RUTGERS

Towards Efficient CNN Model Compression with Tensor Decomposition and Optimization

March 6, 2023

miao.yin@rutgers.edu

Yin, Miao, et al. "Towards efficient tensor decomposition-based DNN model compression with optimization framework." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 2021.

Yin, Miao, et al. "HODEC: Towards Efficient High-Order DEcomposed Convolutional Neural Networks." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 2022.

Background

- ❑ **Demands of on-device AI applications grow rapidly**

Source: Qualcomm

- ❑ **Challenges for on-device deployment:**

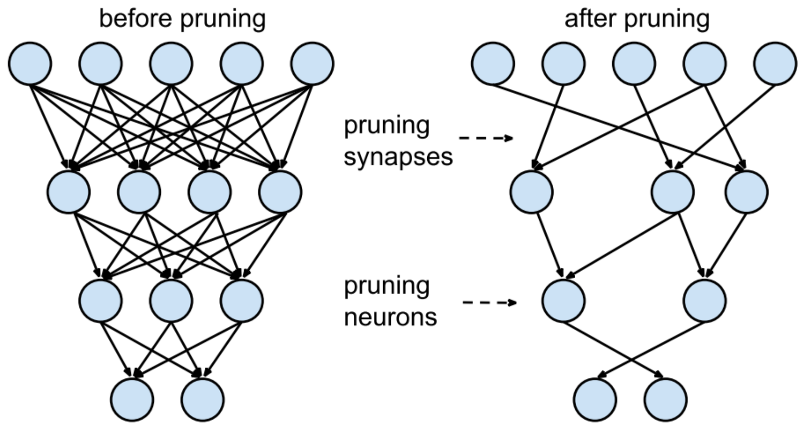
- Cutting-edge DNN models becomes increasingly large, e.g., #param. of ViT-base is **86M**
- Memory and computation resource on IoT devices are limited
- Power consumption problem is severe

Model Compression

Three main directions of model compression techniques to improve the DNN model efficiency and make it executable on edge devices:

- Pruning
  - Unstructured pruning
  - Structured pruning
- Low-rank matrix decomposition
- Higher-order tensor decomposition

