2025 International Conference on Intelligent Computing
July 26-29, Ningbo, China
https://www.ic-icc.cn/2025/index.php

RanpCode: Rank-based Pruning after Complete CP Decomposition for Model Compression

Lianhua Yu[1[0009-0002-5048-5344]], Guodong Zou[2[0009-0008-0311-3656]], Guoming Lu[3[0000-0001-7477-5800]*1], Jielei Wang[4[0000-0003-2882-7053]], KexinLi[5[0000-0003-3511-2582]], and Guangchun Luo[6[0000-0001-7330-2139]]

University of Electronic Science and Technology of China, No. 2006, Xiyuan Avenue, West Hi-Tech Zone, Chengdu 611731, Sichuan, China
202312081635@std.uestc.edu.cn, 202321080217@std.uestc.edu.cn, lugm@uestc.edu.cn*, jieleiwang@uestc.edu.cn, 956487315@qq.com, gcluo@uestc.edu.cn

Abstract. Traditional convolutional neural networks (CNNs) architecture has limitations such as over-parameterization and a large computational demand. One effective approach to address these issues is to replace convolutional kernels with their low-rank tensor approximations. Among the various methods available, Canonical Polyadic (CP) tensor decomposition stands out as particularly promising. However, employing CP decomposition for CNNs compression presents two major challenges. First, numerical optimization algorithms used to fit convolutional tensors can cause rank-one tensors to cancel each other out, leading to instability and complicating the fine-tuning of the resulting model. Second, determining the appropriate rank for CP decomposition is inherently complex. To overcome these challenges, we propose RanpCode, a novel compression method based on CP decomposition. This method incorporates specially designed numerical fitting techniques to ensure complete CP decomposition and address instability issues. Furthermore, it employs a rank pruning scheme to automatically determine the optimal rank for each layer, with the rank globally optimized and adjusted according to the desired compression rate. Our evaluations on popular CNNs architectures for image classification demonstrate that RanpCode achieves higher compression rates while maintaining superior accuracy.

Keywords: model compression, convolutional neural networks, CP decomposition, instability, rank selection.

[1] *Corresponding Author

1 Introduction

Recent years people have witnessed the widespread application of convolutional neural networks (CNNs) in various fields, including image detection [12], semantic segmentation [11] and object recognition [26], achieving significant progress and breakthroughs. However, deep convolutional networks are currently facing a serious issue of parameter redundancy. To address the parameter redundancy problem in deep convolutional networks, researchers have proposed various methods to compress and accelerate large intelligent models, such as pruning [7], quantization [6] and knowledge distillation [31]. In addition to these, low-rank decomposition methods based on tensor decomposition [9, 15], which explore the low-rank structure of tensors in convolutional layers, use combinations of low-rank tensors to approximate the original tensors, and have demonstrated excellent performance, garnering increasing attention. Examples include Tensor Ring methods [24] and Tucker Decomposition methods [15], which have achieved better results in recent years. Despite being one of the earliest tensor decomposition methods applied to model compression, Canonical Polyadic (CP) decomposition [17], with its straightforward and efficient decomposition form, has not achieved satisfactory results in recent years compared to other tensor decomposition techniques. The primary issues contributing to this are CP decomposition instability and rank selection.

CP Decomposition Instability: The inefficiency of CP decomposition in compressing CNNs is partly due to CP instability, which often impairs fine-tuning after decomposition [27]. This issue has been addressed in various studies [22, 27], leading to some effective solutions. For instance, the Tensor Power Method (TPM) proposed in 2017 [1] shows effectiveness by individually fitting rank-one tensors, which helps mitigate the severe rank-one tensor cancellation that is a fundamental cause of CP instability. However, this method requires that the original tensor have orthogonal decomposability; otherwise, it cannot effectively decompose the tensor when the rank is small.

Rank Selection: Rank selection remains a significant challenge in tensor decomposition-based compression methods, as it is crucial for balancing the model compression ratio and accuracy loss. Research has shown that many tensor analogues of efficiently computable problems in numerical linear algebra are NP-hard [23], including the determination of the rank of a third-order tensor. While some tensor decomposition methods, such as Tucker-based approaches [29], have developed effective automatic rank selection techniques BC-ADL with promising experimental results, CP decomposition lacks sufficiently effective automatic rank selection schemes. Methods like the Variational Bayesian Matrix Factorization analysis [2] and the heuristic binary search method [22], either incur high computational costs or fail to provide the globally optimal rank combination, leading to suboptimal outcomes.

To effectively tackle these challenges, we propose RanpCode, conducting rank-based pruning after complete CP decomposition, a robust compression technique with automatic rank selection, which is adjusted according to the target compression rate. Specifically, we begin by conducting a complete CP decomposition, ensuring that the number of parameters remains unchanged before and after the decomposition in TPM.

