Exploiting Elasticity in Tensor Ranks for Compressing Neural Networks

Jie Ran^, Rui Lin, Hayden K.H. So, Graziano Chesi, Ngai Wong

THE UNIVERSITY OF HONG KONG

^ Presenter

Introduction

Low Rank Approximation

Tensor Factorization

- Canonical Polyadic (CP) decomposition
  - ➢ Drawbacks:
    1) Ranks need to be selected manually.
    2) Only works well when one or two layers are compressed.
- Tucker decomposition

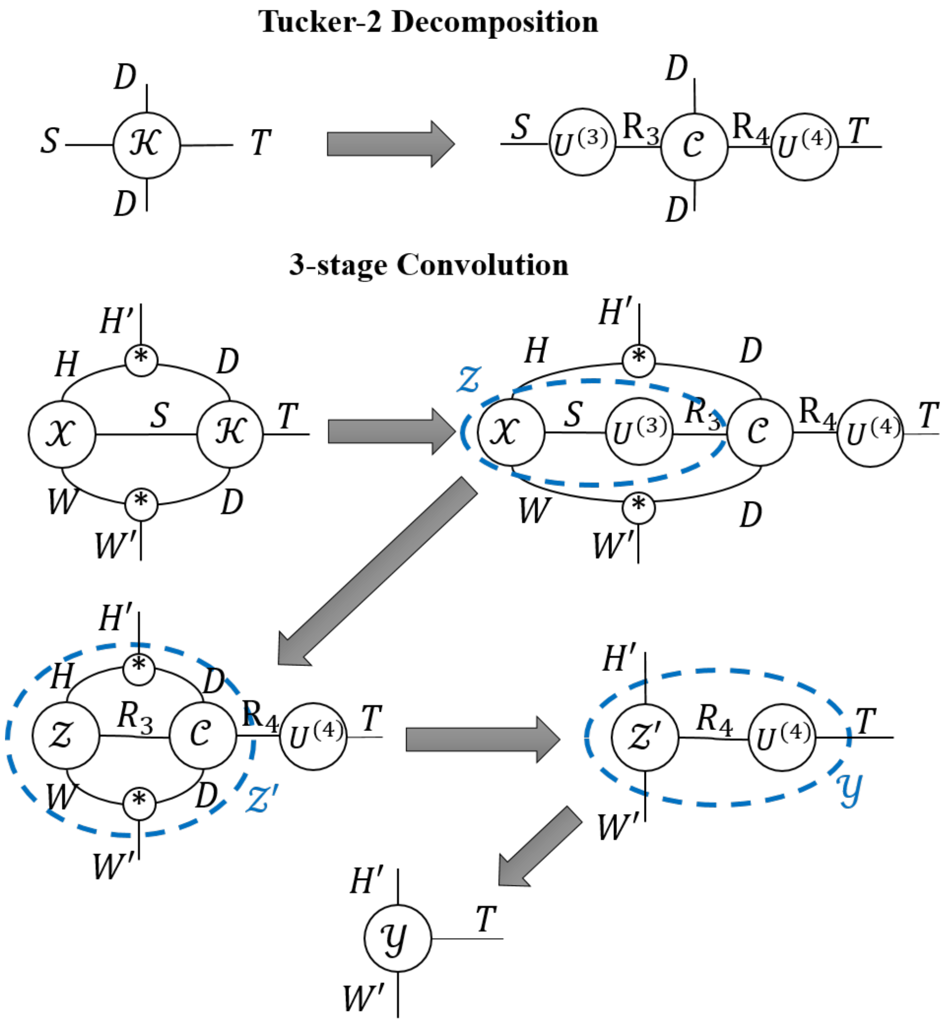
Rank Selection

- Variational Bayesian Matrix Factorization (VBMF)
  - ➢ Drawbacks:
    1) A globally or locally optimal combination of ranks is not guaranteed.
    2) Once the ranks are set, they remain fixed during the fine-tuning stage.

Contributions

- Exploit the elasticity in tensor ranks during training by adding a nuclear-norm-like regularizer to the loss function.
- Analyze variation of ranks in early convolution layers to deeper ones, which could be guidance to remove redundancy in wide layers without much information loss.
- Propose a generic rank selection method which can be applied for low-rank convolutional neural network (CNN) approximation together with other techniques.