Pruning



Structured pruning

Bit Mask

1	0	1
1	0	1
1	0	1

×

Weight

.7	.2	.1
-.2	.8	.9
.2	.1	.3

=

Pruned

.7	0	1
-.2	0	1
.2	0	1

Unstructured pruning

Bit Mask

1	0	1
0	1	1
1	1	0

×

Weight

.7	.2	.1
-.2	.8	.9
.2	.1	.3

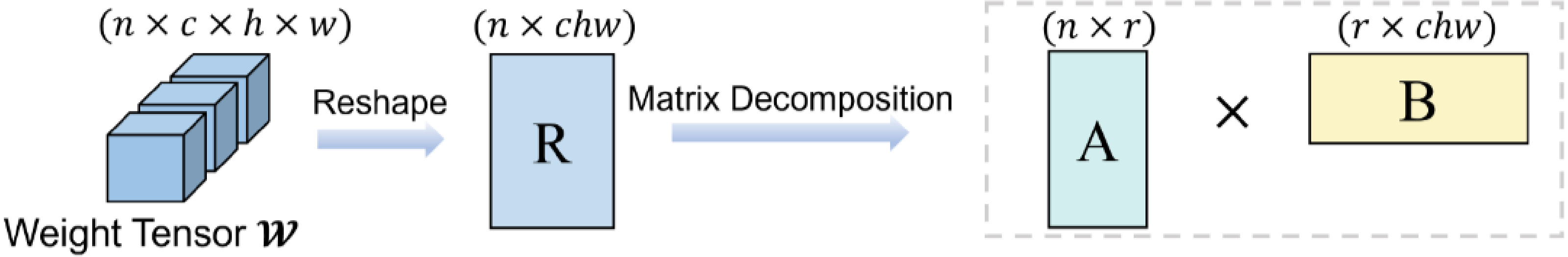
=

Pruned

.7	0	1
0	.8	1
.2	1	0

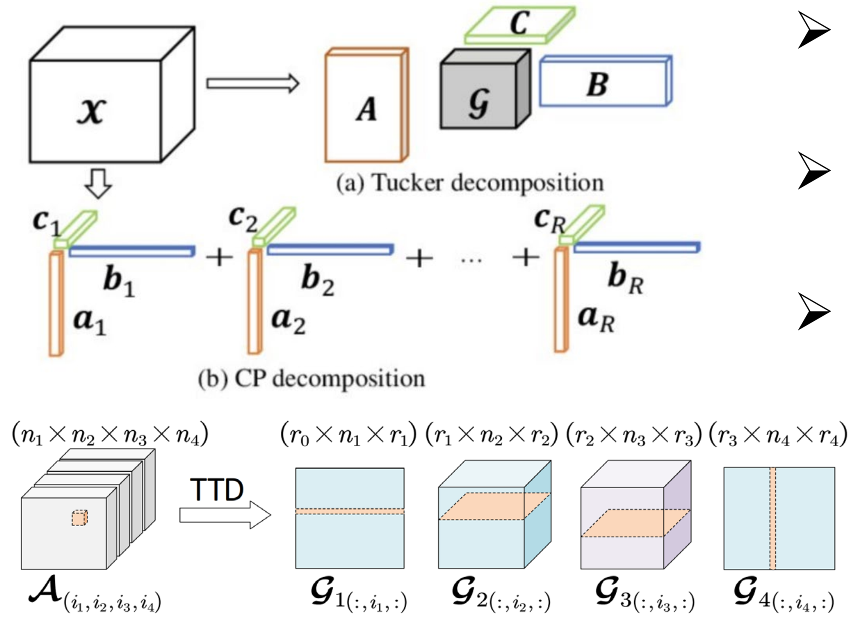
Source: https://towardsdatascience.com/model-compression-via-pruning-ac9b730a7c7b

Matrix Decomposition



- For CNNs, the original 4-D kernel tensor needs to be flattened to a matrix
- Only one-dimensional linear correlation can be leveraged
- The reshape may lead to unbalanced matrix shape, e.g., 64*32*3*3 -> 64*9216, the number of columns is much larger than rows

Higher-Order Tensor Decomposition

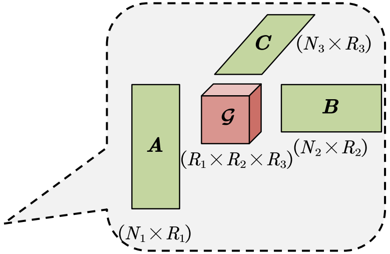


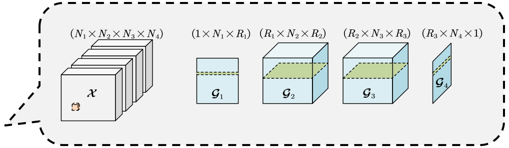
(c) Tensor train decomposition: A mode-4 tensor is decomposed into 4 small tensor cores

- Decompose weight tensors into a series of small TT-cores
- Ultra-high compression ratio, e.g., >1000X for RNNs
- Hardware friendly, parallel memory accessible
  - Multi-dimensional correlation can be leveraged
  - Avoid to reshape to an unbalanced matrix

Comparison among Multiple Tensor Formats

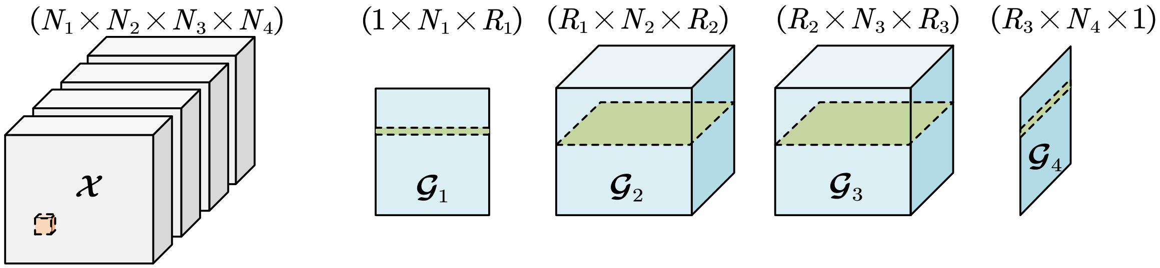
Format	Theoretical Storage	Actual Storage
Tucker-alike (CP, BT)	$\mathcal{O}(R^d + dNR)$	1510 (16x17x18, R=10) 10700 (16x17x18x19, R=10) 100900 (16x17x18x19x20, R=10)
TT-alike (TR, HT)	$\mathcal{O}(dNR^2)$	2040 (16x17x18, R=10) 3850 (16x17x18x19, R=10) 5760 (16x17x18x19x20, R=10)





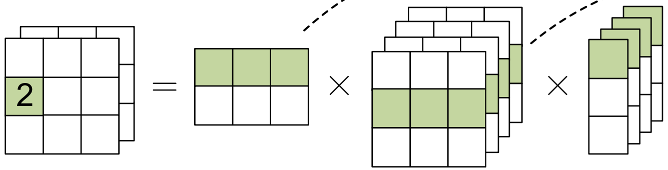
- For a large-scale and low-order tensor, we can increase its order via reshaping. e.g., 512x256x128 -> 32x16x16x16x16x8
- TT-alike format suitable for compressing cutting-edge DNN models with flexibility