This approach yields a relatively large initial rank and avoids the orthogonality constraints often assumed in TPM. Following the complete decomposition, we evaluate the significance of each rank-one tensor and selectively prune the least impactful ones. This rank-based pruning scheme systematically reduces the rank of each layer while minimizing the loss of important information. To sum up, the contributions of this paper are summarized as follows:

- Based on the TPM, we propose the complete CP decomposition method, establishing a relatively large initial rank and creating the foundation for subsequent pruning.
- Interpreting rank selection as model pruning, we propose rank-based pruning scheme with automatic rank selection.
- We design and execute validation experiments on classic ResNet architectures using the CIFAR and ImageNet datasets and the experimental results demonstrate that our method delivers competitive performance.

2 Related Work

2.1 Low-rank Decompositions

The research on low-rank structures for neural network compression has been extensively explored. Typically, more compact convolutional neural networks (CNNs) can be obtained by applying low-rank matrix or tensor decomposition techniques to the original, larger models. In matrix factorization-based methods [4, 5, 14], all weight tensors, including the 4D tensors in convolutional layers, are flattened into 2D matrices and then decomposed into smaller matrices. In contrast, tensor decomposition-based methods directly factorize high-order tensor objectives into a series of tensor and matrix cores. These approaches include various decomposition techniques [10, 21, 22, 25, 27-30], such as CP, Tucker, and Tensor Train (TT), among others. Regardless of the specific decomposition method employed, a key challenge lies in the effective determination of rank. Currently, most existing methods rely on manual trials and experimentation to determine the rank for each layer, a process that requires significant engineering effort and still leaves considerable room for improvement in the final results.

2.2 Automatic Rank Selection

Recent studies have increasingly focused on the efficient determination of tensor ranks. In [10], the authors propose a variational Bayesian matrix factorization approach for determining the ranks of tensor-decomposed deep neural networks (DNNs) in a multi-stage fashion. However, the inherent nature of the Bayesian approach necessitates the introduction of additional auxiliary hyperparameters during the search process, which can complicate the procedure. In [18], a genetic algorithm-based search strategy is introduced to progressively identify tensor ranks for tensor decomposition (TR) based

DNNs models. Drawing inspiration from neural architecture search, this method iteratively explores the model space through a set of predefined rules. Nevertheless, the overall search process remains highly time-consuming. In [29], the authors propose an

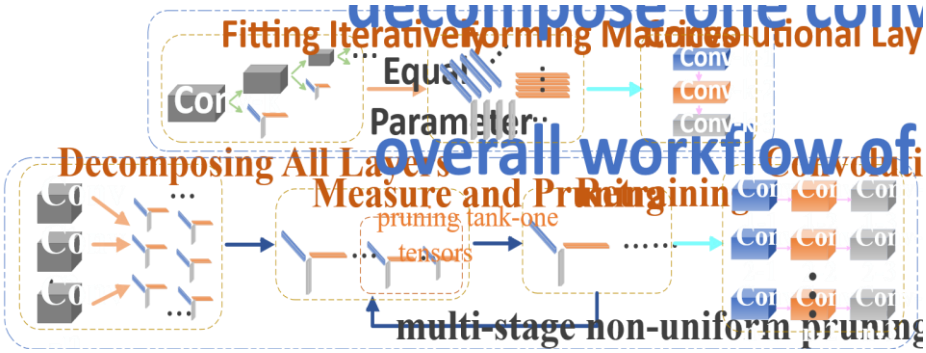

Fig. 1. The method for handling a convolutional layer and the overall workflow of RanpCode. In the overall workflow, an initial rank is obtained through minimal computational overhead. All convolutional layers are decomposed, and the resulting rank-one tensors are aggregated for evaluation and pruning. The tensors with the least importance are pruned, followed by a training recovery phase. This process is iteratively repeated until the model size is reduced to the target compression rate.

iterative training-based rank selection method that determines tensor ranks for each layer in a sequential manner. However, this approach overlooks the interdependencies of ranks across different layers, which may lead to suboptimal global rank configurations. In [25], the authors introduce a global rank search method, where the automatic search for appropriate tensor ranks is framed as an automatic search for an optimal low-rank structure. Despite improvements to the search space, the method's overall complexity remains high.

3 Methodology

In this section, we will provide a brief overview of the general form of CP decomposition, as well as its application to convolutional layers. We will then explain the complete CP decomposition based on the TPM. Additionally, we will detail the rank-based pruning scheme used for rank selection. The method for handling a convolutional layer and the overall workflow of our approach are illustrated in **Fig. 1**.

