

Full Length Article

A novel approach to lossless convolutional neural network compression via progressive knowledge distillation-incorporated low-rank compression ^{*}

Yaping He ^{id a}, Hao Wu ^{id a,*}, Weibo Liu ^{id b,*}, Xin Luo ^{id a}

^a College of Computer and Information Science, Southwest University, Chongqing, 400715, China

^b Department of Computer Science, Brunel University of London, Uxbridge, Middlesex, UB8 3PH, United Kingdom

ARTICLE INFO

Keywords:

Model compression
Knowledge distillation
Progressive learning
Dynamic temperature
Residual compensation

ABSTRACT

Model compression is widely used to deploy large neural networks on resource-constrained edge devices. Among existing techniques, low-rank composition is theoretically grounded in approximation theory and provides a strong basis for preserving model performance after compression. However, in practice, even state-of-the-art low-rank methods such as Tucker, Tensor Train, and Tensor Ring decomposition may still introduce substantial reconstruction errors. These errors are difficult to recover using conventional fine-tuning strategies, especially under high compression ratios. To address this issue, this paper proposes Progressive Knowledge Distillation-Incorporated Low-Rank Compression (PKD-LRC). The proposed framework introduces three key components. First, a residual-compensation mechanism is incorporated into convolutional compression to reduce information loss. Second, a dynamic temperature adjustment strategy is used to improve the fine-tuning of low-rank models and enhance prediction accuracy. Third, a progressive learning paradigm is developed to balance compression ratio and classification performance. Theoretical analysis is conducted to investigate the effectiveness of the residual compensation mechanism for compressing convolutional neural networks (CNNs). The experimental results on the CIFAR-100 dataset show that PKD-LRC achieves a 0.35% higher Top-1 accuracy with a 7.75× compression ratio compared to the uncompressed VGG-16 model. On the ImageNet dataset, the proposed method yields a 0.20% Top-1 accuracy gain with a 2.78× compression ratio over the baseline ResNet-18 model. Above results indicate that PKD-LRC effectively mitigates reconstruction errors and enhances the performance of compressed models through adaptive fine-tuning.

1. Introduction

Over the past few years, convolutional neural networks (CNNs) (Song et al., 2025) have emerged as a core technique in many areas. They are especially effective in computer vision, powering tasks such as image classification (Chen et al., 2024; Yin et al., 2021), object detection (Chen et al., 2025; Yuan et al., 2024), and semantic segmentation (Su et al., 2024). In addition, they have also been extended to multimodal learning scenarios, where information from multiple modalities is jointly exploited to improve performance (Zhan et al., 2025). By leveraging multiple convolutional layers and nonlinear transformations, CNNs can automatically extract high-level spatial features, which significantly enhances the performance of learning models. The application of CNNs is no longer limited to computer vision. For example, CNNs have demonstrated promising performance in natural language processing (e.g., text categorization and sentiment detection), where fast process-

ing and parallel computation are crucial. Moreover, CNNs are capable of effectively modelling multimodal information with strong adaptability, thereby facilitating speech recognition, recommender systems, and autonomous driving. Overall, CNNs continue to play a vital role in advancing artificial intelligence technologies (Xue & Shen, 2024).

Despite their excellent performance in various domains, CNNs pose practical challenges in resource-constrained settings due to their substantial parameter counts. For many downstream applications, achieving strong performance with CNNs typically involves models with tens to hundreds of millions of parameters, which in turn incurs heavy computation during training and inference. The large model size significantly increases the cost of training and updating, making it difficult to efficiently optimize and maintain such models in real-world scenarios. For example, ResNet-50 has 23 million parameters, whereas recent models like ConvNeXt (Yu et al., 2024) contain billions of parameters. In addition to the high training cost, the massive parameter scale

^{*} This work was supported in part by the Royal Society of the UK under Grant IES\R3\243021, and the National Natural Science Foundation of China under Grant 62302402.

^{*} Corresponding authors.

E-mail addresses: hyp15823653754@email.swu.edu.cn (Y. He), haowuf@swu.edu.cn (H. Wu), Weibo.Liu2@brunel.ac.uk (W. Liu), luoxin@swu.edu.cn (X. Luo).

<https://doi.org/10.1016/j.neunet.2026.109631>

Received 9 December 2025; Received in revised form 19 August 2026; Accepted 9 September 2026

Available online 12 September 2026

0893-6080/© 2026 The Author(s). Published by Elsevier Ltd. This is an open access article under the CC BY license (<http://creativecommons.org/licenses/by/4.0/>).

also creates deployment challenges. The large size of CNNs significantly increases storage and memory requirements. The high computational requirements of large CNN architectures also pose significant obstacles to their deployment on constrained hardware, including mobile phones, drones, and various embedded platforms, all of which operate under strict memory and power limits. Consequently, model compression has become a central strategy to address these challenges.

Among various model compression techniques, low-rank decomposition stands out due to its strong compression capability and solid theoretical foundations in approximation theory. Matrix-based methods, such as Truncated Singular Value Decomposition (TSVD) (Ruan et al., 2020), compress convolutional kernels by reshaping high-order tensors into matrices, but this process may discard their inherent multi-dimensional structures and inter-mode correlations. Tensor decomposition methods, such as Canonical Polyadic (CP) (Phan et al., 2020) and Tucker decompositions (Dai et al., 2021), better preserve tensor structures. However, CP-based compression is often sensitive to rank selection and may suffer from unstable optimization under aggressive compression, whereas the dense core tensor in Tucker-based methods may limit the achievable compression ratio. Tensor Train (TT) (Senderovich et al., 2022) and Tensor Ring (TR) (Xie et al., 2024) further reduce parameter redundancy through sequential factorization, but their strongly constrained representations may restrict the expressive flexibility of convolutional kernels in low-rank settings. Despite these differences, the above methods mainly improve the retained low-rank representation and generally replace the original weights with a single low-rank approximation. Therefore, the residual information discarded during rank truncation is rarely modeled or compensated for explicitly.

Building upon low-rank decomposition-based compression, recent studies have explored hybrid compression frameworks to overcome the limitations of relying on a single compression technique. One line of research treats low-rank decomposition as the primary compression mechanism and incorporates knowledge distillation into the subsequent fine-tuning stage. Specifically, knowledge distillation has been employed in Hajimolahoseini et al. (2022) and Zeng et al. (2024) to recover the predictive performance of low-rank models after decomposition. However, in these methods, knowledge distillation mainly serves as an auxiliary recovery strategy and generally follows a predefined temperature configuration, without explicitly adapting the knowledge-transfer process to the evolving representational capacity and learning state of the compressed student. Another line of research takes knowledge distillation as the primary framework and applies low-rank decomposition to the teacher model. In Dai et al. (2021) and Sepahvand et al. (2022), the teacher network has been compressed to reduce the capacity and structural gaps between the teacher and student, thereby facilitating knowledge transfer. Nevertheless, such approaches mainly improve teacher-student compatibility through teacher compression, while the distillation process itself still lacks a dynamic mechanism for adapting knowledge transfer to the evolving representational capacity and learning state of the compressed student. Therefore, existing hybrid frameworks remain insufficiently adaptive during the post-compression recovery process, particularly under high compression ratios.

