Deep Learning through Efficient Model Design

Van Tien Pham[a]

[a] *Université de Toulon, France*

Abstract

In the rapidly evolving landscape of deep learning, the proliferation of large-scale neural networks has brought forth new horizons of artificial intelligence. However, their sheer size and computational demands have posed significant challenges, limiting their applicability in resource-constrained environments. The overarching problem of network compression seeks to mitigate these challenges by efficiently designing deep neural models. In this report, we embark on a journey to address this problem, exploring novel avenues for model compression.

Our first contribution, CORING, introduces a pioneering filter pruning method that harnesses tensor decomposition, preserving the multidimensional essence of filters. By leveraging the power of CORING, we achieve impressive reductions in model size and computational requirements while retaining or even enhancing performance. CORING's ability to generalize models through pruning is demonstrated across various architectures and datasets.

The second contribution, NORTON, unveils a hybrid network compression technique that combines tensor decompositions with structured pruning. NORTON offers a comprehensive approach to model compression, optimizing architecture, and reducing the number of parameters. With NORTON, we attain superior compression ratios and accuracy retention, making it a versatile tool for model optimization.

Looking ahead, our research sets the stage for future investigations. Potential avenues include delving deeper into the compression domain, expanding to encompass various decomposition techniques, exploring a broader spectrum of neural network architectures, and applying these efficient models to diverse applications. As we navigate the evolving landscape of deep learning, the pursuit of efficient model design remains at the forefront, driving innovation and unlocking the potential for AI in resource-constrained scenarios.

Keywords: network compression, low-rank representation, filter pruning, frugal machine learning

This first-year report has been examined by the Thesis Follow-up Committee (CSI):

Thesis Supervisor:	Thanh Phuong Nguyen,	Associate Professor,	Université de Toulon
Thesis Co-supervisor:	Yassine Zniyed,	Associate Professor,	Université de Toulon
CSI Member:	Seong G. Kong,	Professor,	Sejong University
CSI Member:	Mohammed Nabil El Korso,	Professor,	Paris Saclay University

This research is conducted within the Laboratoire d'Informatique et Systèmes, Université de Toulon (UTLN), Aix Marseille University, CNRS, with funding provided through a doctoral contract with UTLN. Van Tien Pham, who received a master's degree in computer science from Hanoi University of Science and Technology, Vietnam, in 2020, embarks on a journey to contribute to the field of network compression.

1. Introduction

Over the past decade, the widespread adoption of large-scale deep neural networks (DNNs) [31, 10] has ushered in a transformative era across various domains. These DNNs have reshaped the landscape of object classification [31], natural language understanding [26], and have played pivotal roles in autonomous navigation and advanced pattern recognition tasks.

Email address: van-tien-pham@etud.univ-tln.fr (Van Tien Pham)

However, amid these achievements, significant challenges have emerged. The deployment of these large DNNs, especially in resource-constrained environments, has revealed their substantial computational and memory demands [42]. These demands have spurred research efforts to make large-scale DNNs more accessible and practical [8, 15, 12, 41, 37, 24, 11, 5].

Model compression techniques offer several compelling advantages:

- Reduced Computational Demands: Model compression techniques significantly decrease the computational requirements for both training and inference, making DNNs feasible for a broader range of applications. Memory Efficiency: By reducing the model size, memory usage is optimized, enabling deployment on resource-constrained devices.
- Environmental Impact: Smaller, more efficient models consume fewer computational resources, contributing to reduced energy consumption and environmental sustainability.
- Faster Inference: Compressed models lead to quicker inference times, making them suitable for real-time applications.
- Improved Generalization: Model compression often leads to improved model generalization, enhancing performance on various tasks.

Yet, the path to effective network compression is fraught with challenges:

- Preserving Essential Features: Achieving compression without compromising critical model features is a delicate balance.
- Maintaining Accuracy: Ensuring that compressed models maintain accuracy levels comparable to their larger counterparts is a significant challenge.
- Robustness in Compression: Developing methods to ensure the robustness of compressed models, especially in the face of noisy or incomplete data.
- Exploration of Novel Architectures: Investigating new model architectures that are inherently more compressible and efficient.
- Scalability: Adapting compression techniques to large and complex DNNs poses scalability challenges.

In response to these challenges, prior research has embarked on a quest to make large-scale DNNs more accessible and practical [8, 15, 12, 41, 37, 24, 11, 5]. Several techniques have emerged, aiming to mitigate the computational burden and environmental impact of these models. Among these methods, two prominent approaches hold particular promise: 1. Network pruning systematically identifies and removes unnecessary parameters or structures from neural networks. Techniques include weight pruning [8], channel pruning [24], and N:M sparsity [22], all aimed at streamlining models while minimizing performance degradation. 2. Low-rank representation offer a mathematical framework to represent the weight tensors of neural networks in lower-dimensional subspaces [13, 12]. Common decompositions include CP decomposition [13], Tucker decomposition [36], and high-order SVD [3], which reduce the memory footprint of models while preserving their representational power [14]. These techniques, along with other methods like quantization [37], knowledge distillation [11], and neural architecture search (NAS) [5], collectively contribute to the endeavor of making deep neural networks more efficient and resource-friendly.

This report is organized into four sections. Sections 2 and 3 introduce our contributions, CORING and NORTON. In Section 4, we draw conclusions and outline directions for future research.