3.1 CP Decomposition

In CP decomposition, the original tensor is decomposed into a linear combination of rank-one tensors, the number of which is the tensor rank R. We focus on the three-way tensor and consider that a tensor λ with size $T \times S \times D$ is decomposed and denoted as:

$$\lambda = \sum_{r=1}^{R} a_r \otimes b_r \otimes c_r \tag{1}$$

where $\otimes$ is the outer product of the tensors and $\{a_r\}, \{b_r\}, \{c_r\}$ are vectors with sizes T, S and D respectively.

Then, we start to focus on the tensor of the convolutional layers. Convolutional layers in CNNs typically map an input tensor $\mathcal{X}$ into an output tensor $\mathcal{Y}$, resulting in a change in dimension from $S \times W \times H$ to $T \times W' \times H'$. This transformation is realized by utilizing a four-dimensional tensor $\mathcal{K}$, with size T × S × D × D:

$$\mathcal{Y}_{t,w',h'} = \sum_{s=1}^{S} \sum_{j=1}^{D} \sum_{i=1}^{D} \mathcal{K}_{t,s,j,i} \mathcal{X}_{s,w_j,h_i} \tag{2}$$

The subsequent objective is to approximate the tensor $\mathcal{K}$ with rank-R CP decomposition, as depicted in (3). The spatial dimensions of tensor $\mathcal{K}$ are preserved in decomposition process due to their relatively modest size, such as 3×3 or 5×5:

$$K_{t,s,j,i} = \sum_{r=1}^{R} U_{r,s}{}^{(1)} \mathcal{U}_{r,j,i}{}^{(2)} U_{t,r}{}^{(3)} \tag{3}$$

where $U_{r,s}^{(1)}$, $\mathcal{U}_{r,j,i}^{(2)}$ and $U_{t,r}^{(3)}$ are the three tensors with sizes $R \times S$, $R \times D \times D$, and $T \times R$, respectively.

The above equations indicates that the output tensor $\mathcal{Y}$ is obtained by applying a sequence of three separate convolutional operations to the input tensor $\mathcal{X}$ with smaller kernels ($U_{r,s}^{(1)}$, $\mathcal{U}_{r,j,i}^{(2)}$, $U_{t,r}^{(3)}$). Therefore, the process represented by (2) can be decomposed as follows:

$$\mathcal{Z}_{r,w,h} = \sum_{s=1}^{S} U_{r,s}{}^{(1)} \mathcal{X}_{s,w,h} \tag{4}$$

$$\mathcal{Z}'_{r,w',h'} = \sum_{j=1}^{D} \sum_{i=1}^{D} \mathcal{U}_{r,j,i}{}^{(2)} \mathcal{Z}_{r,w_j,h_i} \tag{5}$$

$$\mathcal{Y}_{t,w',h'} = \sum_{r=1}^{R} U_{t,r}{}^{(3)} \mathcal{Z}'_{r,w',h'} \tag{6}$$

where $\mathcal{Z}_{r,w,h}$ and $\mathcal{Z}_{r,w',h'}$ are intermediate tensors with sizes $R \times W \times H$ and $R \times W' \times H'$, respectively.

3.2 Complete CP Decomposition

In general, tensor decomposition is an optimization problem, i.e., minimizing the difference between the decomposed tensor and the target tensor. Therefore, the fundamental framework of TPM is to add rank-one tensors iteratively for fitting the original tensor, and the difference consequently decreases progressively. The process is denoted as:

$$\min \| \lambda - a_r \otimes b_r \otimes c_r \|_2 \tag{7}$$

$$\lambda = \lambda - a_r \otimes b_r \otimes c_r \tag{8}$$

where $r \in (0, R]$. R is the rank of tensor λ, set in advance.

However, this approach assumes that the original tensor has orthogonal decomposability; otherwise, when the rank is low, the tensor cannot be effectively decomposed. In complete CP decomposition, the goal is not to directly achieve model compression through decomposition, but rather to set a relatively large initial rank for the decomposition, which lays the foundation for subsequent rank-based pruning. This approach helps avoid issues related to orthogonal decomposability, thereby addressing the instability of CP decomposition. To avoid introducing unnecessary hyperparameters for tuning, we propose that the initial rank be set in a way that ensures the number of parameters before and after decomposition remains the same. This not only helps preserve the performance of the model after decomposition but also facilitates the adjustment of the compression rate hyperparameters. Therefore, the number of rank-one tensors with rank R can be computed as:

$$R = \left\lceil \frac{T*S*D}{T+S+D} \right\rceil \tag{9}$$

3.3 Rank-based Pruning Scheme for Rank Selection

Inspired by some work of pruning scheme [20], we propose rank-based pruning scheme, an automatic rank selection method for CP decomposition: After performing complete CP decomposition with higher rank, one can iteratively prune the rank-one tensors until the desired compression rate is achieved. This process can be interspersed with retraining to restore the model, reducing the difficulty of final fine-tuning and thereby determining the rank of each layer simultaneously. Due to its maximal retention of the original model's crucial parameters along with timely retraining to recover the lost parameters, this approach yields a globally optimized result, representing a smooth transition from the original model.