Based on the above analysis, two key challenges remain to be addressed.

1. *Lack of an innovative error compensation mechanism:* The approximation error induced by low-rank composition is neither explicitly modeled nor effectively compensated, which often leads to noticeable performance degradation.
2. *Lack of effective accuracy-recovery strategies:* Low-rank compression inevitably causes performance degradation, and conventional fine-tuning methods based on static-temperature distillation cannot sufficiently accommodate the evolving representational capacity and learning state of the compressed student.

To address the aforementioned critical challenges, a Progressive Knowledge Distillation-incorporated Low-Rank Compression (PKD-

LRC) method is proposed in this paper, drawing on ideas from transfer learning (Dai et al., 2021; Hu et al., 2024; Yang et al., 2024) and the residual compensation mechanism (RCM) (Bai et al., 2025; Wang et al., 2024). Specifically, the proposed PKD-LRC method is developed based on three key components:

1. a Residual-compensated Tucker-2 (ResTucker-2) compression framework for enhancing the performance of low-rank compression models;
2. a dynamic temperature adjustment scheme for fine-grained optimization and improved accuracy recovery; and
3. a growth mechanism for balancing compression ratio and predictive performance more effectively.

The main contributions of this paper are summarized as follows:

1. *A novel low-rank compression framework.* A high-accuracy and parameter-efficient CNN compression framework is developed for deployment on resource-constrained devices such as drones, robotic platforms, and other mobile intelligent systems.
2. *Theoretical analysis of residual compensation.* A theoretical guarantee is established to show that the proposed residual-compensated decomposition yields a strictly tighter reconstruction error bound than conventional Tucker-2 decomposition, thereby ensuring that the residual compensation step does not increase the approximation error.
3. *Empirical validation across multiple benchmarks.* Extensive experiments are conducted on five benchmark datasets, demonstrating that the proposed method outperforms some state-of-the-art low-rank and distillation-based compression methods in image classification tasks.

The rest of this paper is organized as follows. Section 2 discusses the related work. Section 3 introduces the preliminaries. Section 4 presents the methodology for the proposed PKD-LRC method. Experimental settings and results are introduced in Section 5. Finally, Section 6 summarizes the findings of this paper and points out potential future research directions.

2. Related work

2.1. Pruning

Model pruning has gained widespread attention as an effective model compression methods. Pruning approaches can be broadly grouped into unstructured and structured variants. The unstructured type eliminates weight elements one by one to produce sparse models, although real speedup often requires dedicated hardware support (Cheng et al., 2024). In contrast, structured pruning removes entire rows or columns of weights, which helps reduce storage and accelerates inference while maintaining hardware friendliness (Shuvo et al., 2022). However, traditional structured pruning methods often rely on retraining to recover model performance, which incurs significant computational cost for large-scale language models (LLMs) (An et al., 2024). To address this issue, researchers have proposed various pruning methods that do not require retraining. For example, FLAP (An et al., 2024b) has introduced volatility metrics, adaptive global compression structures, and bias compensation mechanisms, enabling efficient structured pruning of LLMs without retraining. Similarly, FASP (Zayed et al., 2024) has developed a fairness- and performance-oriented structured pruning method, selectively removing attention heads that negatively impact fairness while preserving critical performance, by quantifying each head's contribution to both fairness and language modeling. In the federated learning context, Gao et al. (2024) have proposed a device-level network pruning approach that generates device-specific sub-networks via supernetworks and embedding layers, facilitating efficient pruning under heterogeneous data distributions. Although pruning-based approaches

effectively reduce model size and computational cost, they mainly operate at the structural level by removing weights, filters, or channels. As a result, the remaining parameters are still represented by dense weight matrices or tensors. This means that pruning methods do not explicitly address the intrinsic redundancy within the retained parameters. In contrast, low-rank composition focuses on modeling and compressing this dense parameter redundancy by exploiting the low-dimensional structure of convolutional kernels.

2.2. Knowledge distillation

Knowledge Distillation (KD) serves as an effective compression strategy by passing the knowledge of a high-capacity teacher network to a more compact student network, making it feasible to deploy strong models on resource-restricted hardware. Since Hinton et al. (2015) introduced the classic KD method in 2015, researchers have continuously explored ways to extract knowledge more effectively from teacher models. Most classical KD techniques work by encouraging the student to mimic the teacher's softened logits. This is achieved through a temperature parameter that expands the output distribution and reveals the teacher's hidden information. However, as research has advanced, it has become apparent that a single temperature parameter has been insufficient to handle the diversity of output distributions across different samples, resulting in over-softened or under-softened logits for certain samples, which has hampered the effectiveness of knowledge transfer. To address this, several improvements have emerged in recent years. For instance, DKD (Zhao et al., 2022) has decoupled the distillation loss into target-class loss and non-target-class loss, enabling the student model to learn both intra-class and inter-class relationships simultaneously. CTKD (Li et al., 2023b) has introduced a dynamic temperature mechanism, which adaptively has adjusted the temperature based on the differences in the output distributions of the teacher and student models. SDD (Wei et al., 2024) has decoupled the global logit output into multiple local logit branches and further has divided them into consistent and complementary knowledge, thereby enhancing the student model's ability to distinguish fine-grained and ambiguous samples. Additionally, CustomKD (Xiang et al., 2025) has performed customized alignment of features from large vision foundation models, effectively mitigating the structural discrepancy with edge models and significantly enhancing their performance in unlabeled or weakly labeled scenarios. Despite the significant progress in knowledge distillation, applying KD directly to low-rank-compressed models remains challenging. In low-rank compression, the student model is derived from a decomposed structure whose representational capacity is fundamentally constrained by the approximation error introduced during decomposition. Standard KD objectives mainly encourage output-level alignment between teacher and student models, but they do not explicitly compensate for the structural information loss caused by low-rank truncation. Consequently, the effectiveness of conventional KD strategies may be limited when applied to aggressively compressed low-rank models.

2.3. Low-rank composition

Low-rank composition has emerged as a crucial technique for compressing and accelerating CNNs, with recent studies having explored its potential from various perspectives and having formed interconnected advancements. Trained Rank Pruning (TRP) (Xu et al., 2020) has integrated low-rank approximation and regularization into the training process, eliminating the need for step-by-step SVD and simplifying the compression workflow. This approach has complemented strategies like singular value sparsification and singular vector orthogonality (Yang et al., 2020), further optimizing low-rank structures in models. Building on this, TT decomposition (Senderovich et al., 2022) has achieved compression while enhancing adversarial robustness by controlling the singular values of convolutional layers. To automate rank selection, the HALOC

framework (Xiao et al., 2023) has employed hardware-aware architecture search, while iterative adaptive rank reduction (Dai et al., 2025) has dynamically adjusted low-rank structures to improve compression flexibility and efficiency. Moreover, combining low-rank composition with other regularization techniques has demonstrated strong synergistic effects. The CEPD framework (Sui et al., 2024) has integrated structured sparsity with low-rank composition, significantly reducing computational costs while maintaining accuracy. Similarly, PELA (Guo et al., 2024) has incorporated feature distillation and weight perturbation regularization into low-rank compressed pre-trained models, achieving efficient compression alongside performance improvement. Despite the strong performance of these methods, the basic decomposition techniques can introduce significant reconstruction errors, which pose challenges for optimization. Although low-rank composition provides an effective way to compress dense convolutional kernels by exploiting tensor structures, aggressive rank truncation inevitably introduces reconstruction errors. These approximation errors accumulate across layers and may significantly degrade model performance. Existing approaches typically rely on fine-tuning or knowledge distillation to recover accuracy, yet they rarely explicitly model or compensate for the truncated residual information, leaving room for further improvement.