Tensor Train (TT) Decomposition



$$\mathcal{X}(n_1, n_2, n_3, n_4) = \mathcal{G}_1(:, n_1, :) \cdot \mathcal{G}_2(:, n_2, :) \cdot \mathcal{G}_3(:, n_3, :) \cdot \mathcal{G}_4(:, n_4, :)$$

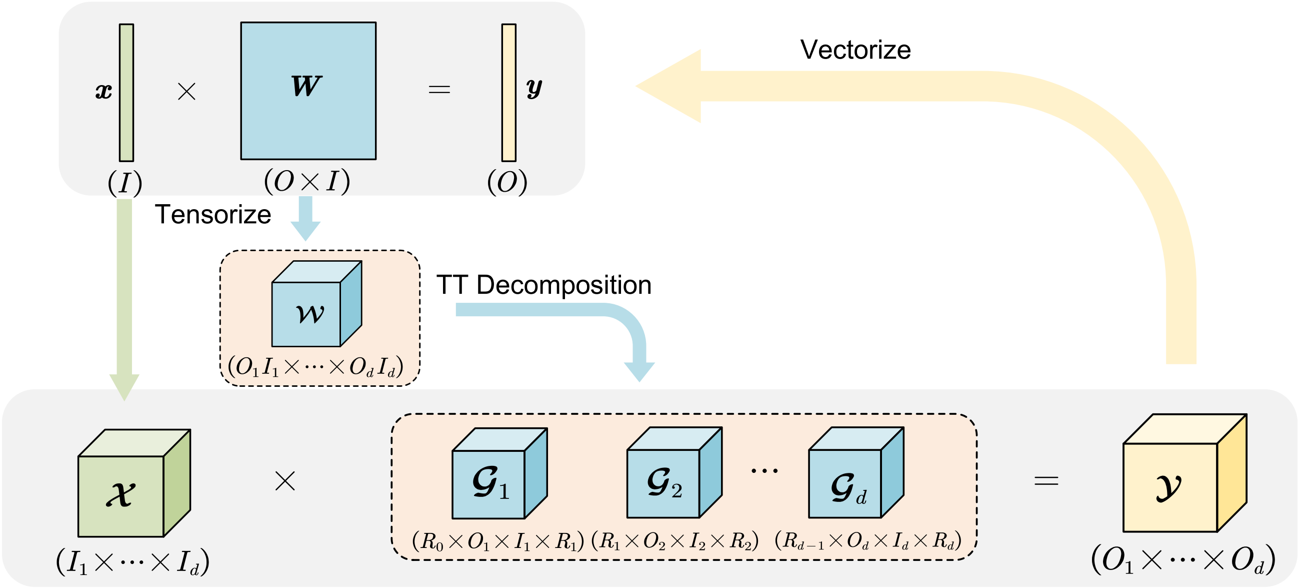
A numeric example for a 3-order tensor (2x3x3):



$$= \begin{bmatrix} 1 & 3 & 2 \end{bmatrix} \times \begin{bmatrix} -2 & -1 & 4 & 3 \\ 2 & 2 & -1 & -2 \\ -1 & -1 & 2 & 3 \end{bmatrix} \times \begin{bmatrix} 3 \\ -2 \\ 1 \\ -1 \end{bmatrix}$$

$$\mathcal{X}(1, 2, 1) = \mathcal{G}_1(:, 1, :) \cdot \mathcal{G}_2(:, 2, :) \cdot \mathcal{G}_3(:, 1, :)$$

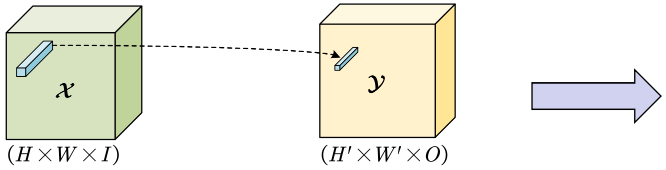
TT-based Fully Connected Layer (TT-FC)



$$\mathcal{Y}(o_1,\cdots,o_d) = \sum_{i_1,\cdots,i_d} \sum_{r_0,\cdots,r_d} \mathcal{X}(i_1,\cdots,i_d)\,\mathcal{G}_1(r_0,o_1,i_1,r_1)\cdots\mathcal{G}_d(r_{d-1},o_d,i_d,r_d)$$