In section 3.2, we have set a relatively large rank and obtained satisfactory results: these rank-one tensors should exhibit gradually decreasing values of the L_2-norm, which is beneficial for assessing the significance of each rank-one tensor. The following will provide a detailed explanation of how to measure the impact of all rank-one tensors and how to perform pruning and retraining accordingly.

Norms Combined with Fisher Information. Although we have obtained rank-one tensors with decreasing norms, the two-norm cannot directly serve as a global criterion for comparing the importance of all rank-one tensors since these tensors originate from different layers of the model. Inspired by [13], the Fisher information, which represents the amount of information a parameter w carries about the dataset D, can serve as an indicator of parameter importance. The empirical Fisher information is estimated as:

$$\begin{aligned} I_w &= E\left[\left(\frac{\partial}{\partial w}\log p(D|w)\right)\right] \\ &\approx \frac{1}{|D|}\sum_{i=1}^{|D|}\left(\frac{\partial}{\partial w}y(x_i;w)\right)^2 \end{aligned} \tag{10}$$

where $d_i \in D$ and $\mathcal{Y}(x_i; w)$ represents the model's output with x_i denoting the model's input. Hence, the Fisher information is combined with the norm of the rank-one tensors as follows:

$$\hat{I}_{a_r,b_r,c_r} = \sum_{w_r \in \{a_r,b_r,c_r\}} \| w_r \circ \frac{\partial}{\partial w_r} \mathcal{Y}(D; w_r) \|_2 \tag{11}$$

The aforementioned estimation of Fisher information relies solely on first-order derivatives and has been demonstrated to effectively gauge the significance of parameters. Therefore, in CP decomposition, we can use (11) to evaluate the impact of all rank-one tensors and then selectively pruning a subset of them with smaller influence globally. In the context of deep learning models, each convolutional layer is decomposed into three convolutional layers, each having a dimension equal to the rank R. Pruning involves simultaneously removing slices corresponding to a specific index along this dimension.

$$\mathcal{Z}_{r,w,h} = \sum_{s=1}^{S} U_{r,s}{}^{(1)} \mathcal{X}_{s,w,h}, \quad r \notin P \tag{12}$$

$$\mathcal{Z}'_{r,w',h'} = \sum_{j=1}^{D} \sum_{i=1}^{D} \mathcal{U}_{r,j,i}{}^{(2)} \mathcal{Z}_{r,w_j,h_i}, \quad r \notin P \tag{13}$$

$$\mathcal{Y}_{t,w',h'} = \sum_{r \notin P} U_{t,r}{}^{(3)} \mathcal{Z}'_{r,w',h'} \tag{14}$$

where the P denotes the set of indices employed for the truncation of a rank-one tensor.

Multi-Stage Non-Uniform Pruning. Although pruning is performed by selecting the least impactful set of rank-one tensors globally, their combined impact remains significant when the number of rank-one tensors is large. If only a single pruning step is performed to achieve the target compression rate, the resulting model performance loss is considerable, requiring substantial time for fine-tuning with potentially suboptimal recovery. To address this, we propose a multistage nonuniform pruning strategy that allows the original model to be smoothly and stably pruned to the target model: the total rank of the model can be equated to the number of model parameters, so the target pruning amount can be calculated based on the desired compression rate. This target pruning amount is then distributed non-uniformly across multiple global pruning steps based on impact size. The amount of pruning decreases progressively, with minimal retraining performed after each pruning step to reduce loss; finally, fine-tuning is carried out until the model achieves optimal performance. To sum up, the overall procedure of the proposed RanpCode is summarized in Algorithm 1.

Algorithm 1 Rank-based Pruning After CP Decomposition

Input: Pretrained Model M, Number of pruning iterations N, Pruning amount list. L

Output: Compact Model. M

obtain R of all layers by using (9);
obtain (S_r, D_r, D_r, T_r) by using (7) and (8);
$U_{r,s}, \mathcal{U}_{r,i,j}, U_{t,r} \longleftarrow S_r, D_r \otimes D_r, T_r$;
for each $i \in [1.N]$ **do**
    obtain I_r of all layers by using (11);
    $P \longleftarrow argmin(I_r, L_i)$, P is an index set.;
    $U_{r,s}.\mathcal{U}_{r,i,j}, U_{t,r} \xleftarrow{r \notin P} U_{r,s}, \mathcal{U}_{r,i,j}, U_{t,r}$;
    $U_{r,s}, \mathcal{U}_{r,i,j}, U_{t,r} \longleftarrow model\_train(U_{r,s}, \mathcal{U}_{r,i,j}, U_{t,r})$;
end for
$M \longleftarrow model\_constitute(U_{r,s}, \mathcal{U}_{r,i,j}, U_{t,r})$;

Table 1. A component-wise analysis conducted to demonstrate two key aspects: (1) the superior stability of complete CP decomposition compared to ALS method, and (2) the effectiveness of Fisher information-based pruning in improving the compression performance with ResNet-32 on the CIFAR-100 dataset.