3. Preliminaries

In this paper, we denote an N -order tensor with bold uppercase letters, such as $\mathbf{X} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$. In addition, bold italic lowercase letters are used to represent vectors, such as $\mathbf{x}$, and regular uppercase letters for matrices, such as $\mathbf{X}$.

Definition 1 (Linear Layer Representation). Both convolutional layers and fully connected (FC) layers can be expressed as linear transformations with respect to their parameters. Specifically, given an input feature $\mathbf{X}$ and weight parameters $\mathbf{W}$, the output $\mathbf{Y}$ can be written as

$$\mathbf{Y} = \mathcal{F}(\mathbf{W}, \mathbf{X}), \quad (1)$$

where $\mathcal{F}(\cdot)$ denotes a linear operator corresponding to either convolution or matrix multiplication.

Definition 2 (Low-rank Approximation and Residual). Given a weight tensor or matrix $\mathbf{W}$, a low-rank approximation $\widehat{\mathbf{W}}$ is obtained by a decomposition method with rank constraint r . The corresponding residual is defined as

$$\mathbf{R} = \mathbf{W} - \widehat{\mathbf{W}}, \quad (2)$$

which captures the information discarded during rank truncation.

Definition 3 (Standard Knowledge Distillation). Given a training dataset $D = \{(x, y)\}$, a teacher model $f_{tea}(\cdot; \mathbf{W}_{tea})$ with parameters $\mathbf{W}_{tea}$, and a student model $f_{stu}(\cdot; \mathbf{W}_{stu})$ with parameters $\mathbf{W}_{stu}$, standard knowledge distillation trains the student model by minimizing a combination of the task-specific loss and a distillation loss:

$$\min_{\mathbf{W}_{stu}} \sum_{(x, y) \in D} \mathcal{L}_{task}(f_{stu}(x; \mathbf{W}_{stu}), y) + \alpha \mathcal{L}_{kd}(f_{tea}(x; \mathbf{W}_{tea}), f_{stu}(x; \mathbf{W}_{stu}), \tau), \quad (3)$$

where $\mathcal{L}_{task}$ denotes the cross-entropy loss for classification, $\mathcal{L}_{kd}$ is typically implemented as the Kullback–Leibler (KL) divergence between the softened output distributions of the teacher and student models, α balances the two loss terms, and τ is a fixed temperature hyperparameter used to smooth the logits during distillation.

4. Methodology

To shrink CNN models and lower their computational demands, yet still retain high predictive accuracy, we propose a unified compression strategy that integrates low-rank composition with knowledge distillation. This approach leverages the complementary strengths of structural

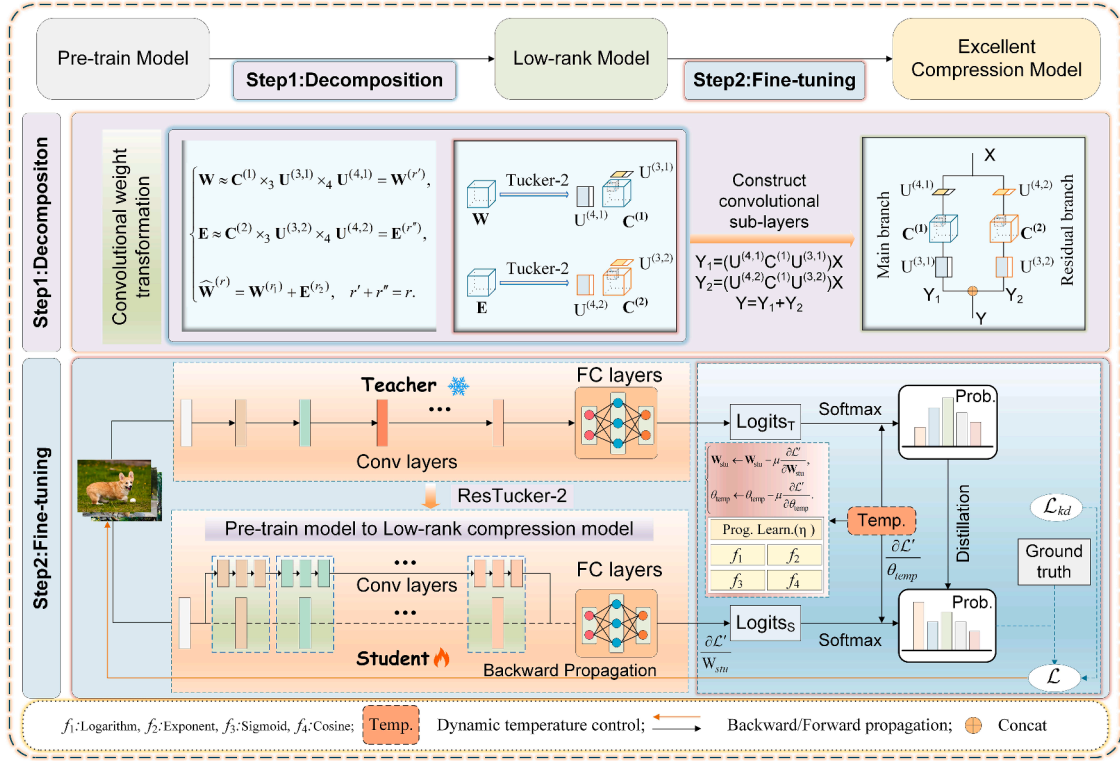


Fig. 1. PKD-LRC learning framework diagram. It consists of two main stages: decomposition and fine-tuning. In the first stage, RCM and Tucker-2 decompositions are employed to reduce approximation errors. In the second stage, a dynamic temperature knowledge distillation mechanism is designed to efficiently transfer the dark knowledge from the teacher model to the student model. During the distillation process, a progressive learning strategy is adopted to gradually update the temperature parameter, thereby enhancing the overall effectiveness of knowledge distillation.

compression and knowledge transfer, where low-rank factorization compacts the model and distillation compensates for potential accuracy loss. The proposed framework lays the foundation for the subsequent design of residual-compensated decomposition and dynamic, progressively guided fine-tuning.

4.1. Overview

Fig. 1 illustrates the overall framework of the PKD-LRC method. The framework consists of three sub-models: the Pre-trained Model, the Low-rank Model, and the Excellent Compression Model, and involves two main step: Decomposition and Fine-tuning. In Step 1, the RCM method is introduced into the decomposition process, forming a low-rank dual-branch structure that effectively reduces the approximation error caused by decomposition. In Step 2, the teacher model is employed to fine-tune the low-rank student model obtained in Step 1. Specifically, a dynamic temperature and progressive learning distillation strategy are adopted to further refine the low-rank student model. In brief, ResTucker-2 decomposes the residual tensor and constructs a new weight tensor to further reduce the approximation error. In addition, the fixed temperature is parameterized and updated through backpropagation. It is worth noting that the progressive learning strategy is applied only to the update of the temperature parameter rather than to all model parameters.