2. The CORING Framework

2.1. Introduction

We propose a novel approach, called CORING, for effiCient tensOr decomposition-based filteR prunING. It is a filter pruning technique using tensor decompositions [13]. Specifically, we decompose each layer's filters using the higher-order singular value decomposition (HOSVD) [3] and use this representation to measure similarity between filters, rather than considering the entire filter in its tensor, matrix, or vector form. This method allows to (i) preserve the multidimensional structure of the filters and their essential information while providing a low-rank approximation, and (ii) reduce computational time as we will show later. This approach is general and can work with any similarity metric. In our experiments, we evaluate our approach using the Euclidean and cosine distances, as well as a novel metric we introduced called VBD (Variance-Based Distance), which is derived from the Signal to Noise Ratio (SNR) distance [43]. We show that our approach is effective across all these metrics and outperforms SOTA.

This work brings the following contributions:

- Introducing tensor decompositions, namely the HOSVD, for filter pruning, which preserve the multidimensional structure of filters while providing a low-rank approximation to reduce computational time.
- Demonstrating a novel and general way of calculating similarity between filters that utilizes the low-rank approximation provided by the HOSVD, rather than relying on the full filter or its reshaped versions. Our approach is tested with Euclidean, cosine, and VBD distances.
- Presenting a simple filter selection method based on the similarity matrix, which considers the relationships between filters in the same layer.
- Evaluating the proposed framework across a range of representative computer vision tasks, encompassing image classification, object detection, instance segmentation, and keypoint detection. Extensive experiments are conducted to show the effectiveness of CORING in terms of accuracy, parameter reduction, and MACs reduction compared to SOTA.

2.2. Approach

The CORING approach, illustrated in Fig. 1, can be divided into three distinct steps, each of which we will discuss separately in the following subsubsections.

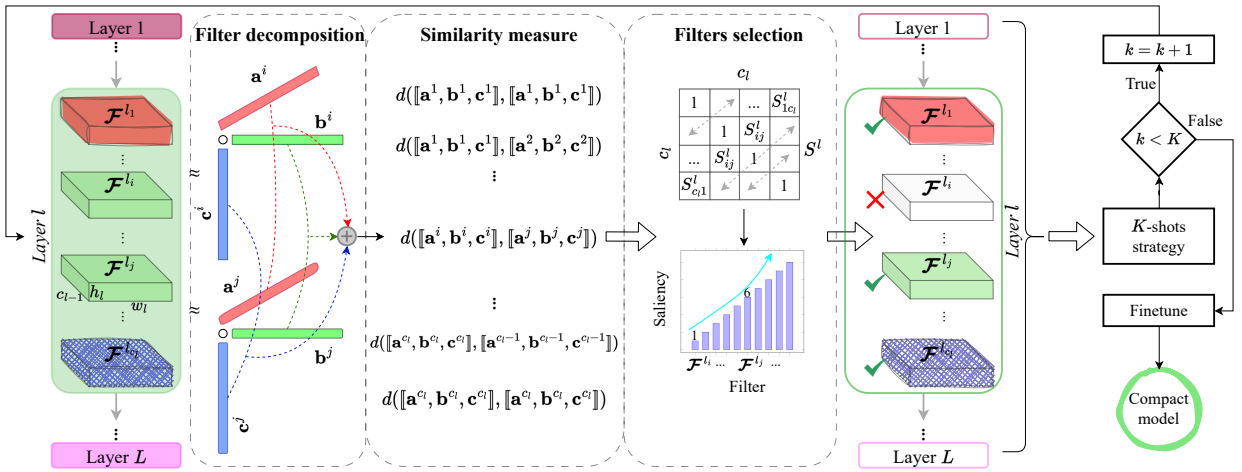

Figure 1: The CORING approach for filter pruning in one layer, summarized in three steps.

2.2.1. Filter decomposition

In this section, for the sake of clarity and to simplify the notation, we consider a filter without its subscript as $\mathcal{F}$ of size $c_{l-1} \times h_l \times w_l$. Now, if we apply the TD [13] to $\mathcal{F}$ by considering that $R_1 = R_2 = R_3 = 1$, the model reduces to

$$\mathcal{F} \approx s \times_1 \mathbf{a} \times_2 \mathbf{b} \times_3 \mathbf{c} = [\![s; \mathbf{a}, \mathbf{b}, \mathbf{c}]\!], \tag{1}$$

where $\mathbf{a} \in \mathbb{R}^{c_{l-1}}$, $\mathbf{b} \in \mathbb{R}^{h_l}$, $\mathbf{c} \in \mathbb{R}^{w_l}$, and s is a scalar that can also be seen as a 3-order tensor $s \in \mathbb{R}^{1\times1\times1}$, . Without loss of generality, we can now denote $\mathcal{F}$ simply as

$$\mathcal{F} \approx [\![\mathbf{a}, \mathbf{b}, \mathbf{c}]\!]. \tag{2}$$

To decompose $\mathcal{F}$ as in (2), there are multiple tensor decomposition methods available. In this work, we chose to use the HOSVD algorithm [3]. This choice is justified by several factors. Firstly, the HOSVD is a non-iterative algorithm, unlike ALS-based techniques [9], which can be computationally expensive. Secondly, it is based on the SVD, which ensures that the approximation with respect to the decomposed matrices is optimal [6]. Finally, the HOSVD is relatively easy to implement, making it a practical choice for many applications.