Complete CP	Fisher Information	Top-1% Accuracy original	Top-1% Accuracy compact	epoch ratio	#para↓
		70.16	55.67	80	3
✓		70.16	65.69	80	3
	✓	70.16	-	80	3
✓	✓	70.16	69.75	80	3

4 Experiments

In this section, we validate the effectiveness of RanpCode on image classification tasks across multiple relevant datasets including CIFAR and ImageNet. First, we conduct comprehensive ablation studies on the key components of RanpCode, encompassing the complete CP decomposition, Fisher information-based evaluation metrics, and the multi-stage non-uniform pruning strategy. Subsequently, we perform detailed comparisons with recent state-of-the-art low-rank decomposition compression methods. Finally, we present an in-depth analysis and discussion of RanpCode based on visualized results.

Implementation Details

We compressed all convolutional layers remaining fully-connected layers intact. The TensorLy [16] library provides an implementation of TPM method. We have set the parameters appropriately, including the rank R and an iteration count of 10 for each fitting. During the non-uniform pruning process, a total of 8 pruning steps are performed, with preset ratios for each pruning. We use the standard SGD optimizer with Nesterov momentum set to 0.9 for model retraining. The learning rates are initialized

to 0.05 and 0.01 for CIFAR and ImageNet, respectively. After pruning to the target compression ratio, the learning rates are scaled down by 0.1 for training, and then further scaled down by 0.1 for fine-tuning. In addition, batch size are set as 128, 256 and 128 for CIFAR-10, CIFAR-100 and ImageNet, respectively.

Ablation Studies

1) Complete CP decomposition and Fisher information: Here we present a systematic ablation study to quantitatively assess the individual contributions of the key components in our RanpCode framework: (1) the complete CP decomposition module for advanced tensor factorization and (2) the theoretically motivated Fisher information-based evaluation metric. All experiments were conducted using ResNet32 on CIFAR-100 under identical training conditions (80 epochs) with consistent fine-tuning protocols.

Table 2. A comparison of the effects of varying pruning rounds and decrementing pruning amounts on performance, using the same simple training strategy and same compression, was conducted on ResNet32 with the CIFAR-100 dataset.

Rounds	Pruning Amounts	Top-1% Accuracy		epoch	#para↓
		original	compact		ratio
1	reduction	70.16	63.81	60	×3
2	reduction	70.16	64.83	60	×3
4	reduction	70.16	65.01	60	×3
8	**reduction**	**70.16**	**65.36**	60	**×3**
8	Uniform	70.16	64.74	60	×3

To precisely evaluate each component's impact, we performed controlled experiments through method substitution: (i) replacing our CP decomposition with conventional Alternating Least Squares (ALS) and (ii) substituting our Fisher Information-based pruning with standard random pruning. Crucially, the ALS-Fisher combination exhibited fundamental instability - the nearly degenerate Fisher Information values produced by ALS for rank-1 tensors frequently led to catastrophic layer pruning. This pathological behavior induced severe architectural degradation, making this configuration incapable of yielding stable or interpretable results.

Our experimental results demonstrate that both proposed components synergistically contribute to performance improvements. The complete CP decomposition provides more stable tensor factorization compared to ALS, while the Fisher information metric enables principled pruning decisions that maintain network integrity. These findings in **Table 1** underscore the importance of our carefully designed components in achieving robust model compression.

2) multi-stage non-uniform pruning: Through systematic ablation studies, we conducted a comprehensive comparison between the proposed multi-stage non-uniform pruning strategy and the baseline approach employing uniform pruning. All experiments were performed on the ResNet32 architecture using the CIFAR-100 dataset under identical training conditions (60 epochs), while maintaining consistent fine-tuning protocols. Our extensive comparisons across various pruning stage configurations conclusively demonstrate the superiority of the multi-stage non-uniform pruning approach.

Experimental results indicate that progressively increasing the number of pruning iterations while gradually reducing the pruning amount contributes significantly to performance enhancement.

Considering that further increasing the number of pruning stages would exponentially escalate the difficulty and workload associated with optimizing the hyperparameter allocation of pruning amounts across stages, we ultimately fixed the number of pruning stages at eight. This configuration was subsequently adopted as the standard hyperparameter setting for all follow-up experiments. The experimental findings in **Table 2** provide conclusive validation of the effectiveness of the multi-stage non-uniform pruning strategy.