4.2. Low-rank compression framework

To further alleviate the information loss caused by truncation in Tucker-2 decomposition, this work designs a RCM, referred to as ResTucker-2 decomposition. The proposed approach preserves the computational efficiency and compactness of Tucker-2 while significantly improving the approximation fidelity of weight tensors.

4.2.1. Compressing the convolutional layer

Given a convolutional kernel tensor $\mathbf{W} \in \mathbb{R}^{D \times D \times M \times N}$, where D denotes the spatial kernel size and M, N are the numbers of input and output channels, respectively, we apply the Tucker-2 decomposition to compress the channel dimensions. The first-stage Tucker-2 decomposition is formulated as

$$\mathbf{W} \approx \mathbf{C}^{(1)} \times_3 \mathbf{U}^{(3,1)} \times_4 \mathbf{U}^{(4,1)} = \mathbf{W}^{(r')}, \quad (4)$$

where $\mathbf{C}^{(1)} \in \mathbb{R}^{D \times D \times R_3 \times R_4}$ is the core tensor, and $\mathbf{U}^{(3,1)} \in \mathbb{R}^{M \times R_3}$, $\mathbf{U}^{(4,1)} \in \mathbb{R}^{N \times R_4}$ are the factor matrices corresponding to the input and output channel modes. The tuple (R_3, R_4) defines the multilinear ranks of the first-stage decomposition, and $r' = (R_3, R_4)$ denotes the corresponding rank pair.

4.2.2. Residual compensation stage

Conventional Tucker-2 truncation discards the residual tensor $\mathbf{E} = \mathbf{W} - \mathbf{W}^{(r')}$, which may contain non-negligible energy from the discarded singular subspaces. Inspired by the residual compensation mechanism (RCM) in ResSVD (Bai et al., 2025; Wang et al., 2024), we further decompose the residual tensor using a second-stage Tucker-2:

$$\mathbf{E} \approx \mathbf{C}^{(2)} \times_3 \mathbf{U}^{(3,2)} \times_4 \mathbf{U}^{(4,2)} = \mathbf{E}^{(r'')}, \quad (5)$$

where $\mathbf{C}^{(2)} \in \mathbb{R}^{D \times D \times S_3 \times S_4}$ is the second-stage core tensor, and $\mathbf{U}^{(3,2)} \in \mathbb{R}^{M \times S_3}$, $\mathbf{U}^{(4,2)} \in \mathbb{R}^{N \times S_4}$ are the second-stage factor matrices, with ranks $r'' = (S_3, S_4)$.

The final residual-compensated approximation of the convolutional kernel is then defined as

$$\widehat{\mathbf{W}}^{(r)} = \mathbf{W}^{(r')} + \mathbf{E}^{(r'')}, \quad r = (R_3 + S_3, R_4 + S_4). \quad (6)$$

The two-stage Tucker-2 process effectively recovers the residual energy discarded in the first truncation, resulting in a more accurate reconstruction of the original convolutional kernel.

Theorem 1 (Non-increasing Error Property of Residual Compensation). Let $\mathbf{W} \in \mathbb{R}^{D \times D \times M \times N}$ denote a convolutional kernel tensor. The residual $\mathbf{E}$ is defined as:

$$\mathbf{E} = \mathbf{W} - \mathbf{W}^{(r')}, \quad (7)$$

suppose the residual $\mathbf{E}$ is further approximated by a low-rank tensor $\mathbf{E}^{(r'')}$, and $\mathbf{W}^{(r')}$ and $\mathbf{E}^{(r')}$ are combined to form a two-stage approximation:

$$\widehat{\mathbf{W}}^{(r)} = \mathbf{W}^{(r')} + \mathbf{E}^{(r')}, \quad (8)$$

then, the Frobenius norm of the reconstruction error satisfies

$$\|\mathbf{W} - \widehat{\mathbf{W}}^{(r)}\|_F \leq \|\mathbf{W} - \mathbf{W}^{(r')}\|_F. \quad (9)$$

Proof 1. From the definition of the residual,

$$\mathbf{E} = \mathbf{W} - \mathbf{W}^{(r')}, \quad (10)$$

the overall approximation error after the second stage can be written as:

$$\mathbf{W} - \widehat{\mathbf{W}}^{(r)} = \mathbf{W} - (\mathbf{W}^{(r')} + \mathbf{E}^{(r'')}) \quad (11)$$

$$= (\mathbf{W} - \mathbf{W}^{(r')}) - \mathbf{E}^{(r'')} \quad (12)$$

$$= \mathbf{E} - \mathbf{E}^{(r'')}, \quad (13)$$

hence, the overall reconstruction error of $\widehat{\mathbf{W}}^{(r)}$ can be expressed in terms of the residual tensor $\mathbf{E}$ as follows:

$$\|\mathbf{W} - \widehat{\mathbf{W}}^{(r)}\|_F = \|\mathbf{E} - \mathbf{E}^{(r'')}\|_F, \quad (14)$$

suppose that the second-stage tensor $\mathbf{E}^{(r'')}$ is obtained as a best low-rank approximation of $\mathbf{E}$ over the admissible set $S_{r''} = \{\mathbf{A} : \text{multilinear rank}(\mathbf{A}) \leq r''\}$, i.e.

$$\mathbf{E}^{(r'')} = \arg \min_{\mathbf{A} \in S_{r''}} \|\mathbf{E} - \mathbf{A}\|_F, \quad (15)$$

since the zero tensor $\mathbf{0} \in S_{r''}$, the optimality of $\mathbf{E}^{(r'')}$ implies that its approximation error cannot exceed that of choosing the zero tensor, i.e.,

$$\|\mathbf{E} - \mathbf{E}^{(r'')}\|_F \leq \|\mathbf{E} - \mathbf{0}\|_F = \|\mathbf{E}\|_F, \quad (16)$$

substituting $\mathbf{E} = \mathbf{W} - \mathbf{W}^{(r')}$ yields

$$\|\mathbf{W} - \widehat{\mathbf{W}}^{(r)}\|_F \leq \|\mathbf{W} - \mathbf{W}^{(r')}\|_F, \quad (17)$$

therefore, the residual-compensation step cannot increase the Frobenius-norm reconstruction error. Moreover, the inequality is strict whenever the second-stage approximation $\mathbf{E}^{(r'')}$ captures a nonzero component of $\mathbf{E}$ (i.e., $\mathbf{E}^{(r'')} \neq \mathbf{0}$ and $\|\mathbf{E} - \mathbf{E}^{(r'')}\|_F < \|\mathbf{E}\|_F$).

