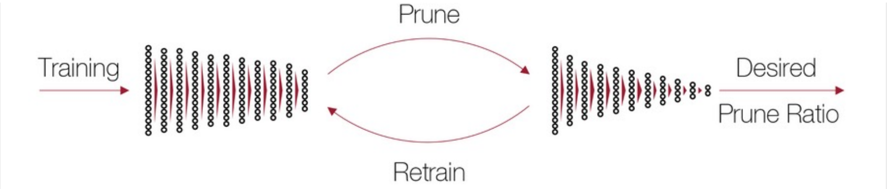
## Acknowledgements

Oren Gal

## Code

**About**

A research library for pytorch-based neural network pruning, compression, and more.




https://github.com/lucaslie/torchprune