Table 3. Comparison with different tensor decomposition-based approaches for ResNet on CIFAR. Typically, two different compression ratios are tested for comparison, and the same experimental procedure is followed in subsequent studies.

Model	Method	Top-1% Accuracy		Δ	#para↓
		original	compact		ratio
ResNet-20 Cifar10	PSTR-M [18]	91. 25	88. 50	-2. 75	×6. 8
	PSTR-S [18]	91. 25	90. 80	-0. 45	×2. 5
	BATUDE [29]	91. 25	89. 47	-1. 78	×6. 8
	BATUDE [29]	91. 25	91. 02	-0. 23	×2. 6
	RanpCode(ours)	91. 73	90. 24	-1. 49	×7. 1
	RanpCode(ours)	91. 73	91. 80	+0. 07	×2. 7
ResNet-32 Cifar100	PSTR-M [18]	68. 10	66. 77	-1. 33	×5. 2
	PSTR-S [18]	68. 10	68. 05	-0. 05	×2. 4
	BATUDE [29]	68. 10	66. 96	-1. 14	×5. 2
	BATUDE [29]	68. 10	68. 95	+0. 85	×2. 6
	RanpCode(ours)	70. 16	68. 82	-1. 34	×5. 2
	RanpCode(ours)	70. 16	70. 83	+0. 67	×2. 6

CIFAR Results

Table 3 presents the evaluation results on the CIFAR dataset, with each method assessed for both low and high compression rates. For compressing the ResNet-20 model, RanpCode achieves up to 0.07% higher top-1 accuracy compared to the original model at a higher compression ratio of × 2.7.

In the high compression rate group, our method also surpasses the highly regarded BATUDE method at a compression ratio of × 7.1. For the ResNet-32 model, although the relative performance of RanpCode is somewhat constrained by the already high baseline (2% higher), our method still outperforms most other compression methods.

ImageNet Results

We also evaluate our RanpCode on the ImageNet dataset for the compression of ResNet-18 and ResNet-50 models, with a focus on reducing FLOPs and parameters, respectively. **Table 4** presents a comparison of our method with other tensor decomposition approaches.

Table 4. Comparison with tensor decomposition-based approaches for ResNet on ImageNet.

Model	Method	Top-1 (%)	Top-5 (%)	#FLOPs (%↓)	#para (%↓)
Resnet18	baseline-Torchvision	69.76	89.08	-	-
	Stable EPC [22]	69.07	88.93	67.64	-
	ALDS [19]	69.22	89.03	43.51	66.70
	HALOC [25]	70.14	89.38	63.81	71.31
	RanpCode(Ours)	69.97	89.25	68.25	79.33
	RanpCode(Ours)	70.31	89.37	63.78	77.64
Resnet50	baseline-Torchvision	76.13	92.86	-	-
	HT-2 [8]	74.79	92.15	63.50	64.91
	RRTD [3]	74.80	92.42	53.92	57.63
	RanpCode(Ours)	75.41	92.47	58.08	65.75
	RanpCode(Ours)	75.81	92.59	49.65	59.96

For the ResNet-18 model, RanpCode achieves a 0.55% improvement in top-1 accuracy over the baseline model while also reducing FLOPs by 63.78%. Additionally, compared to the state-of-the-art automatic low-rank compression method HALOC, RanpCode provides a 0.17% higher top-1 accuracy with significantly fewer parameters and comparable computational costs. Despite achieving the highest FLOPs reduction of 68.25%, RanpCode maintains a 0.23% higher accuracy than the baseline model, whereas Stable EPC, the previously leading CP-based compression method, shows a 0.69% decrease in performance post-compression. These results strongly demonstrate RanpCode's effectiveness in reducing FLOPs.

For compressing the ResNet-50 model, RanpCode exhibits superior performance, particularly in parameters reduction. With a 65.75% reduction in parameters, RanpCode achieves a top-1 accuracy of 75.41%, which is 0.62% higher than the HT-2 method, which achieves less parameters reduction. Furthermore, compared to RRTD, our approach yields a 1.01% higher accuracy with a higher compression ratio. These results provide compelling evidence of RanpCode's proficiency in parameters reduction.

Analysis & Discussion

Fig. 2 presents an analysis of the ResNet-18 training process under two target compression levels. Both the training and test accuracies show a steady increase starting from a relatively high initial value (above 50%), demonstrating the effectiveness of the complete CP decomposition: First, it provides a strong initial rank, transforming the model into a structure that adapts well to rank pruning while maintaining relatively good performance. Second, it successfully mitigates the instability typically associated with CP decomposition, ensuring stable performance improvement throughout the subsequent training process without significant fluctuations. Moreover, the intermittent pruning steps do not cause substantial accuracy fluctuations, indicating that the rank-based pruning strategy effectively minimizes the impact of each pruning operation, allowing the model to quickly recover and achieve high performance in later stages of training.