Theorem 2 (Network-level Error Propagation Bound). Consider a compressed layer with weight tensor $\mathbf{W}$ and input $\mathbf{X}$. Let $\mathbf{W}^{(r')}$ and $\widehat{\mathbf{W}}^{(r)}$ denote the weights reconstructed by Tucker-2 and ResTucker-2, respectively. Assume that the subsequent network layers are Lipschitz continuous with constants $L_{l+1}, \dots, L_L$ and are identical for the two compressed networks. Then, the network-level error propagation bound induced by ResTucker-2 is no larger than that induced by Tucker-2:

$$\prod_{i=l+1}^L L_i \|\mathcal{F}(\cdot, \mathbf{X})\|_{\text{op}} \|\mathbf{W} - \widehat{\mathbf{W}}^{(r)}\|_F \leq \prod_{i=l+1}^L L_i \|\mathcal{F}(\cdot, \mathbf{X})\|_{\text{op}} \|\mathbf{W} - \mathbf{W}^{(r')}\|_F, \quad (18)$$

where $\|\mathcal{F}(\cdot, \mathbf{X})\|_{\text{op}}$ denotes the operator norm of the linear mapping with respect to the weight parameters for a fixed input $\mathbf{X}$. Moreover, the inequality is strict whenever $\|\mathbf{W} - \widehat{\mathbf{W}}^{(r)}\|_F < \|\mathbf{W} - \mathbf{W}^{(r')}\|_F$ and the propagation factors are nonzero.

Proof 2. According to Definition 1, $\mathcal{F}(\mathbf{W}, \mathbf{X})$ is linear with respect to $\mathbf{W}$ for a fixed input $\mathbf{X}$. Therefore, the perturbation introduced at the compressed layer by ResTucker-2 satisfies:

$$\|\mathcal{F}(\mathbf{W}, \mathbf{X}) - \mathcal{F}(\widehat{\mathbf{W}}^{(r)}, \mathbf{X})\|_F \leq \|\mathcal{F}(\cdot, \mathbf{X})\|_{\text{op}} \|\mathbf{W} - \widehat{\mathbf{W}}^{(r)}\|_F. \quad (19)$$

After propagation through the subsequent Lipschitz-continuous layers, the corresponding network-output perturbation is bounded by:

$$\prod_{i=l+1}^L L_i \|\mathcal{F}(\cdot, \mathbf{X})\|_{\text{op}} \|\mathbf{W} - \widehat{\mathbf{W}}^{(r)}\|_F. \quad (20)$$

Similarly, Tucker-2 gives the network-level bound:

$$\prod_{i=l+1}^L L_i \|\mathcal{F}(\cdot, \mathbf{X})\|_{\text{op}} \|\mathbf{W} - \mathbf{W}^{(r')}\|_F. \quad (21)$$

From Theorem 1,

$$\|\mathbf{W} - \widehat{\mathbf{W}}^{(r)}\|_F \leq \|\mathbf{W} - \mathbf{W}^{(r')}\|_F. \quad (22)$$

Since the two networks share the same nonnegative propagation factor:

$$\prod_{i=l+1}^L L_i \|\mathcal{F}(\cdot, \mathbf{X})\|_{\text{op}}, \quad (23)$$

Eq. (18) follows directly. When the inequality in Theorem 1 is strict and the propagation factor is positive, the corresponding network-level bound is also strictly smaller.

Remark 1. Theorem 2 extends the reconstruction guarantee in Theorem 1 from the parameter level to the network level. It shows that the smaller approximation error of ResTucker-2 leads to a tighter upper bound on the error propagated through subsequent network layers.

Remark 2. This theorem ensures that the residual compensation step never increases the reconstruction error, implying that a two-stage decomposition achieves at least the same or better approximation accuracy compared to performing only the first-stage factorization.

low-rank composition techniques can achieve substantial compression ratios; however, directly applying them to multiple layers of a neural network leads to accumulated approximation errors across layers. The presence of nonlinear transformations, such as ReLU and Batch Normalization (BN), further exacerbates the propagation and amplification of these errors. Moreover, an inappropriate choice of rank can significantly reduce the representational capacity of the model. Under high compression ratios, even with refined fine-tuning strategies, it is challenging to recover the lost accuracy fully. To address these issues, the next section proposes a Progressive knowledge distillation to mitigate the accuracy degradation caused by low-rank composition effectively.

4.3. Fine-grained tuning by dynamic temperature and progressive learning

Low-rank composition is a lossy compression scheme, which inevitably leads to some degree of performance degradation. However, knowledge distillation can effectively compensate for the information loss caused by low-rank approximation by guiding the student model to align with the feature distribution of the teacher model. Furthermore, this work designs a dynamic temperature and progressive distillation scheme to further enhance the model performance.

4.3.1. Dynamic temperature adjustment

Building on the standard knowledge distillation framework in Definition 3, prior works that combine low-rank composition with KD typically use a fixed distillation strategy across all layers. This static approach cannot adapt to the varying sensitivity of layers under high compression ratios, motivating our dynamic temperature mechanism to enable layer-adaptive knowledge transfer.

To flexibly adjust the smoothness of the teacher model's output, the temperature τ is treated as a tunable parameter θ_{temp} that dynamically evolves during training. This process can be viewed as a form of adaptive control, where the temperature is adjusted according to the learning state to improve knowledge transfer efficiency (He et al., 2024; Shao et al., 2025). Meanwhile, optimizing the temperature schedule can also be interpreted as a dynamic optimization problem aimed at enhancing the student model performance (Fang et al., 2024; Xue et al., 2024).

$$\min_{\mathbf{W}_{\text{stu}}, \theta_{\text{temp}}} \mathcal{L}(\mathbf{W}_{\text{stu}}, \theta_{\text{temp}}) = \min_{\mathbf{W}_{\text{stu}}, \theta_{\text{temp}}} \sum_{x \in \mathcal{D}} \mathcal{L}_{\text{task}}(f^s(x; \mathbf{W}_{\text{stu}}), y) + \alpha \mathcal{L}_{\text{kd}}(f^t(x; \mathbf{W}_{\text{tea}}), f^s(x; \mathbf{W}_{\text{stu}}), \theta_{\text{temp}}). \quad (24)$$

We adopt an alternating optimization algorithm to solve the above problem, decomposing it into two subproblems for gradual optimization of each part.

$$\begin{cases} \hat{\mathbf{W}}_{stu} = \arg \min_{\mathbf{W}_{stu}} \mathcal{L}(\mathbf{W}_{stu}, \hat{\theta}_{temp}), \\ \hat{\theta}_{temp} = \arg \min_{\theta_{temp}} \mathcal{L}(\hat{\mathbf{W}}_{stu}, \theta_{temp}). \end{cases} \quad (25)$$

The above two subproblems are solved using stochastic gradient descent. The update process for the student model parameters $\mathbf{W}_{stu}$ and the temperature parameter θ_{temp} is as follows:

$$\begin{cases} \mathbf{W}_{stu} \leftarrow \mathbf{W}_{stu} - \mu \frac{\partial \mathcal{L}}{\partial \mathbf{W}_{stu}}, \\ \theta_{temp} \leftarrow \theta_{temp} - \mu \frac{\partial \mathcal{L}}{\partial \theta_{temp}}, \end{cases} \quad (26)$$

where μ represents the learning rate.