Tucker-2 Decomposition



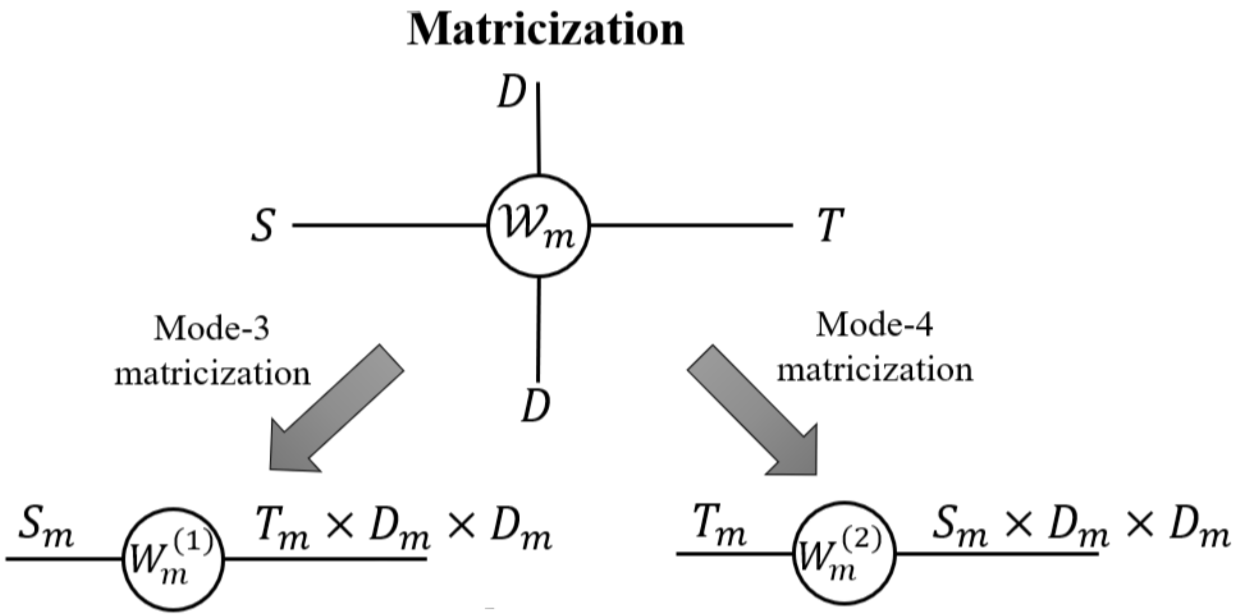
Tucker-2 decomposition on a kernel tensor.

The convolution process after the decomposition.

Key: Find Tucker ranks.

Kernel tensor: $\mathcal{K} \in \mathbb{R}^{S\times T\times D\times D}$
Input feature: $\mathcal{X} \in \mathbb{R}^{H\times W\times S}$
Output feature: $\mathcal{Y} \in \mathbb{R}^{H'\times W'\times T}$
Core tensor: $\mathcal{C} \in \mathbb{R}^{R_3\times R_4\times D\times D}$
Factorization matrix: $U^{(3)} \in \mathbb{R}^{S\times R_3}$, $U^{(4)} \in \mathbb{R}^{T\times R_4}$

Main Idea



- Nuclear-norm-based regularizer:

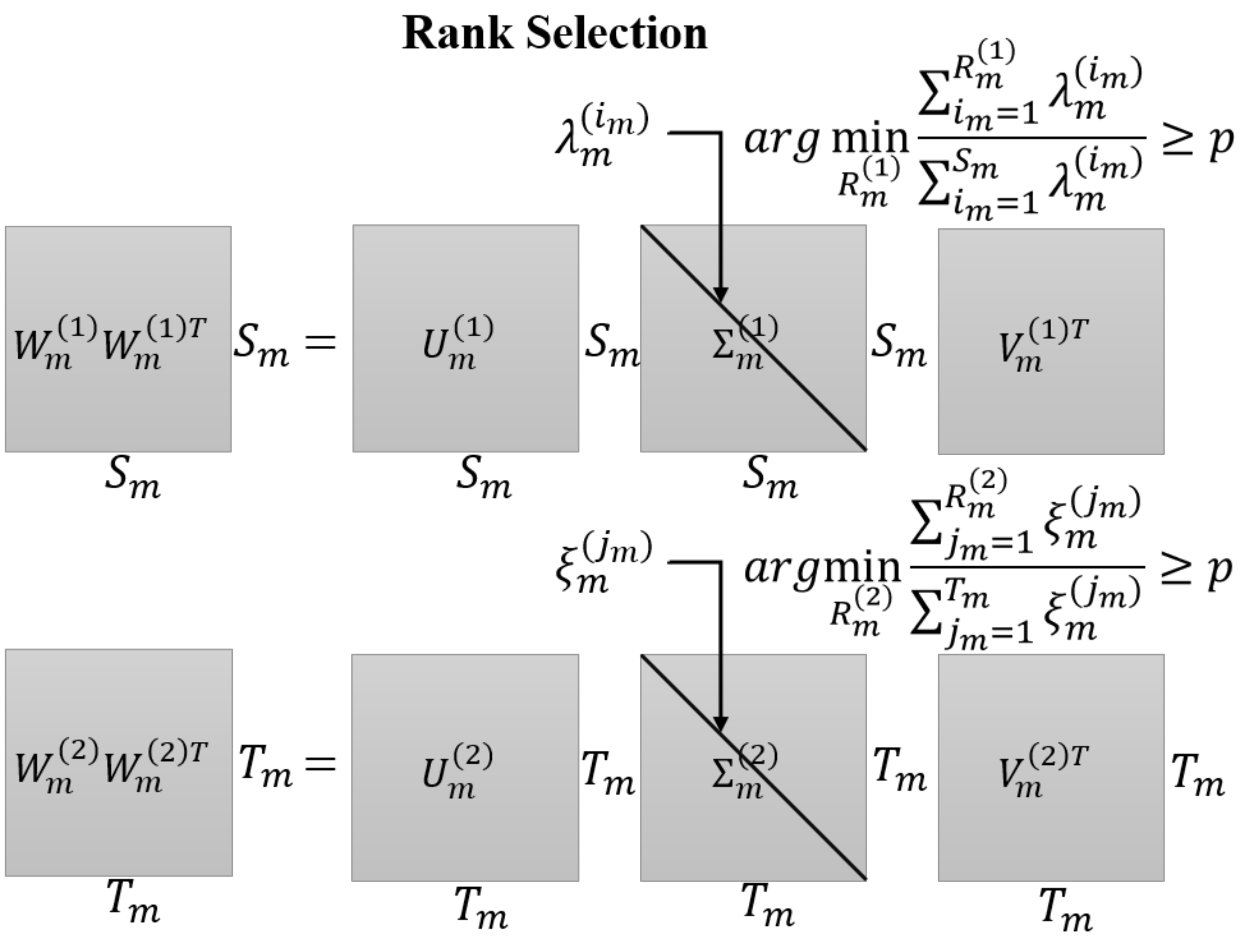
$$L_n = \frac{1}{2M} \sum_{m=1}^{M} (tr(\mathbf{W}_m^{(1)} \mathbf{W}_m^{(1)T}) + tr(\mathbf{W}_m^{(2)} \mathbf{W}_m^{(2)T}))$$

- Given the training dataset $D = \{(\boldsymbol{x}_1, \boldsymbol{y}_1), (\boldsymbol{x}_2, \boldsymbol{y}_2), \cdots, (\boldsymbol{x}_N, \boldsymbol{y}_N)\}$ and weights parameter $\boldsymbol{\mathcal{W}}$ in the network, $\boldsymbol{\mathcal{W}}_{cl} \subseteq \boldsymbol{\mathcal{W}}$ are all convolution layers with kernel size larger than 1×1, α is the scaling factor. Our initialization is determined by:

$$\mathcal{J}(\boldsymbol{\mathcal{W}}) = \frac{1}{N} (\sum_{i=1}^{N} \mathcal{L}((\boldsymbol{x}_i, \boldsymbol{y}_i), \boldsymbol{\mathcal{W}})) + \alpha L_n(\boldsymbol{\mathcal{W}}_{cl}),$$

$$\boldsymbol{w}^* = \arg\min_{\boldsymbol{\mathcal{W}}} \mathcal{J}(\boldsymbol{\mathcal{W}})$$

Main Idea



- Considering sum of the singular values (SVs) in $\boldsymbol{W}_m^{(1)} \boldsymbol{W}_m^{(1)T}$ and $\boldsymbol{W}_m^{(2)} \boldsymbol{W}_m^{(2)T}$ as "energy".
- Retain the leading singular values using a threshold ratio p.
- Define new rank as the minimum number of SVs which preserve a certain percentage of energy.