2.2.2. Similarity measure

The authors of [43] proposed an SNR-based metric to measure the similarity of image pairs for deep metric learning. While this *quasi*-metric has been shown to be effective, it has one major caveat that should be noted: it does not satisfy the symmetry property, which is important for distance functions. To remedy this, we define the VBD as

$$d_{VBD}(\boldsymbol{\mathcal{F}}^i, \boldsymbol{\mathcal{F}}^j) = \frac{\mathrm{Var}(\boldsymbol{\mathcal{F}}^i - \boldsymbol{\mathcal{F}}^j)}{\mathrm{Var}(\boldsymbol{\mathcal{F}}^i) + \mathrm{Var}(\boldsymbol{\mathcal{F}}^j)}. \tag{3}$$

Let us now consider $d(.,.)$ as a general distance function. To calculate the distance between a pair of filters $\boldsymbol{\mathcal{F}}^i$, $\boldsymbol{\mathcal{F}}^j$, we will use their HOSVD as in (2). Let us assume that $\boldsymbol{\mathcal{F}}^i = [\![\mathbf{a}^i, \mathbf{b}^i, \mathbf{c}^i]\!]$ and $\boldsymbol{\mathcal{F}}^j = [\![\mathbf{a}^j, \mathbf{b}^j, \mathbf{c}^j]\!]$. In this case, we can calculate the distance between $\boldsymbol{\mathcal{F}}^i$ and $\boldsymbol{\mathcal{F}}^j$ as follows.

$$d(\boldsymbol{\mathcal{F}}^i, \boldsymbol{\mathcal{F}}^j) = d\big([\![\mathbf{a}^i, \mathbf{b}^i, \mathbf{c}^i]\!], [\![\mathbf{a}^j, \mathbf{b}^j, \mathbf{c}^j]\!]\big). \tag{4}$$

2.2.3. Filters selection

The algorithm takes as input a similarity matrix between all pairs of filters, the set of filters, and a sparsity target. The output of the algorithm is a set of κ selected filters. The procedure works by iteratively deleting filters that are most similar to the other filters. The algorithm starts by finding the pair of filters with the highest similarity and deleting one of the filters. The choice of the filter to delete is based on the sum of its similarities with the other filters in the layer. The algorithm then updates the similarity matrix by deleting the row and column of the deleted filter and continues to delete filters until the desired sparsity target is reached.

2.2.4. Pruning strategy

It is a variation of multi-shot pruning, which prunes the network in K rounds with the possibility of fine-tuning between rounds. By fixing the number of pruning rounds, K-shots pruning provides better control over the pruning process and allows for more efficient use of computational resources. In K-shots pruning, the number of filters to prune in each round is calculated based on the total number of filters in the network and the desired overall sparsity level. After each round of pruning, the pruned network can be fine-tuned to recover any loss in accuracy. This fine-tuning step can be repeated between rounds to further improve the performance of the pruned network.

2.3. Experiments

The CORING method is assessed in three variations, namely CORING-C, CORING-E, and CORING-V, each corresponding to a specific distance metric: cosine similarity, Euclidean distance, and VBD, respectively. We denote CORING-X-K as the combination of distance X and strategy of K shots, where X $\in \{$C, E, V$\}$ and $K \in \{5, 10, 15\}$. If the suffix does not contain the variable K, then it implies that one-shot pruning is being performed.

VGG-16-BN. Table 1 shows the pruning results of VGGNet on CIFAR-10. In all three levels, compared with other methods, CORING consistently gets the highest accuracy while maintaining the same level of pruning. Results from both pruning strategies outperform the state-of-the-art.

ResNet-50. To assess the scalability of CORING, we conduct experiments on the extensive dataset ImageNet by addressing ResNet-50 as shown in Table 2. Across all evaluated scenarios, CORING consistently outperforms other approaches in terms of both performance and complexity reduction.

3. The NORTON Framework

3.1. Introduction

Our proposed method operates on the original weight tensor in a filter-by-filter manner. The key insight lies in the fact that in a convolution layer, the input undergoes convolution separately with each filter, and the outputs are then aggregated to generate the final feature map. Therefore, it is intuitive to decompose each 3-order filter tensor individually. The filters decomposition approach offers a higher level of granularity compared to its counterpart, layer decomposition. With filters decomposition, not only the multidimensional property is strictly preserved, but also the layer's 4-order weight is spontaneously interpreted as a set of 3-order filters. Additionally, an interesting

Table 1: Pruning results of VGG-16-BN on CIFAR-10

Model	Top1	Params (↓%)	MACs (↓%)
VGG-16-BN	93.96	14.98M(00.0)	313.73M(00.0)
CHIP [34]	93.86	2.76M(81.6)	131.17M(58.1)
EZCrop [23]	93.01	2.76M(81.6)	131.17M(58.1)
DECORE-500 [1]	94.02	5.54M(63.0)	203.08M(35.3)
CORING-C	**94.16**		
CORING-E	94.10		
CORING-V	94.11		
CORING-C-5 (Ours)	94.25	**2.76M(81.6)**	**131.17M(58.1)**
CORING-E-5	**94.42**		
CORING-V-10	94.36		
FSM [4]	93.73	N/A(86.3)	N/A(66.0)
FPAC [39]	93.86	2.50M(83.3)	104.78M(66.6)
AutoBot [2]	94.01	6.44M(57.0)	108.71M(65.3)
CORING-C	93.79		
CORING-E	**94.20**		
CORING-V	94.19		
CORING-C-15 (Ours)	**94.07**	**2.50M(83.3)**	**104.78M(66.6)**
CORING-E-15	94.03		
CORING-V-10	94.04		
DECORE-100 [1]	92.44	0.51M(96.6)	51.20M(81.5)
RASP-70M [45]	92.81	1.13M(92.4)	70.15M(77.7)
CHIP [34]	93.18	1.90M(87.3)	66.95M(78.6)
CORING-C	93.56		
CORING-E	93.54		
CORING-V	**93.63**		
CORING-C-10 (Ours)	93.68	1.90M(87.3)	66.95M(78.6)
CORING-E-15	**93.83**		
CORING-V-5	93.71		

side effect of filters decomposition is that it leads to a narrower range of ranks, which simplifies the rank selection process, as demonstrated later in subsection 3.2.1. Another notable difference is that filters decomposition replaces the original layer with 3 sublayers, while layer decomposition requires 4 sublayers, potentially increasing network depth unnecessarily and introducing the possibility of gradient vanishing issues.