4.3.2. Progressive learning

Note that training begins with low-difficulty tasks, allowing the student model to focus on mastering the basics despite its initially weak representation. As training progresses, task difficulty is gradually increased. This progression from easy to challenging tasks embodies the concept of progressive learning, enabling the student model to build its understanding incrementally and absorb knowledge more efficiently. Conversely, if high-difficulty tasks are introduced from the outset, the student model may struggle to follow guidance, which can adversely affect model convergence.

Inspired by progressive learning (Chan et al., 2024), this paper integrates the growth factor η into the loss function $\mathcal{L}$, forming an enhanced loss $\mathcal{L}'$ that organizes the distillation tasks progressively from easy to difficult. Consequently, the temperature parameter θ_{temp} is updated as follows:

$$\begin{cases} \mathbf{W}_{stu} \leftarrow \mathbf{W}_{stu} - \mu \frac{\partial \mathcal{L}'}{\partial \mathbf{W}_{stu}}, \\ \theta_{temp} \leftarrow \theta_{temp} - \mu \frac{\partial \mathcal{L}'}{\partial \theta_{temp}}. \end{cases} \quad (27)$$

During the initial phase of low-rank compression, the decomposed student model exhibits limited representational capacity, primarily capturing the teacher's core structural patterns. As the granularity of the decomposition adaptively expands (controlled by the rank parameter η), the model gradually enhances its ability to capture high-dimensional semantic correlations.

Four incremental functions are designed for η to facilitate progressive learning.

$$\eta_n = \begin{cases} \tilde{\eta} + (\hat{\eta} - \tilde{\eta}) \cdot \frac{1}{1 + e^{-\frac{n-T/2}{T/k}}}, \\ \tilde{\eta} + (\hat{\eta} - \tilde{\eta}) \cdot \max \left(0, \min \left(1, \frac{\log \left(\frac{n}{k+\epsilon} \right)}{\log \left(\frac{T}{k+\epsilon} \right)} \right) \right), \\ \tilde{\eta} + \frac{1}{2} (\hat{\eta} - \tilde{\eta}) \cdot \left(1 + \cos \left(\pi - \frac{n}{T} \pi \right) \right), \\ \tilde{\eta} + (\hat{\eta} - \tilde{\eta}) \cdot \frac{1}{1 + e^{-k \left(\frac{n}{T} - c \right)}}. \end{cases} \quad (28)$$

where $\tilde{\eta}$ and $\hat{\eta}$ represent the lower and upper bounds of η . T represents the total number of epochs. k stands for the scaling factor. ϵ is a very small positive number. Four incremental functions are designed to allow η to increase smoothly in the early stages and stabilize later, ensuring better coordination of the learning process.

Algorithm 1: PKD-LRC.

Input : Total epoch N , Loss balance parameter α ,
Training dataset $D = \{(x, y)\}$, Pre-train Teacher f_t ,
Pre-train Student f_s , Initial temperature θ_{temp}

Output: Compressed low-rank student model f_s^*

- 1 Initialize: Set progressive learning function η_n ;
- // low-rank composition of Student Model;
- 2 **for** each layer L_i in f_s **do**
- 3 **if** L_i is the convolutional layer **then**
- 4 **if** $i = 1$ **then**
- 5 Keep the same;
- 6 **end**
- 7 **else**
- 8 RCM via (4), (5) and (6);
- 9 **end**
- 10 **end**
- 11 **else**
- 12 Keep the same;
- 13 **end**
- 14 **end**
- 15 The student model acquires a low-rank model structure;
- // Progressive knowledge distillation;
- 16 **for** epoch = 1 to T **do**
- 17 Shuffle D into mini-batches;
- 18 **for** each mini-batch $(x, y) \in D$ **do**
- 19 $y_s \leftarrow f_s(x)$;
- 20 $y_t \leftarrow f_t(x)$;
- 21 $\eta_n \leftarrow$ Calculating via (28);
- 22 $\mathcal{L}_{task} \leftarrow \text{CrossEntropy}(y_s, y)$;
- 23 $\mathcal{L}_{kd} \leftarrow \text{KL-Divergence}(y_s, y_t)$;
- 24 $\mathcal{L} \leftarrow \mathcal{L}_{task} + \alpha \cdot \eta_n \cdot \mathcal{L}_{kd}$;
- 25 $\mathbf{W} \leftarrow$ Updating via (27);
- 26 $\theta_{temp} \leftarrow$ Updating via (27);
- 27 **end**
- 28 **end**

To ensure the non-negativity of the temperature parameter and constrain it within a reasonable range, this paper adopts the following formula for computation:

$$\theta = \theta_{init} + \text{range}(a(\theta_{pred})), \quad (29)$$

where θ_{init} represents the initial value of the temperature, range denotes the range of temperature, and a is the Sigmoid function. θ_{pred} is the predicted temperature value. In this paper, θ_{init} and range are set to 1 and 21, respectively, to ensure the temperature parameter changes within a reasonable range. Ultimately, PKD-LRC is summarized in Algorithm 1.

Remark 3. In the adaptive fine-tuning scheme, we propose a dynamic temperature distillation strategy combined with a progressive learning approach. In standard distillation, the temperature is typically treated as a fixed hyperparameter, which limits the ability to adjust the smoothness of the teacher's output based on data characteristics. In contrast, the dynamic temperature strategy effectively addresses this limitation. Moreover, the progressive learning approach allows the student model to focus on simpler knowledge in the early stages of training and gradually shift attention to more difficult knowledge in later stages, thereby enabling a low-rank student model to fully absorb the teacher's knowledge.

Table 1
Statistical results for different datasets.

Datasets	Class	Train	Test	Size
MNIST	10	6×10^4	1×10^4	28×28
CIFAR-10	10	5×10^4	1×10^4	32×32
CIFAR-100	100	5×10^4	1×10^4	32×32
SVHN	10	73,257	26,032	32×32
ImageNet	1000	1,281,167	5×10^4	224×224

5. Empirical studies

5.1. Datasets and baselines

We employ five datasets for image classification to comprehensively assess the performance of the compressed models. Detailed statistical information is summarized in Table 1.

1. **MNIST.** A simple handwritten digit recognition dataset with 10 categories. It contains 60,000 grayscale images for training and 10,000 for testing, making it a common benchmark for classification tasks in deep learning.
2. **SVHN.** A dataset of street view house numbers, also with 10 categories, featuring color images. It includes over 70,000 images for training and 26,032 for testing. With its complex backgrounds, SVHN is closer to real-world scenarios for digit recognition tasks.
3. **CIFAR-10/CIFAR-100.** A dataset of small-sized color images, where CIFAR-10 consists of 10 categories and CIFAR-100 consists of 100 categories. It includes 50,000 images for training and 10,000 for testing, making it suitable for various image classification tasks.
4. **ImageNet.** A large-scale dataset of real-world images, covering thousands of object categories. It contains over 1.2 million images for training and 50,000 for testing, and is widely used in deep learning research.