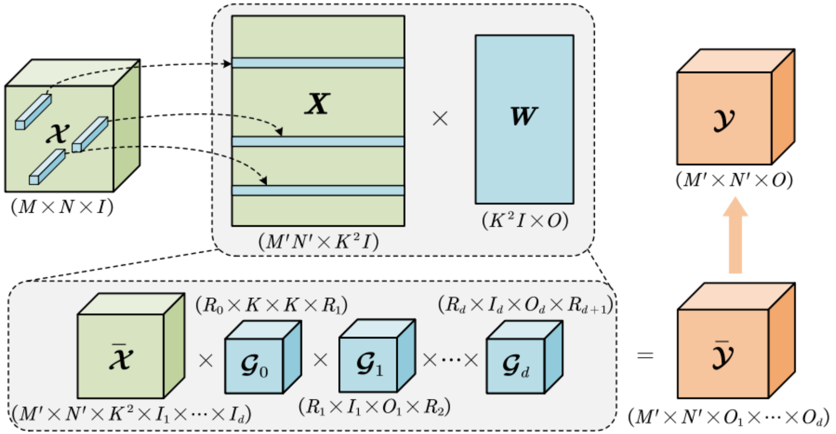
TT-based Convolutional Layer (TT-CONV)

- Original convolution:



$$\mathcal{Y}(h', w', o) = \sum_{k_1}^{K} \sum_{k_2}^{K} \sum_{i}^{I} \mathcal{W}(o, i, k_1, k_2)\, \mathcal{X}(h, w, i)$$

- Classical TT-CONV:



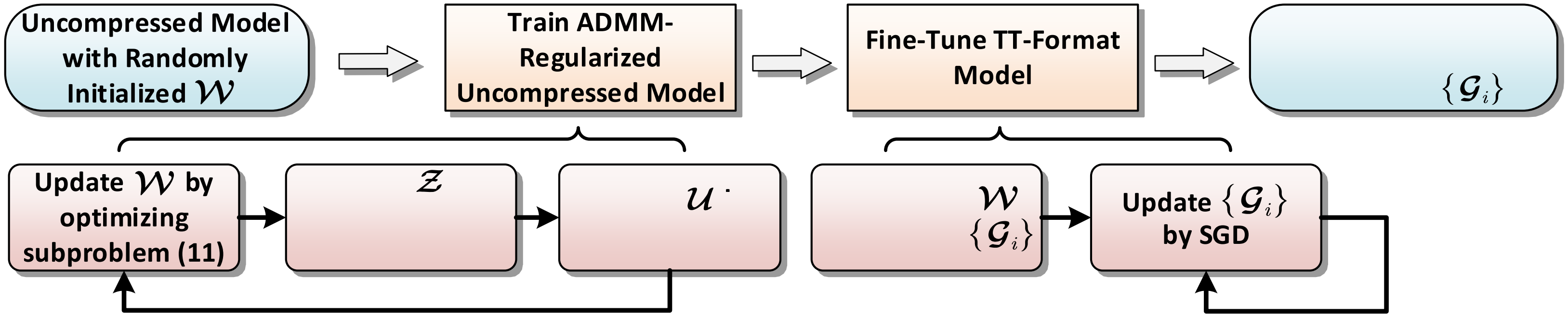
$$\tilde{\mathcal{Y}}(h', w', o_1, \cdots, o_d) = \sum_{k_1}^{K} \sum_{k_2}^{K} \sum_{r_0, \cdots, r_{d+1}} \sum_{i_1, \cdots, i_d} \tilde{\mathcal{X}}(h, w, i_1, \cdots i_d, k_1, k_2)\, \mathcal{G}_0(r_0, k_1, k_2, r_1)\, \mathcal{G}_1(r_1, o_1, i_1, r_2) \cdots \mathcal{G}(r_d, o_d, i_d, r_{d+1})$$

Performance Degradation Problem

- Why limited performance?
  - **Train from randomly initialized tensor cores**: very challenging to train decomposed models due to limited model capacity
  - **Train from the decomposition of a pre-trained uncompressed model**: the pre-trained model is full-rank, decomposing it to TT-format introduces significant approximation error, thereby leading to difficulty to recover the performance

Optimization-Incorporated Training Process

- **Key Idea:** Minimize the approximation error after decomposition
- **Procedure:** Impose low-TT-rank properties on the **uncompressed model** while training → Eliminate the approximation error → Train the TT-format model at an optimal starting point



Overall procedure of the proposed tensor decomposition-based compression framework

Problem Formulation and Solution

- Formulated optimization problem:

$$\min_{\mathcal{W}} \ \ell(\mathcal{W}), \quad \text{s.t.} \ \ \operatorname{rank}(\mathcal{W}) \leqslant \boldsymbol{r}^*$$