Experiments

Effect of Regularizer on SVs of the Parameters

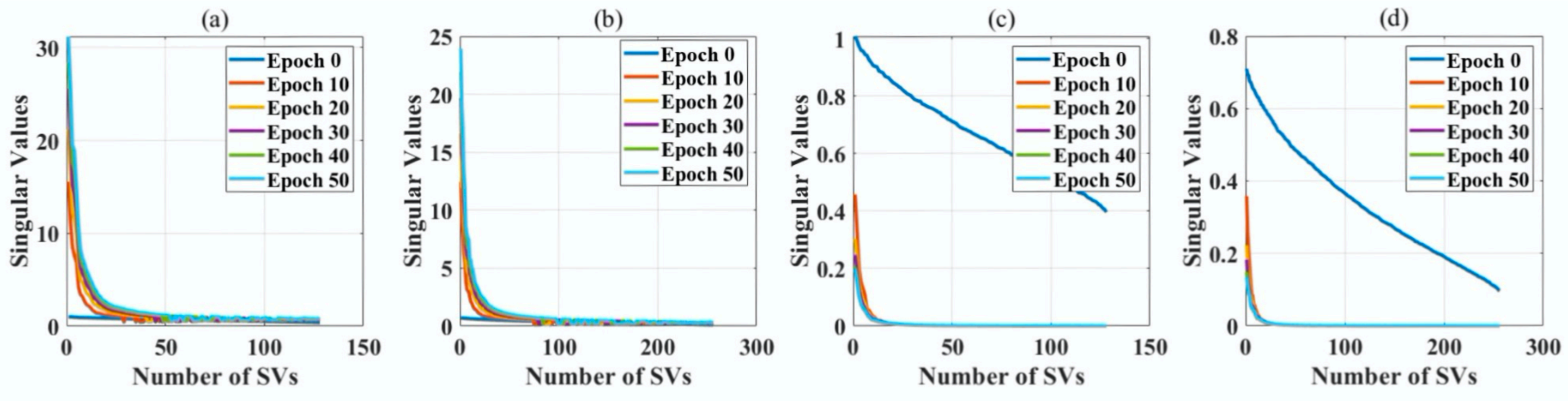


Fig. (a) and (c) show the trends of SVs of $\boldsymbol{W}_m^{(1)}\boldsymbol{W}_m^{(1)T}$ without and with regularizer, respectively. (b) and (d) show the trends of SVs $\boldsymbol{W}_m^{(2)}\boldsymbol{W}_m^{(2)T}$ without and with regularizer, respectively. It is obvious that when the regularizer is included in the training process, SVs keeps decreasing as the number of training epochs increases.

Experiments

Layer-wise Analysis of Compression Ratios

LAYER-WISE ANALYSIS ON RESNET18. S: INPUT CHANNEL DIMENSION, T: OUTPUT CHANNEL DIMENSION, R_3 AND R_4 ARE TUCKER-2 RANKS. $p = 95\%$ TO SELECT RANKS.

Layer	S/R_3	T/R_4	#Parameters
conv1	256	256	$589.82K$
conv1 (VBMF)	168	176	$354.18K(\times 1.67)$
conv1 (NRMF)	144	141	$255.70K(\times 2.31)$
conv2	256	512	$1.18M$
conv2 (VBMF)	194	275	$670.61K(\times 1.76)$
conv2 (NRMF)	222	299	$807.32K(\times 1.46)$
conv3	512	512	$2.36M$
conv3 (VBMF)	332	328	$1.32M(\times 1.79)$
conv3 (NRMF)	292	212	$815.18K(\times 2.89)$
conv4	512	512	$2.36M$
conv4 (VBMF)	348	342	$1.42M(\times 1.66)$
conv4 (NRMF)	160	69	$216.61K(\times 10.89)$
conv5	512	512	$2.36M$
conv5 (VBMF)	382	392	$1.74M(\times 1.35)$
conv5 (NRMF)	31	39	$46.72K(\times 50.50)$

- Show the advantages of dynamic rank search in NRMF over the fixed-rank approach in VBMF
- Reveals the unexploited data redundancy for deeper compression.

Experiments

Performances on Various Datasets and Neural Networks

PERFORMANCE COMPARISON ON CIFAR-10

Model	Rank Selection	Top-1 Accuracy (%)	#Parameters
AlexNet	Baseline	91.85	$57.04M$
	VBMF	91.29	$55.93M$
	NRMF	91.03	$55.05M$
GoogLeNet	Baseline	95.53	$5.61M$
	VBMF	96.18	$4.20M$
	NRMF	95.57	$4.08M$
DenseNet	Baseline	96.56	$6.96M$
	VBMF	95.29	$5.85M$
	NRMF	96.99	$5.85M$

PERFORMANCE COMPARISON ON CIFAR-100

Model	Rank Selection	Top-1 Acc. (%)	Top-5 Acc. (%)	#Parameters
AlexNet	Baseline	71.12	91.75	$57.41M$
	VBMF	69.73	90.51	$56.32M$
	NRMF	68.97	90.06	$55.45M$
GoogLeNet	Baseline	78.96	95.56	$5.70M$
	VBMF	79.50	95.88	$4.27M$
	NRMF	78.93	95.25	$4.14M$
DenseNet	Baseline	81.43	96.30	$7.06M$
	VBMF	82.98	96.13	$5.92M$
	NRMF	83.53	96.70	$5.90M$

PERFORMANCE COMPARISON ON IMAGENET

Model	Rank Selection	Top-1 Acc. (%)	Top-5 Acc. (%)	#Parameters
ResNet18	Base	69.76	89.08	$11.69M$
	VBMF	67.20	87.88	$7.50M$
	NRMF	67.27	87.7	$6.81M$

Thank you!