Baselines. To evaluate the effectiveness of the proposed PKD-LRC framework, we compare it with three categories of representative network compression methods:

1. **Low-rank compression methods:** LROS (Yang et al., 2020), ADMM-Tu2 (Yin et al., 2021), RR-Tu2 (Dai et al., 2025), RR (Dai et al., 2025), PSTR-S (Li et al., 2021), BATUDE (Yin et al., 2022), ELRT (Sui et al., 2024a), ALDS (Liebenwein et al., 2021), Stable EPC (Phan et al., 2020), MUSCO (Gusak et al., 2019), and HALOC (Xiao et al., 2023). These methods compress convolutional neural networks by exploiting the low-rank structure of convolutional kernels through various tensor decomposition techniques.
2. **Pruning-based methods:** EDP (Ruan et al., 2020), CHEX (Hou et al., 2022), EE (Zhang et al., 2021), SCOP (Tang et al., 2020), HOSA (Chatzikonstantinou et al., 2020), and DCP (Zhuang et al., 2018). These methods reduce model redundancy by removing unimportant channels, filters, or weights, thereby achieving network compression through structured pruning strategies.
3. **Low-rank + knowledge distillation methods:** TKD (Hajimolahoseini et al., 2022), PELA (Guo et al., 2024), TDKD (Dai et al., 2021), DTD (Wen et al., 2021), DTKD (Wei & Bai, 2024), and CTKD (Li et al., 2023b). These methods combine low-rank composition with knowledge distillation to improve the performance of compressed models, making them the most closely related approaches to our PKD-LRC framework.

For fair comparison, all methods are evaluated with the same backbone and teacher models. Specifically, MNIST uses LeNet-5 as the backbone (teacher: ResNet-20); SVHN uses ResNet-20 as the backbone (teacher: ResNet-32); CIFAR-10 and CIFAR-100 use VGG-16 (teacher: VGG-19), ResNet-20 (teacher: ResNet-32), and ResNet-32 (teacher: ResNet-50); ImageNet uses MobileNetV2 and ResNet-18 as backbones (teacher: ResNet-50).

5.2. Training details

All CNN models are trained using the PyTorch framework (1.13.1) and the Tensorly tool (0.8.1) for tensor operations. The optimizer used is stochastic gradient descent (SGD) with a momentum (0.9) and weight decay ($5e-4$) to improve convergence stability and prevent overfitting. A total of epochs (200) of fine-tuning are performed, with an initial learning rate (0.1), which is reduced by a factor of 0.1 at epochs 80, 120, and 160. The learning rate for the dynamic temperature parameter is kept constant at 0.01 throughout training to ensure stable distillation. To ensure fair comparison across all methods and datasets, the backbone and teacher architectures are matched consistently as described in Section 5.1, and the same optimizer, learning rate schedule, and batch size are applied to all models. The training is carried out on a system equipped with an Intel(R) Xeon(R) Gold 5218 CPU (2.30 GHz), 512 GB RAM, and an NVIDIA GeForce RTX 4090 24 GB GPU. These details, together with the explicit backbone-teacher pairings provided above each result table, ensure the experiments are fully reproducible.

5.3. Evaluation metrics

To better evaluate the compressed model, this paper uses Top-1 and Top-5 accuracy to measure model performance. Additionally, Compression Ratio (CR) is introduced to assess the reduction in parameters after compression. The Speedup Ratio (SR) is used to measure the reduction in FLOPs.

$$\begin{cases} CR = \frac{\text{Uncompressed Weights}}{\text{Compressed Weights}}, \\ SR = \frac{\text{Uncompressed FLOPs}}{\text{Compressed FLOPs}}, \end{cases} \quad (30)$$

a higher CR indicates better compression, meaning a significant reduction in model parameters, which typically leads to reduced storage and computational overhead. A higher SR value indicates that the compressed model is more efficient in terms of computational complexity.

5.4. Comparison results and analysis

We now present a detailed analysis of PKD-LRC's performance across different datasets.

1. **Results on MNIST (Table 2).** For LeNet-5 on MNIST, PKD-LRC demonstrates the effectiveness of our ResTucker-2 decomposition in mitigating low-rank approximation errors (Challenge 1). Compared to vanilla Tucker decomposition (Std.Tucker) and other optimized low-rank variants such as RR-Tu2, PKD-LRC achieves a higher Top-1 accuracy (+0.28%) while maintaining a competitive compression ratio, indicating that the residual compensation mechanism successfully preserves representational capacity. Knowledge distillation with dynamic temperature adjustment further supports accuracy retention, illustrating the importance of an adaptive accuracy-recovery strategy (Challenge 2) even in relatively small networks.
2. **Results on SVHN (Table 3).** For ResNet-20 on SVHN, vanilla low-rank methods (Std.SVD, Std.Tucker) suffer noticeable accuracy drops due to approximation errors, while hybrid methods that combine low-rank composition with optimization techniques partially recover performance. PKD-LRC surpasses these baselines, achieving a Top-1 accuracy improvement of 0.49% over the uncompressed model under high compression. This outcome highlights the synergistic effect of ResTucker-2 for explicit error compensation and the progressive learning strategy combined with dynamic temperature adjustment to adaptively recover accuracy across layers, effectively addressing both challenges.
3. **Results on CIFAR-10 and CIFAR-100 (Table 4).** PKD-LRC demonstrates consistent improvements across multiple student-teacher

Table 2

Results on MNIST. LeNet-5 as student and ResNet-20 as the teacher.

Methods	Comp. Meth.	Top-1 (Δ)	CR	SR
Uncomp.	–	98.75 (+0.00)	1	1
Std.SVD	SVD	98.50 (−0.25)	1.53	1.51
Std.Tucker	Tucker	98.33 (−0.42)	2.14	2.09
RR (Dai et al., 2025)	Tucker	98.83 (+0.08)	6.19	2.04
RR-Tu2 (Dai et al., 2025)	Tucker	98.83 (+0.08)	6.19	5.95
ELRT (Sui et al., 2024a)	Tucker	98.80 (+0.05)	4.12	3.97
TDKD (Dai et al., 2021)	Tucker + KD	98.91 (+0.16)	4.12	3.97
PELA (Guo et al., 2024)	TSVD + KD	98.76 (+0.01)	4.12	3.97
TKD (Hajimolahoseini et al., 2022)	Tucker-2 + KD	98.89 (+0.14)	4.35	4.07
PKD-LRC(Ours)	ResTucker-2 + KD	99.03 (+0.28)	6.20	4.83

Table 3

Results on SVHN. ResNet-20 as the student and ResNet-32 as the teacher.