- Augmented Lagrangian form:

$$\mathcal{L}_\rho(\mathcal{W},\mathcal{Z},\mathcal{U}) = \ell(\mathcal{W}) + g(\mathcal{Z}) + \frac{\rho}{2}\|\mathcal{W}-\mathcal{Z}+\mathcal{U}\|_F^2 + \frac{\rho}{2}\|\mathcal{U}\|_F^2,$$

- ADMM scheme:

$$\mathcal{W}^{t+1} = \underset{\mathcal{W}}{\operatorname{argmin}} \ \mathcal{L}_\rho\left(\mathcal{W},\mathcal{Z}^t,\mathcal{U}^t\right),$$

$$\mathcal{Z}^{t+1} = \underset{\mathcal{Z}}{\operatorname{argmin}} \ \mathcal{L}_\rho\left(\mathcal{W}^{t+1},\mathcal{Z},\mathcal{U}^t\right),$$

$$\mathcal{U}^{t+1} = \mathcal{U}^t + \mathcal{W}^{t+1} - \mathcal{Z}^{t+1},$$

- W-update:

$$\mathcal{W}^{t+1} = \mathcal{W}^t - \eta \frac{\partial \mathcal{L}_\rho(\mathcal{W},\mathcal{Z}^t,\mathcal{U}^t)}{\partial \mathcal{W}},$$

- Z-update:

$$\mathcal{Z}^{t+1} = \boldsymbol{\Pi}_{\mathcal{S}}(\mathcal{W}^{t+1} + \mathcal{U}^t),$$

- **Overall algorithm:**

Algorithm 2 ADMM-Regularized Training Procedure

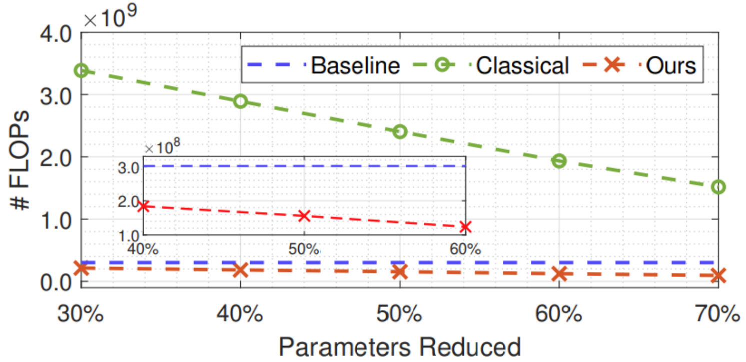
Input: Weight tensor $\mathcal{W}$, target TT-ranks $\boldsymbol{r}^*$, penalty parameter ρ, feasibility tolerance ϵ, maximum iterations T.

Output: Optimized $\mathcal{W}$.

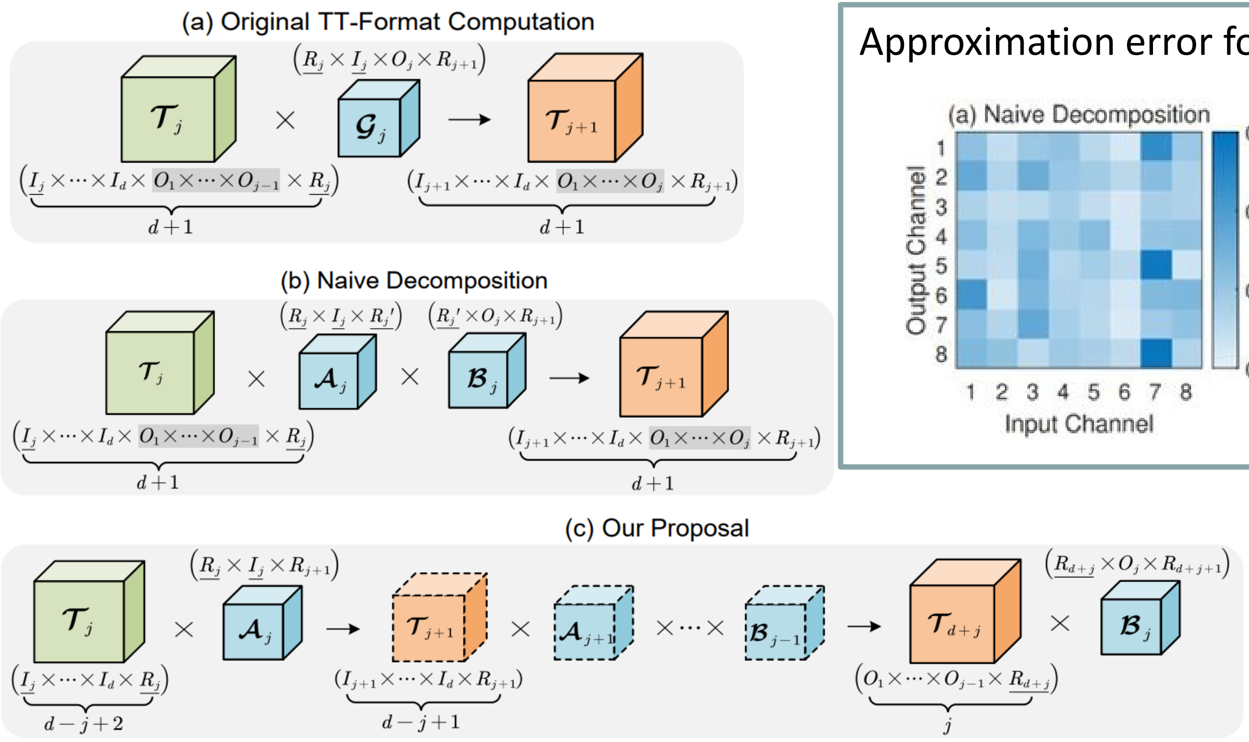
Randomly initialize $\mathcal{W}$;
$\mathcal{Z} := \mathcal{W}$, $\mathcal{U} := \mathbf{0}$;
while $\|\mathcal{W}^t - \mathcal{Z}^t\| > \epsilon$ and $t \leqslant T$ **do**
  Updating $\mathcal{W}$ via (16);
  Updating $\mathcal{Z}$ via (17) (Algorithm 1);
  Updating $\mathcal{U}$ via (13);
end