The second consideration of this work is to combine filters decomposition and filter pruning to leverage the independent advantages of each method. While previous studies [30, 7, 16, 40] have used low-rank representations and network pruning for compression purposes, they have not explored an orthogonal combination of these techniques. In these works, low-rank representations have been used only in the pruning step without directly contributing to the reduction of model size. In contrast, our approach takes a different direction by sequentially applying low-rank representations and pruning in two separate phases, enabling a dual compression process. To the best of our knowledge, this direction has not been extensively investigated in the literature.

Briefly, the contributions of this work are three-fold:

- Firstly, we introduce a novel filters decomposition method, promoting its differentiation with existing layer decomposition and reshaped decomposition methods.
- Secondly, we investigate the sequential combination of filters decomposition and filter pruning. We propose a novel filter pruning algorithm designed to address the challenges associated with this integration scheme.
- Thirdly, we evaluate the proposed framework on representative vision tasks including image classification, object detection, instance segmentation, and keypoint detection. We compare NORTON with SOTA methods in both low-rank representations and structured pruning domains to demonstrate the superiority of our method. Notably, our model with different compression levels can consistently outperform prior arts.

Table 2: Pruning results of ResNet-50 on ImageNet

Model	Top1	Top5	Params (↓%)	MACs (↓%)
ResNet-50	76.15	92.87	25.55M(00.0)	4.11B(00.0)
CLR-RNF-0.2 [19]	74.85	92.31	16.92M(33.6)	2.45B(40.1)
DECORE-8 [1]	76.31	93.02	22.69M(11.0)	3.54B(13.4)
CHIP [34]	76.30	93.02	15.10M(40.8)	2.26B(44.8)
CORING-V (Ours)	**76.78**	**93.23**	**15.10M(40.8)**	**2.26B(44.8)**
DECORE-6 [1]	74.58	92.18	14.10M(44.7)	2.36B(42.3)
SCOP [35]	75.95	92.79	14.59M(42.8)	2.24B(45.3)
Zhang et. *al* [44]	75.83	92.76	14.23M(44.2)	2.10B(48.7)
CORING-C (Ours)	**76.34**	**93.06**	**14.23M(44.2)**	**2.10B(48.7)**
EZCrop [23]	74.33	92.00	11.05M(56.7)	1.52B(62.8)
RASP-1G [45]	74.48	92.02	16.29M(36.3)	1.50B(63.6)
Zhang et. *al* [44]	74.80	92.39	11.04M(56.7)	1.52B(62.8)
CORING-V (Ours)	**75.55**	**92.61**	**11.04M(56.7)**	**1.50B(63.6)**
CLR-RNF-0.44 [19]	72.67	91.09	9.00M(64.7)	1.23B(69.9)
Zhang et. *al* [44]	73.18	91.32	8.01M(68.6)	0.95B(76.7)
HRel-3 [32]	73.67	91.70	9.10M(64.4)	1.38B(66.4)
CORING-V (Ours)	**73.99**	**91.71**	**8.01M(68.6)**	**0.95B(76.7)**

3.2. Approach

Fig. 2 illustrates the overall pipeline of our approach, which comprises two main phases: decomposition and pruning.

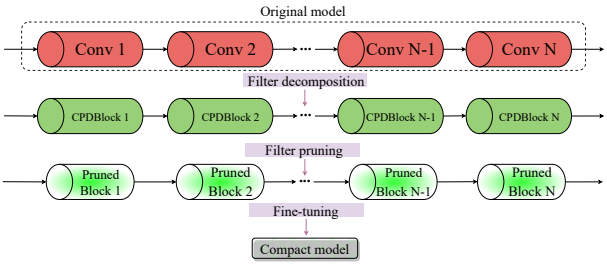

Figure 2: Graphic illustration of the NORTON approach.

3.2.1. Filters Decomposition using the CP decomposition

Consider a convolutional layer with O filters of size $K_h \times K_w \times I$, where I represents the number of input channels, and K_h and K_w represent the height and width of the kernel, respectively. The weight tensor $\boldsymbol{\mathcal{W}}$ can be represented as a 4-order tensor of size $K_h \times K_w \times I \times O$. Alternatively, it can be viewed as a set of O individual 3-order filters denoted as $\{\boldsymbol{\mathcal{W}}_1, \boldsymbol{\mathcal{W}}_2, \ldots, \boldsymbol{\mathcal{W}}_O\}$. It is worth noting that the k-th filter can be obtained by extracting the sub-tensor $\boldsymbol{\mathcal{W}}_{:,:,:,k}$, which is equivalent to $\boldsymbol{\mathcal{W}}_k$. These weights map an input tensor $\boldsymbol{\mathcal{I}}$ of size $H_{in} \times W_{in} \times I$ into an output tensor $\boldsymbol{\mathcal{O}}$ of size $H_{out} \times W_{out} \times O$, where H_{in}, W_{in}, H_{out}, and W_{out} are the height and width of the input and output tensors, respectively. For simplicity, the dimension of the batch is disregarded.