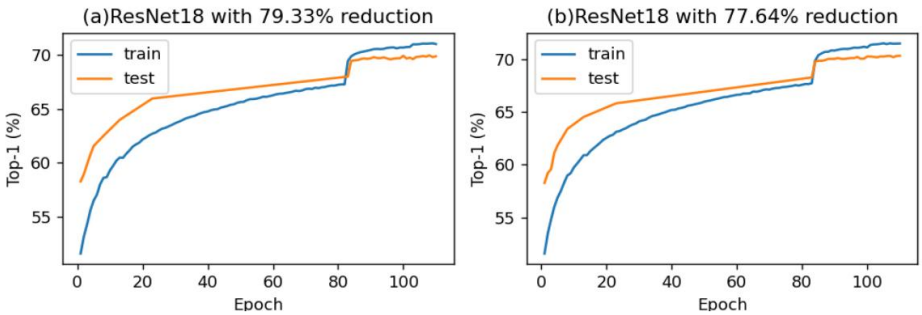

Fig. 2. The change of Top-1 accuracy in training with two compression 79.33% and 77.64% for the rank-one tensors. The eight pruning occur sequentially after epoch [1,2,3,4,5,8,13,23].

In fact, the concept of using importance factors for comparison and rank-based pruning on a global scale can also be combined with other tensor decomposition-based compression methods. This is because tensor decomposition-based compression methods can generally be understood as rank pruning from a theoretical perspective—performing a fully equivalent decomposition of the tensor (which may not be practically feasible) and then pruning it to a preset rank. Some compression methods obtain the rank of each layer in the model through complex iterative calculations, global search, or other techniques. In such cases, instead of directly using these ranks for decomposition, one can combine the idea of RanpCode by employing some significance factor to measure all rank tensors, progressively pruning them to the previously obtained ranks. This approach allows for more stable compression and makes it easier to recover the model's performance, thus achieving better overall performance.

5 Conclusion

In this paper, we propose RanpCode, a rank-based pruning method performed after the completion of CP decomposition. The complete CP decomposition addresses the inherent instability of traditional CP methods, providing an initial rank tensor that enables stable training and performance improvement. The rank-based pruning strategy offers an efficient automatic rank selection, without introducing any additional parameters or time-consuming training processes. It dynamically selects the global rank during model training based on a simple significance criterion. Evaluations on different models demonstrate that our approach significantly outperforms existing state-of-the-art model compression techniques.

Acknowledgments. This work was supported by the Postdoctoral Fellowship Program of CPSF (under Grant No. GZB20240113), the Sichuan Science and Technology Program (granted No. 2024ZDZX0011 and No. 2025ZNSFSC1472), and the Sichuan Central-Guided Local Science and Technology Development Program (under Grant No. 2023ZYD0165). Additionally, all the authors gratefully acknowledge the computational resources provided by the Supercomputing Center of University of Electronic Science and Technology of China.

Disclosure of Interests. The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Astrid, M., Lee, S.I.: Cp-decomposition with tensor power method for convolutional neural networks compression. In: 2017 IEEE International Conference on Big Data and Smart Computing (BigComp). pp. 115–118. IEEE (2017)
2. Astrid, M., Lee, S.I., Seo, B.S.: Rank selection of cp-decomposed convolutional layers with variational bayesian matrix factorization. In: 2018 20th International Conference on Advanced Communication Technology (ICACT). pp. 347–350. IEEE (2018)
3. Dai, W., Fan, J., Miao, Y., Hwang, K.: Deep learning model compression with rank reduction in tensor decomposition. IEEE Transactions on Neural Networks and Learning Systems (2023)
4. Deng, C., Yin, M., Liu, X.Y., Wang, X., Yuan, B.: High-performance hardware architecture for tensor singular value decomposition. In: 2019 IEEE/ACM International Conference on Computer-Aided Design (ICCAD). pp. 1–6. IEEE (2019)
5. Denton, E.L., Zaremba, W., Bruna, J., LeCun, Y., Fergus, R.: Exploiting linear structure within convolutional networks for efficient evaluation. Advances in neural information processing systems 27 (2014)
6. Dong, Z., Yao, Z., Arfeen, D., Gholami, A., Mahoney, M.W., Keutzer, K.: Hawq-v2: Hessian aware trace-weighted quantization of neural networks. Advances in neural information processing systems 33, 18518–18529 (2020)
7. Fang, G., Ma, X., Song, M., Mi, M.B., Wang, X.: Depgraph: Towards any structural pruning. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 16091–16101 (2023)
8. Gabor, M., Zdunek, R.: Compressing convolutional neural networks with hierarchical tucker-2 decomposition. Applied Soft Computing 132, 109856 (2023)
9. Garipov, T., Podoprikhin, D., Novikov, A., Vetrov, D.: Ultimate tensorization:compressing convolutional and fc layers alike. arXiv preprint arXiv:1611.03214(2016)
10. Gusak, J., Kholiavchenko, M., Ponomarev, E., Markeeva, L., Blagoveschensky, P., Cichocki, A., Oseledets, I.: Automated multi-stage compression of neural networks.In: Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops. pp. 0–0 (2019)
11. Hao, S., Zhou, Y., Guo, Y.: A brief survey on semantic segmentation with deep learning. Neurocomputing 406, 302–321 (2020)
12. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In:Proceedings of the IEEE conference on computer vision and pattern recognition.pp. 770–778 (2016)
13. Hsu, Y.C., Hua, T., Chang, S., Lou, Q., Shen, Y., Jin, H.: Language model compression with weighted low-rank factorization. arXiv preprint arXiv:2207.00112(2022)
14. Idelbayev, Y., Carreira-Perpinán, M.A.: Low-rank compression of neural nets:Learning the rank of each layer. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8049–8059 (2020)
15. Kim, Y.D., Park, E., Yoo, S., Choi, T., Yang, L., Shin, D.: Compression of deep convolutional neural networks for fast and low power mobile applications. arXiv preprint arXiv:1511.06530 (2015)