Limitations of Existing TT-based Compression

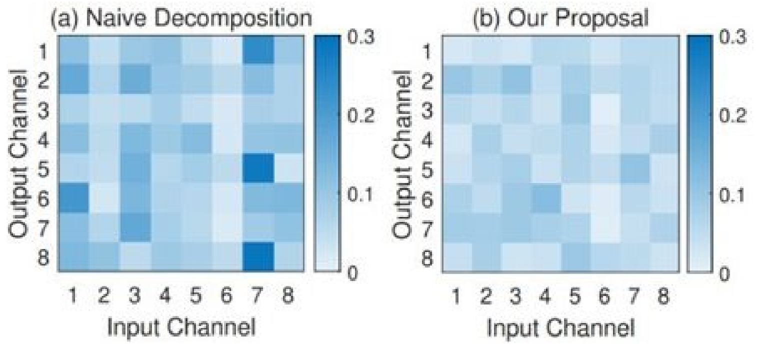
- **Significant performance degradation** for CNNs
  - E.g., on CIFAR-10 dataset for ResNet-32 with 4.2x parameters reduction, **4.2% accuracy drop** [Wang, Wenqi, et al. "Wide compression: Tensor ring nets." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018.]
- **Unbalanced FLOPs and Parameters reduction**
  - E.g., FLOPs vs Parameter reduction for layer3.0.conv1 in ResNet-18 when using conventional TT-CONV. It is seen that conventional TT-CONV ("Classical") causes even **higher FLOPs consumption** than the uncompressed one ("Baseline")



Analysis for Unbalanced FLOPs and Parameters



Approximation error for one layer in ResNet-32:



FLOPs

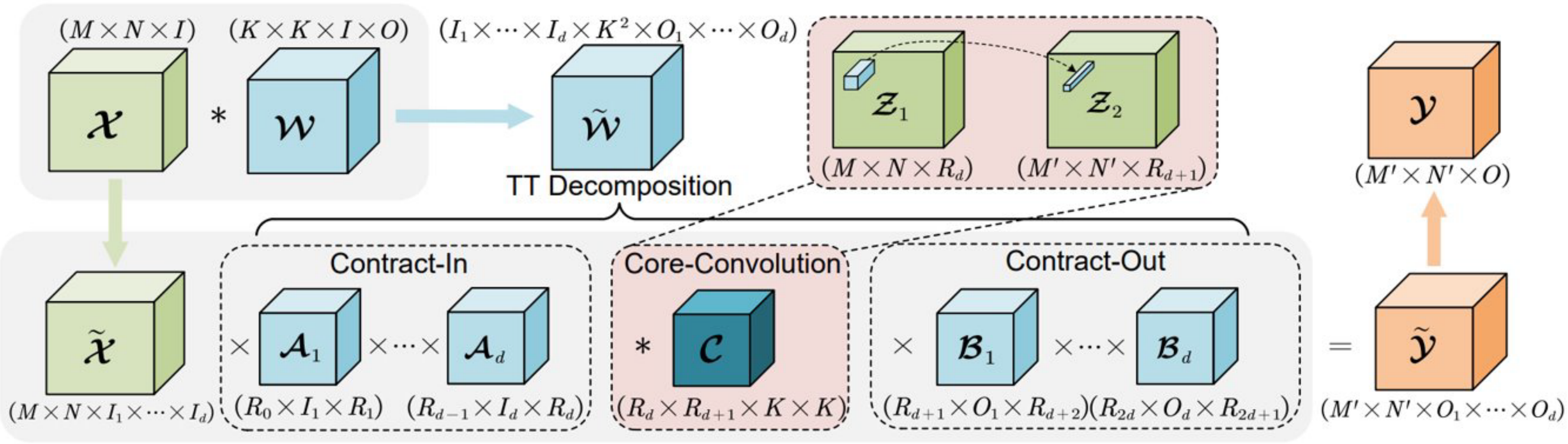
(a) $\mathcal{O}(I_m{}^{d-j+1}O_m^j R^2 M'N')$