In convolutional neural networks, the mapping of the input tensor $\boldsymbol{\mathcal{I}}$ to the output tensor $\boldsymbol{\mathcal{O}}$ is achieved through the convolution operation. This convolution is given by the following expression:

$$\boldsymbol{\mathcal{O}}_k(i,j) = \sum_{m=0}^{K_h-1} \sum_{n=0}^{K_w-1} \sum_{p=0}^{I-1} \boldsymbol{\mathcal{I}}(i+m, j+n, p) \cdot \boldsymbol{\mathcal{W}}_k(m,n,p), \tag{5}$$

where, for $0 \leq k \leq O-1$, $\boldsymbol{\mathcal{O}}_k = \boldsymbol{\mathcal{O}}_{:,:,k}$, and is of size $H_{out} \times W_{out}$. Based on (5) and the CPD definition in [13], we can apply the CPD to each individual filter $\boldsymbol{\mathcal{W}}_k$ in order to obtain a compact representation. By decomposing $\boldsymbol{\mathcal{W}}_k$

using CPD, we have

$$\boldsymbol{\mathcal{W}}_k(m,n,p) = \sum_{r=0}^{R-1} \mathbf{A}_k(m,r) \cdot \mathbf{B}_k(n,r) \cdot \mathbf{C}_k(p,r), \tag{6}$$

where $\mathbf{A}_k$, $\mathbf{B}_k$ and $\mathbf{C}_k$ are 3 factor matrices of size $K_h \times R$, $K_w \times R$ and $I \times R$, respectively.

By substituting (6) into (5), we obtain a new CPD-based approach to compute the convolution. This approach involves a sequence of mappings using the factor matrices instead of high-order tensors. The resulting equation for this CPD-based convolution is:

$$\begin{aligned}\boldsymbol{\mathcal{O}}_k(i,j) = \sum_{m=0}^{K_h-1} \sum_{n=0}^{K_w-1} \sum_{p=0}^{I-1} \sum_{r=0}^{R-1} \boldsymbol{\mathcal{I}}(i+m, j+n, p) \cdot \\ \mathbf{A}_k(m,r) \cdot \mathbf{B}_k(n,r) \cdot \mathbf{C}_k(p,r).\end{aligned} \tag{7}$$

Starting from (7), we observe that the CPD-based convolution involves element-wise multiplications between the input tensor $\boldsymbol{\mathcal{I}}$ and the factor matrices $\mathbf{A}_k$, $\mathbf{B}_k$, and $\mathbf{C}_k$. It is important to note that the order of the convolutions can be rearranged without affecting the final result. This flexibility allows us to describe the computation as a sequential block of convolutions with smaller kernels, followed by a summation. First, the input tensor $\boldsymbol{\mathcal{I}}$ can be convolved with the kernel $\mathbf{C}_k$ along the input channel dimension. The resulting tensor is then convolved with the kernel $\mathbf{B}_k$ along the spatial dimensions. Finally, the output of the second convolution is convolved with the kernel $\mathbf{A}_k$ along the spatial dimensions. The outputs of these convolutions are summed up to compute the final output tensor $\boldsymbol{\mathcal{O}}_k$. The following set of equations captures these operations in a concise manner:

$$\boldsymbol{\mathcal{O}}_k^{\mathbf{C}}(i+m, j+n, r) = \sum_{p=0}^{I-1} \boldsymbol{\mathcal{I}}(i+m, j+n, p) \cdot \mathbf{C}_k(p,r), \tag{8}$$

where $\boldsymbol{\mathcal{O}}_k^{\mathbf{C}}$ is of size $H_{in} \times W_{in} \times R$.

$$\boldsymbol{\mathcal{O}}_k^{\mathbf{B}}(i+m, j, r) = \sum_{n=0}^{K_w-1} \boldsymbol{\mathcal{O}}_k^{\mathbf{C}}(i+m, j+n, r) \cdot \mathbf{B}_k(n,r), \tag{9}$$

where $\boldsymbol{\mathcal{O}}_k^{\mathbf{B}}$ is of size $H_{in} \times W_{out} \times R$.

$$\boldsymbol{\mathcal{O}}_k^{\mathbf{A}}(i,j,r) = \sum_{m=0}^{K_h-1} \boldsymbol{\mathcal{O}}_k^{\mathbf{B}}(i+m, j, r) \cdot \mathbf{A}_k(m,r), \tag{10}$$

where $\boldsymbol{\mathcal{O}}_k^{\mathbf{A}}$ is of size $H_{out} \times W_{out} \times R$. Finally, we have

$$\boldsymbol{\mathcal{O}}_k(i,j) = \sum_{r=0}^{R-1} \boldsymbol{\mathcal{O}}_k^{\mathbf{A}}(i,j,r). \tag{11}$$

3.2.2. CPDBlock pruning

The core idea of CPDBlock pruning is to construct a distance matrix $\mathbf{D}$, where each element $\mathbf{D}_{ij}$ corresponds to the distance between the factor matrices $[\![\mathbf{A}_i, \mathbf{B}_i, \mathbf{C}_i]\!]$ and $[\![\mathbf{A}_j, \mathbf{B}_j, \mathbf{C}j]\!]$. The pruning process involves iteratively identifying the pair of decompositions, i and j, corresponding to the minimum value of $\mathbf{D}_{ij}$, and removing one of them. The removed decomposition in our strategy between i and j is the one that most closely resembles the rest of the decompositions, ensuring that the pruned one retains the representation that is most similar to the remaining ones. The distance matrix $\mathbf{D} \in \mathbb{R}^{O \times O}$ can be expressed as

$$\mathbf{D}_{ij} = \alpha \mathbf{D}_{ij}^{\mathbf{A}} + \beta \mathbf{D}_{ij}^{\mathbf{B}} + \gamma \mathbf{D}_{ij}^{\mathbf{C}}, \tag{12}$$

where $\mathbf{D}_{ij}^{A} = \phi(\mathbf{A}_i, \mathbf{A}_j)$ (similarly for $\mathbf{D}_{ij}^{B}$ and $\mathbf{D}_{ij}^{C}$), and α, β and γ are weight parameters, whose sum is equal to 1.