16. Kossaifi, J., Panagakis, Y., Anandkumar, A., Pantic, M.: Tensorly: Tensor learning in python. Journal of Machine Learning Research 20(26), 1–6 (2019)
17. Lebedev, V., Ganin, Y., Rakhuba, M., Oseledets, I., Lempitsky, V.: Speeding-up convolutional neural networks using fine-tuned cp-decomposition. arXiv preprint arXiv:1412.6553 (2014)
18. Li, N., Pan, Y., Chen, Y., Ding, Z., Zhao, D., Xu, Z.: Heuristic rank selection with progressively searching tensor ring network. Complex & Intelligent Systems pp. 1–15 (2021)
19. Liebenwein, L., Maalouf, A., Feldman, D., Rus, D.: Compressing neural networks: Towards determining the optimal layer-wise decomposition. Advances in Neural Information Processing Systems 34, 5328–5344 (2021)
20. Liu, Z., Li, J., Shen, Z., Huang, G., Yan, S., Zhang, C.: Learning efficient convolutional networks through network slimming. In: Proceedings of the IEEE international conference on computer vision. pp. 2736–2744 (2017)
21. Pan, Y., Xu, J., Wang, M., Ye, J., Wang, F., Bai, K., Xu, Z.: Compressing recurrent neural networks with tensor ring for action recognition. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 33, pp. 4683–4690 (2019)
22. Phan, A.H., Sobolev, K., Sozykin, K., Ermilov, D., Gusak, J., Tichavsk`y, P., Glukhov, V., Oseledets, I., Cichocki, A.: Stable low-rank tensor decomposition for compression of convolutional neural network. In: Computer Vision–ECCV 2020:16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIX 16. pp. 522–539. Springer (2020)
23. Rayens, W.S., Mitchell, B.C.: Two-factor degeneracies and a stabilization of parafac. Chemometrics and Intelligent Laboratory Systems 38(2), 173–181 (1997)
24. Wang, W., Sun, Y., Eriksson, B., Wang, W., Aggarwal, V.: Wide compression:Tensor ring nets. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 9329–9338 (2018)
25. Xiao, J., Zhang, C., Gong, Y., Yin, M., Sui, Y., Xiang, L., Tao, D., Yuan, B.: Haloc:hardware-aware automatic low-rank compression for compact neural networks. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 37, pp. 10464–10472 (2023)
26. Xiong, K., Gong, S., Ye, X., Tan, X., Wan, J., Ding, E., Wang, J., Bai, X.: Cape:Camera view position embedding for multi-view 3d object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 21570–21579 (2023)
27. Yang, C., Liu, H.: Stable low-rank cp decomposition for compression of convolutional neural networks based on sensitivity. Applied Sciences 14(4), 1491 (2024)
28. Yin, M., Liao, S., Liu, X.Y., Wang, X., Yuan, B.: Towards extremely compact rnns for video recognition with fully decomposed hierarchical tucker structure. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12085–12094 (2021)
29. Yin, M., Phan, H., Zang, X., Liao, S., Yuan, B.: Batude: Budget-aware neural network compression based on tucker decomposition. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 36, pp. 8874–8882 (2022)
30. Yin, M., Sui, Y., Yang, W., Zang, X., Gong, Y., Yuan, B.: Hodec: Towards efficient high-order decomposed convolutional neural networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12299–12308 (2022)
31. Zhao, B., Cui, Q., Song, R., Qiu, Y., Liang, J.: Decoupled knowledge distillation. In: Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition. pp. 11953–11962 (2022)