(b) $\mathcal{O}(2I_m{}^{d-j}O_m^j R^2 M'N')$

(c) $\mathcal{O}((I_m{}^{d-j+1}R^2 + O_m^j R^2)M'N')$

RUTGERS

Proposed Efficient TT-CONV Scheme



RUTGERS

The Overall Compression Framework

Algorithm 1 The overall computation scheme of HODEC

Input: TT-cores $\{\mathcal{A}_j\}_{j=1}^{d}$, $\{\mathcal{B}_j\}_{j=1}^{d}$ and $\mathcal{C}$, input tensor $\mathcal{X}$, factorized input channels $[I_1, \cdots, I_d]$;

Output: Output tensor $\mathcal{Y}$;

$\tilde{\mathcal{X}} \leftarrow$ RESHAPE($\mathcal{X}$, $[M, N, I_1, \cdots, I_d]$);

$\mathcal{Z}_1 \leftarrow \tilde{\mathcal{X}}$;

for $j = 1$ to d **do** ▷ *Contract-In*

  $\mathcal{Z}_1 \leftarrow$ TENSORCONTRACT($\mathcal{Z}_1$, $\mathcal{A}_j$);

end for

$\mathcal{Z}_2 \leftarrow$ CONV2D($\mathcal{Z}_1$, $\mathcal{C}$); ▷ *Core-Convolution*

for $j = 1$ to d **do** ▷ *Contract-Out*

  $\mathcal{Z}_2 \leftarrow$ TENSORCONTRACT($\mathcal{Z}_2$, $\mathcal{B}_j$);

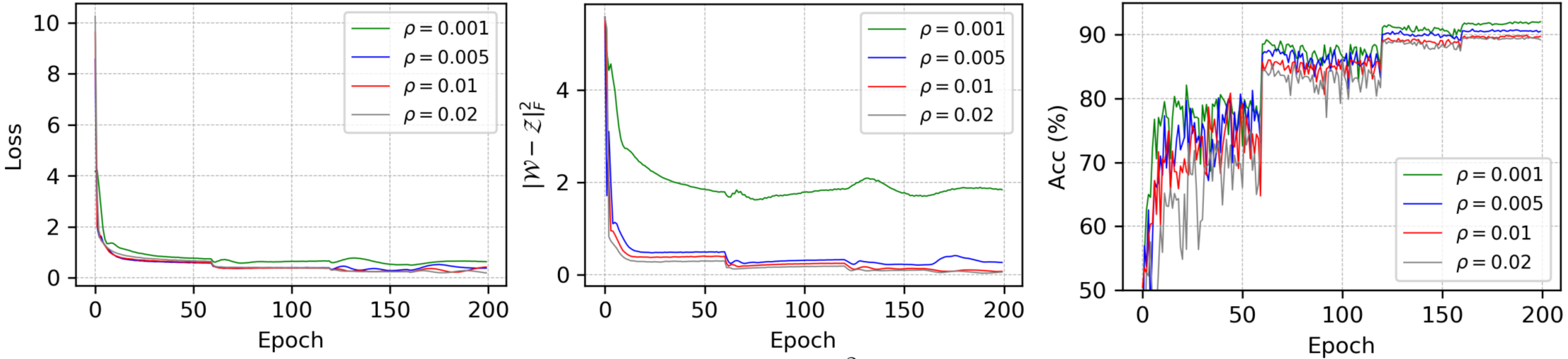
end for

$\mathcal{Y} \leftarrow$ RESHAPE($\mathcal{Z}_2$, $[M', N', O]$).