3.3. Experiments

VGG-16-BN. Tab. 3 shows compression results of VGG on CIFAR-10. In all compression levels, compared with other methods, NORTON consistently achieves the highest accuracy while reducing much more computation costs and enjoying a similar number of parameters.

Table 3: Compression results of VGG-16-BN on CIFAR

Model	Top-1 (%)	MACs (↓%)	Params. (↓%)
VGG-16-BN [33]	93.96	313.73M (00)	14.98M (00)
EZCrop [23]	93.01	131.17M (58)	2.76M (82)
DECORE-500 [1]	94.02	203.08M (35)	5.54M (63)
AutoBot [2]	94.19	145.61M (54)	7.53M (50)
NORTON (Ours)	**94.45**	**126.49M (60)**	**2.58M (83)**
EZCrop [23]	93.70	104.78M (67)	2.50M (83)
CHIP [34]	93.72	104.78M (67)	2.50M (83)
AutoBot [2]	94.01	108.71M (65)	6.44M (57)
NORTON (Ours)	**94.16**	**101.91M (68)**	**2.34M (84)**
AutoBot [2]	93.62	72.60M (77)	5.51M (63)
NORTON (Ours)	**94.11**	**74.14M (77)**	**3.60M (76)**
Lebedev *et al.* [14]	93.07	68.53M (78)	3.22M (78)
HALOC [38]	93.16	43.92M (86)	0.30M (98)
FSM [4]	93.73	106.67M (66)	2.10M (86)
NORTON (Ours)	**93.84**	**37.68M (88)**	1.94M (87)
HRank-3 [20]	91.23	73.70M (77)	1.78M(92)
DECORE-50 [1]	91.68	36.85M (88)	0.26M (98)
NORTON (Ours)	**92.54**	**13.54M (96)**	**0.24M (98)**
NORTON (Ours)	**90.32**	**4.58M (99)**	**0.14M (99)**

ResNet-50. To evaluate the scalability of NORTON, we perform experiments on the extensive ImageNet dataset using the ResNet-50 architecture, as enumerated in Tab. 4. Across all evaluated scenarios, NORTON consistently outperforms other approaches in terms of both performance and complexity reduction.

Table 4: Compression results of ResNet-50 on ImageNet

Model	Top-1	Top-5	MACs(↓%)	Params(↓%)
ResNet-50 [10]	76.15	92.87	4.09G(00)	25.50M(00)
CLR-RNF-0.2 [19]	74.85	92.31	2.45G(40)	16.92M(34)
FilterSketch-0.7 [18]	75.22	92.41	2.64G(36)	16.95M(33)
DECORE-8 [1]	76.31	93.02	3.54G(13)	22.69M(11)
NORTON (Ours)	**76.91**	**93.57**	**2.32G(43)**	**14.51M(43)**
DECORE-6 [1]	74.58	92.18	2.36G(42)	14.10M(45)
EZCrop [23]	75.68	92.70	2.26G(45)	15.09M(41)
SCOP [35]	75.95	92.79	2.24G(45)	14.59M(43)
NORTON (Ours)	**76.58**	**93.43**	**2.08G(50)**	**13.51M(47)**
EZCrop [23]	74.33	92.00	1.52G(63)	11.05M(57)
CC-0.6 [17]	74.54	92.25	1.53G(63)	10.58M(59)
EDP [30]	75.34	92.43	1.92G(53)	14.28M(44)
NORTON (Ours)	**75.95**	**92.91**	**1.49G(64)**	**10.52M(59)**
DECORE-5 [1]	72.06	90.82	1.60G(61)	8.87M(65)
ABCPruner-50% [21]	72.58	90.91	1.30G(68)	9.10M(64)
CLR-RNF-0.44 [19]	72.67	91.09	1.23G(70)	9.00M(65)
NORTON (Ours)	**74.00**	**92.00**	**0.96G(77)**	**7.96M(69)**
FilterSketch-0.2 [18]	69.43	89.23	0.93G(77)	7.18M(72)
DECORE-4 [1]	69.71	89.37	1.19G(71)	6.12M(76)
CURL [25]	73.39	91.46	1.11G(73)	6.67M(74)
NORTON (Ours)	**73.65**	**91.64**	**0.92G(78)**	**5.88M(77)**

4. Conclusion and Future Works

In this report, we have embarked on a journey to enhance the efficiency and accessibility of deep learning-based systems. Our contributions during this first year of Ph.D. research are represented by the novel methods CORING and NORTON for network pruning and tensor decomposition. These approaches have demonstrated their effectiveness in achieving model compression while preserving critical features and maintaining performance. CORING's tensor decomposition approach and K-shots pruning strategy offer a versatile solution, while NORTON's combination of tensor decompositions and structured pruning provides fine-grained control and scalability.

As we look ahead, numerous exciting avenues for future research beckon:

- Deeper Exploration in Pruning/Decomposition: Building on our initial work, there's room for delving deeper into the pruning and tensor decomposition domain. Developing more advanced techniques and algorithms could involve exploring different criteria for pruning or devising more efficient decomposition methods.
- Broadening to Other Decomposition Techniques: Extending our research to encompass a wider range of model compression techniques beyond pruning and tensor decomposition is a promising direction. Investigating methods like quantization, knowledge distillation, and neural architecture search (NAS) will allow us to explore their potential in reducing model size while maintaining performance.
- Widening to Other Model Architectures: Applying compression techniques to various neural network architectures other than CNNs opens up exciting possibilities. Experimenting with models like Transformers (used in natural language processing), recurrent neural networks (RNNs), graph neural networks (GNNs), and large language models (LLMs) will help assess the effectiveness of our methods in different domains.
- Cross-Domain Transfer: Exploring the potential for transferring insights and techniques from one domain (e.g., computer vision) to another (e.g., natural language processing) is a valuable pursuit. Investigating how compression methods developed for one domain can be adapted and applied to other domains can lead to more efficient and versatile AI models.
- Application to Maritime Surveillance: Given our background in maritime surveillance, considering how model compression techniques can be specifically tailored to enhance AI applications in this domain is crucial. Developing specialized models and compression strategies to address the unique challenges and requirements of maritime surveillance, such as real-time processing and anomaly detection, holds significant promise.
- Robustness and Security: Investigating the robustness and security implications of compressed models is essential. Analyzing the vulnerability of compressed models to adversarial attacks and exploring methods to improve their resilience are important research directions.
- Automated Compression Frameworks: Developing automated frameworks that intelligently select and apply compression techniques based on the specific requirements and constraints of a given task or deployment environment is a forward-looking endeavor. This can lead to more adaptive and efficient model compression.
- Explainability and Interpretability: Exploring methods to enhance the interpretability and explainability of compressed models is vital. Making it easier to understand their decisions and behavior is crucial, especially for safety-critical applications.
- Efficient Deployment: Investigating strategies for efficiently deploying compressed models on a variety of hardware platforms, including resource-constrained devices, edge devices, and cloud servers, will be essential to ensure practicality.
- Collaborative Research: Collaborating with experts in related fields such as robotics, environmental science, or healthcare to apply model compression techniques to address specific domain challenges and make AI more accessible and impactful in those areas.

These diverse research directions collectively contribute to the ongoing advancement of efficient deep learning models and their broader applicability across domains and scenarios.

In conclusion, the journey toward making deep learning more efficient and accessible is ongoing. Our contributions during this first year of Ph.D. research mark the beginning of this exploration. As we move forward, we remain committed to advancing the field, fostering collaboration, and unlocking the full potential of deep learning-based systems.

Appendix A. Lists of publications

1. [27] Pham Van Tien, Zniyed Yassine, Nguyen Thanh Phuong, 2023. Élagage efficace des filtres basé sur les décompositions tensorielles, s, in: XXIX‘eme Colloque Francophone de Traitement du Signal et des Images, GRETSI - Groupe de Recherche en Traitement du Signal et des Images, Grenoble. pp. p. 937–940

2. [29] Pham Van Tien, Zniyed Yassine, Nguyen Thanh Phuong, 2023. Enhanced network compression through tensor decompositions and pruning, IEEE Transactions on Neural Networks and Learning Systems, *under review*.

3. [28] Pham Van Tien, Zniyed Yassine, Nguyen Thanh Phuong, 2023. Efficient tensor-based filter pruning, Neural Networks, *to be submitted*.

References

[1] Alwani, M., Madhavan, V., Wang, Y., 2022. Decore: Deep compression with reinforcement learning. CVPR .
[2] Castells, T., Yeom, S.K., 2023. Automatic neural network pruning that efficiently preserves the model accuracy, in: AAAI.
[3] De Lathauwer, L., De Moor, B., Vandewalle, J., 2000. A multilinear singular value decomposition. SIAM Journal on Matrix Analysis and Applications , 1253–1278.
[4] Duan, Y., Zhou, Y., He, P., Liu, Q., Duan, S., Hu, X., 2022. Network pruning via feature shift minimization, in: Proceedings of the Asian Conference on Computer Vision, pp. 4044–4060.
[5] Elsken, T., Metzen, J.H., Hutter, F., 2019. Neural Architecture Search. Springer International Publishing, Cham. pp. 63–77.
[6] Golub, G.H., Van Loan, C.F., 1996. Matrix Computations. Third ed., The Johns Hopkins University Press.
[7] Goyal, S., Roy Choudhury, A., Sharma, V., 2019. Compression of deep neural networks by combining pruning and low rank decomposition, in: IPDPSW.
[8] Han, S., Pool, J., Tran, J., Dally, W.J., 2015. Learning both weights and connections for efficient neural networks. NIPS .
[9] Harshman, R.A., 1970. Foundations of the parafac procedure: Models and conditions for an ’explanatory’ multi-modal factor analysis. UCLA Working Papers in Phonetics 16, 1–84.
[10] He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition, in: CVPR.
[11] Hinton, G., Vinyals, O., Dean, J., 2015. Distilling the knowledge in a neural network, in: NIPS Deep Learning and Representation Learning Workshop.
[12] Jaderberg, M., Vedaldi, A., Zisserman, A., 2014. Speeding up convolutional neural networks with low rank expansions.
[13] Kolda, T.G., Bader, B.W., 2009. Tensor decompositions and applications. SIAM Review 51, 455–500.
[14] Lebedev, V., Ganin, Y., Rakhuba, M., Oseledets, I.V., Lempitsky, V.S., 2015. Speeding-up convolutional neural networks using fine-tuned cp-decomposition, in: ICLR.
[15] Li, H., Kadav, A., Durdanovic, I., Samet, H., Graf, H.P., 2016. Pruning filters for efficient convnets. ICLR .
[16] Li, Y., Gu, S., Mayer, C., Van Gool, L., Timofte, R., 2020. Group sparsity: The hinge between filter pruning and decomposition for network compression, in: CVPR.
[17] Li, Y., Lin, S., Liu, J., Ye, Q., Wang, M., Chao, F., Yang, F., Ma, J., Tian, Q., Ji, R., 2021. Towards compact cnns via collaborative compression, in: CVPR, pp. 6434–6443.
[18] Lin, M., Cao, L., Li, S., Ye, Q., Tian, Y., Liu, J., Tian, Q., Ji, R., 2022a. Filter sketch for network pruning. TNNLS .
[19] Lin, M., Cao, L., Zhang, Y., Shao, L., Lin, C.W., Ji, R., 2022b. Pruning networks with cross-layer ranking & k-reciprocal nearest filters. IEEE Transactions on Neural Networks and Learning Systems , 1–10.
[20] Lin, M., Ji, R., Wang, Y., Zhang, Y., Zhang, B., Tian, Y., Shao, L., 2020. Hrank: Filter pruning using high-rank feature map. CVPR .
[21] Lin, M., Ji, R., Zhang, Y., Zhang, B., Wu, Y., Tian, Y., 2021. Channel pruning via automatic structure search, in: Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence.
[22] Lin, M., Zhang, Y., Li, Y., Chen, B., Chao, F., Wang, M., Li, S., Tian, Y., Ji, R., 2023. 1xn pattern for pruning convolutional neural networks. IEEE Transactions on Pattern Analysis and Machine Intelligence 45, 3999–4008.
[23] Lin, R., Ran, J., Wang, D., Chiu, K., Wong, N., 2022c. Ezcrop: Energy-zoned channels for robust output pruning, in: WACV, pp. 3595–3604.
[24] Liu, J., Zhuang, B., Zhuang, Z., Guo, Y., Huang, J., Zhu, J., Tan, M., 2022. Discrimination-aware network pruning for deep model compression. IEEE Transactions on Pattern Analysis and Machine Intelligence 44, 4035–4051.
[25] Luo, J.H., Wu, J., 2020. Neural network pruning with residual-connections and limited-data, in: CVPR, pp. 1455–1464.
[26] Ogueji, K., Ahia, O., Onilude, G., Gehrmann, S., Hooker, S., Kreutzer, J., 2022. Intriguing properties of compression on multilingual models, in: Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Abu Dhabi, United Arab Emirates. pp. 9092–9110.
[27] Pham, V.T., Zniyed, Y., Nguyen, T.P., 2023. Élagage efficace des filtres basé sur les décompositions tensorielles, in: XXIXème Colloque Francophone de Traitement du Signal et des Images, GRETSI - Groupe de Recherche en Traitement du Signal et des Images, Grenoble. pp. p. 937–940. URL: https://gretsi.fr/data/colloque/pdf/2023_pham1312.pdf.
[28] Pham, V.T., Zniyed, Y., Nguyen, T.P., 2024a. Efficient tensor decomposition-based filter pruning. Neural Networks, to be submitted URL: https://github.com/pvti/CORING.
[29] Pham, V.T., Zniyed, Y., Nguyen, T.P., 2024b. Enhanced network compression through tensor decompositions and pruning. IEEE transactions on neural networks and learning systems, Under review URL: https://github.com/pvti/NORTON.