Algorithm 2 Pseudo-code for high-accuracy training

```
def training(w, tt_shapes, tt_ranks, tau,
dense_epochs, tt_epochs):
  # Train original model with ADAL
  train_dense(w, tau, dense_epochs)
  # Decompose to TT-format
  tt_cores = dense_to_tt(w)
  # Retrain compressed TT-format model
  train_tt(tt_cores, tt_epochs)
def train_dense(w, tau, epochs):
  u, v = zeros(w.shape), Tensor(w)
  for e in range(epochs):
    x, y = sample_data()
    y_ = model_predict(w, x)
    loss = cross_entropy(y, y_)
    v = truncate_tt_ranks(w + u)
    loss += tau * norm(w - v + u, p=2)
    loss.backward()
    u += w - v
```

Training Curves



(a) Training loss. (b) $\|\mathcal{W} - \mathcal{Z}\|_F^2$. (c) Top-1 test accuracy.

(a) Training loss, (b) Frobenius norm and (c) test accuracy in ADMM-regularized training procedure with different ρ for ResNet-32 on CIFAR-10 dataset.

> (b) can be considered as the initial approximation error. It is seen that our method can significantly reduce the approximation error during training.

Experimental Results on CIFAR-10

Model	Compression Method	Top-1 Acc. (%)			FLOPs↓	Params.↓
		Baseline	Compressed	Δ		
ResNet-32						
Rethinking [20]	Pruning	N/A	92.56	N/A	30%	30%
FPGM [12]	Pruning	92.63	92.82	+0.19	53%	N/A
SCOP	Pruning	92.66	92.13	-0.53	56%	56%
Wide [34]	Tensor Ring	92.49	90.30	-2.19	N/A	80%
Ultimate [7, 24]	Classical TT	92.49	88.30	-4.19	×	80%
HODEC (Ours)	Proposed TT	92.49	91.28	-1.21	**72%**	**80%**
HODEC (Ours)	Proposed TT	92.49	**93.05**	**+0.56**	**60%**	65%
ResNet-56						
HRank [19]	Pruning	93.26	93.17	-0.09	50%	42%
SCOP [32]	Pruning	93.70	93.64	-0.06	56%	56%
NPPM [6]	Pruning	93.04	93.40	+0.36	50%	N/A
CHIP [31]	Pruning	93.26	94.16	+0.75	47%	43%
TRP [36]	Low-rank Matrix	93.14	92.63	-0.51	60%	N/A
CC [17]	Low-rank Matrix	93.33	93.64	+0.31	52%	48%
Ultimate [7, 24]	Classical TT	93.04	91.14	-1.90	×	50%
HODEC (Ours)	Proposed TT	93.04	**94.20**	**+1.16**	**62%**	**67%**

Table 1. Performance comparison for compressing CNN models on CIFAR-10 dataset.

Experimental Results on ImageNet

Model	Compression Method	Top-1 Acc. (%)			Top-5 Acc. (%)			FLOPs↓
		Baseline	Compr.	Δ	Baseline	Compr.	Δ	
				ResNet-18				
FPGM [12]	Pruning	70.28	68.41	-1.87	89.63	88.48	-1.15	42%
SCOP [32]	Pruning	69.76	68.62	-1.14	89.08	88.45	-0.63	45%
TRP [36]	Low-rank Matrix	69.10	65.51	-3.59	88.94	86.74	-2.20	60%
Stable [27]	Tucker-CP	69.76	69.07	-0.69	89.08	88.93	-0.15	67%
HODEC (Ours)	Proposed TT	69.76	**69.15**	**-0.61**	89.08	**88.99**	**-0.09**	**68%**
				ResNet-50				
FPGM [12]	Pruning	76.15	75.59	-0.56	92.87	92.63	-0.24	42%
HRank [19]	Pruning	76.15	74.98	-1.17	92.87	92.33	-0.54	44%
SCOP [32]	Pruning	76.15	75.26	-0.89	92.87	92.53	-0.34	55%
NPPM [6]	Pruning	76.15	75.96	-0.19	92.87	92.75	-0.12	56%
CHIP [31]	Pruning	76.15	76.15	0.00	92.87	92.91	+0.04	49%
TRP [36]	Low-rank Matrix	75.90	74.06	-1.84	92.70	92.07	-0.63	45%
CC [17]	Low-rank Matrix	76.15	75.59	-0.56	92.87	92.64	-0.23	53%
Stable [27]	Tucker-CP	76.13	74.66	-1.47	92.87	92.16	-0.71	62%
HODEC (Ours)	Proposed TT	76.13	**76.44**	**+0.31**	92.87	**93.16**	**+0.29**	**63%**

Table 2. Performance comparison for compressing CNN models on ImageNet dataset.

RUTGERS

Thanks!