[30] Ruan, X., Liu, Y., Yuan, C., Li, B., Hu, W., Li, Y., Maybank, S., 2021. Edp: An efficient decomposition and pruning scheme for convolutional neural network compression. TNNLS .
[31] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M.S., Berg, A.C., Fei-Fei, L., 2014. Imagenet large scale visual recognition challenge. IJCV .
[32] Sarvani, C., Ghorai, M., Dubey, S.R., Basha, S.S., 2022. Hrel: Filter pruning based on high relevance between activation maps and class labels. Neural Networks 147, 186–197.
[33] Simonyan, K., Zisserman, A., 2015. Very deep convolutional networks for large-scale image recognition. ICLR .
[34] Sui, Y., Yin, M., Xie, Y., Phan, H., Zonouz, S., Yuan, B., 2021. Chip: Channel independence-based pruning for compact neural networks, in: NeurIPS.
[35] Tang, Y., Wang, Y., Xu, Y., Tao, D., XU, C., Xu, C., Xu, C., 2020. Scop: Scientific control for reliable neural network pruning, in: NeurIPS.
[36] Tucker, L.R., 1966. Some mathematical notes on three-mode factor analysis. Psychometrika 31, 279–311.
[37] Tung, F., Mori, G., 2020. Deep neural network compression by in-parallel pruning-quantization. IEEE Transactions on Pattern Analysis and Machine Intelligence 42, 568–579.
[38] Xiao, J., Zhang, C., Gong, Y., Yin, M., Sui, Y., Xiang, L., Tao, D., Yuan, B., 2023. Haloc: Hardware-aware automatic low-rank compression for compact neural networks, in: AAAI.
[39] Yang, H., Liang, Y., Liu, W., Meng, F., 2023. Filter pruning via attention consistency on feature maps. Applied Sciences .
[40] Yeom, S.K., Shim, K.H., Hwang, J.H., 2022. Toward compact deep neural networks via energy-aware pruning. tinyML .
[41] Yin, M., Sui, Y., Liao, S., Yuan, B., 2021. Towards efficient tensor decomposition-based dnn model compression with optimization framework, in: 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 10669–10678.
[42] Yin, Y., Cheng, X., Shi, F., Zhao, M., Li, G., Chen, S., 2022. An enhanced lightweight convolutional neural network for ship detection in maritime surveillance system. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing 15, 5811–5825.
[43] Yuan, T., Deng, W., Tang, J., Tang, Y., Chen, B., 2019. Signal-to-noise ratio: A robust distance metric for deep metric learning. CVPR .
[44] Zhang, S., Gao, M., Ni, Q., Han, J., 2023. Filter pruning with uniqueness mechanism in the frequency domain for efficient neural networks. Neurocomputing 530, 116–124.
[45] Zhen, C., Zhang, W., Mo, J., Ji, M., Zhou, H., Zhu, J., 2023. Rasp: Regularization-based amplitude saliency pruning. Neural Networks .