Methods	Comp. Meth.	Top-1 (Δ)	CR	SR
Uncomp.	–	95.77 (+0.00)	1	1
Std.SVD	SVD	87.78 (−7.99)	1.99	2.00
Std.Tucker	Tucker	88.57 (−7.20)	2.00	1.91
LROS (Yang et al., 2020)	TSVD + sparsity	94.39 (−1.38)	1.99	2.00
EDP (Ruan et al., 2020)	Tucker + sparsity	95.77 (+0.00)	2.11	2.03
ELRT (Sui et al., 2024a)	Tucker	95.80 (+0.03)	2.01	1.93
TDKD (Dai et al., 2021)	Tucker + KD	95.85 (+0.08)	2.01	1.93
PELA (Guo et al., 2024)	TSVD + KD	94.72 (−1.05)	2.01	1.93
TKD (Hajimolahoseini et al., 2022)	Tucker + KD	95.99 (+0.22)	2.01	1.93
PKD-LRC(Ours)	ResTucker-2 + KD	96.26 (+0.49)	4.44	4.04

pairs (ResNet-20/32, VGG-16) on CIFAR-10/100. On CIFAR-10, Top-1 accuracy reaches 91.73%, 92.37%, and 91.55% for ResNet-20, ResNet-32, and VGG-16, respectively, while achieving substantial compression ratios. On CIFAR-100, Top-1 accuracy gains of 1.35%, 1.62%, and 0.35% are observed. These results confirm that PKD-LRC’s error compensation mechanism (ResTucker-2) effectively mitigates representation loss, and its dynamic temperature with progressive learning strategy ensures that high compression does not lead to significant accuracy degradation. In contrast, vanilla low-rank methods often fail to recover accuracy under similar compression, and optimized low-rank variants may require lower compression ratios to maintain performance, demonstrating PKD-LRC’s ability to balance compression and accuracy across different architectures.

4. *Results on ImageNet (Table 5)*. For large-scale datasets, PKD-LRC continues to provide robust performance for both ResNet-18 and MobileNetV2. Using ResNet-18, Top-1 accuracy is slightly improved (+0.20%) under a 2.78× compression ratio, while maintaining Top-5 performance. For MobileNetV2, PKD-LRC achieves 72.07% Top-1 accuracy, surpassing other pruning- and low-rank-based methods. These results confirm that the residual compensation mechanism effectively reduces approximation errors (Challenge 1) in large networks, and the progressive learning with dynamic temperature reliably restores accuracy under aggressive compression (Challenge 2), validating the generalizability of our approach to complex architectures and datasets.

Overall, across all five datasets and multiple network architectures, PKD-LRC consistently addresses the two key challenges identified in this study: ResTucker-2 compensates for low-rank approximation errors, while dynamic temperature adjustment and progressive learning enable effective accuracy recovery. The consistent improvements across different datasets, architectures, and compression settings indicate that these advantages are not restricted to a specific experimental configuration. This alignment between the identified challenges, proposed mechanisms, and experimental results demonstrates the effectiveness and generalizability of PKD-LRC.

5.5. Comparison with other additive temperature methods

Table 6 reports the comparative results of different adaptive temperature methods. Since existing dynamic temperature adjustment strategies may not simultaneously account for both low-rank composition and knowledge distillation, this paper applies several advanced adaptive temperature adjustment strategies to models that integrate both techniques to ensure a fair comparison. Experimental results demonstrate that our approach not only enhances the performance of the baseline model but also outperforms current mainstream methods across various evaluations. Its advantage primarily stems from the dynamic temperature control and progressive learning mechanism, which gradually restores the model representational capacity while effectively suppressing error propagation in nonlinear layers.

5.6. Impact of temperature ranges on model performance

Fig. 2 shows the impact of temperature ranges and growth functions on model accuracy. For temperature, optimal values vary across architectures and datasets: while ResNet-20 and ResNet-32 on CIFAR-100 prefer lower temperatures, most models benefit from higher ones. This reflects the interaction of model capacity, data complexity, and low-rank compression, as decomposition effectively deepens the network and makes conventional settings less suitable; expanding the search space thus improves convergence to a global optimum. For growth functions, performance depends on their mathematical properties and task complexity. Sigmoid, Cosine, and Exponent perform best at low, medium, and high temperatures, respectively, whereas Logarithm adapts poorly due to slow growth. Overall, Sigmoid suits small datasets and shallow networks, Exponent favors larger datasets, and Cosine shows strong generalizability, offering a good balance between accuracy and compression.

5.7. Dynamic temperature learning analysis

In the analysis of dynamic temperature learning curves (Fig. 3), different progressive learning strategies exhibit distinct characteristics: the

Table 4

Comparison of different neural network architectures on CIFAR-10/CIFAR-100. The experiments evaluate knowledge distillation effectiveness using various teacher-student architectures. Specifically, ResNet-20 is training with ResNet-32 as the teacher, ResNet-32 with ResNet-50, and VGG-16 with VGG-19.

Basic Models	Methods	Comp. Meth.	Top-1 (Δ) (%)	Top-5 (Δ) (%)	CR	SR
CIFAR-10						
ResNet-20	Uncomp.	–	91.25	99.72	1	1
	Std.Tucker	Tucker	87.48 (−3.77)	99.36 (−0.36)	2.49	2.34
	PSTR-S (Li et al., 2021)	TR	90.80 (−0.45)	99.74 (+0.02)	2.50	2.34
	BATUDE (Yin et al., 2022)	Tucker	91.02 (−0.23)	99.73 (+0.01)	2.60	2.36
	ELRT (Sui et al., 2024a)	Tucker	91.27 (+0.02)	99.74 (+0.02)	2.02	1.93
	LROS (Yang et al., 2020)	TSVD + Sparsity	90.39 (−0.86)	99.57 (−0.15)	2.94	2.21
	EDP (Ruan et al., 2020)	TSVD + Sparsity	91.03 (−0.22)	99.73 (+0.01)	2.11	2.11
	PELA (Guo et al., 2024)	TSVD + KD	90.40 (−0.85)	99.65 (−0.07)	2.02	1.93
	TDKD (Dai et al., 2021)	Tucker + KD	91.66 (+0.41)	99.74 (+0.02)	2.02	1.93
	TKD (Hajimolahoseini et al., 2022)	Tucker + KD	91.54 (+0.29)	99.75 (+0.03)	2.02	1.93
	PKD-LRC(Ours)	ResTucker-2 + KD	91.73 (+0.48)	99.76 (+0.02)	3.05	2.63
ResNet-32	Uncomp.	–	92.49	99.74	1	1
	Std. Tucker	Tucker	87.37 (−5.12)	99.38 (−0.36)	5.55	5.33
	PSTR-S (Li et al., 2021)	TR	91.44 (−1.05)	99.71 (−0.03)	5.80	5.17
	BATUDE (Yin et al., 2022)	Tucker	92.18 (−0.31)	99.77 (+0.03)	2.80	2.61
	ELRT (Sui et al., 2024a)	Tucker	92.25 (−0.24)	99.77 (+0.03)	2.02	1.93
	LROS (Yang et al., 2020)	TSVD + Sparsity	91.22 (−1.27)	99.71 (−0.03)	3.15	3.06
	EDP (Ruan et al., 2020)	TSVD + Sparsity	91.31 (−1.18)	99.76 (+0.02)	2.27	2.15
	PELA (Guo et al., 2024)	TSVD + KD	91.23 (−1.26)	99.75 (+0.01)	2.02	1.93
	TDKD (Dai et al., 2021)	Tucker + KD	92.29 (−0.20)	99.73 (−0.01)	2.02	1.93
	TKD (Hajimolahoseini et al., 2022)	Tucker + KD	92.21 (−0.28)	99.71 (−0.03)	2.02	1.93
	PKD-LRC(Ours)	ResTucker-2 + KD	92.37 (−0.12)	99.77 (+0.03)	5.97	5.22
VGG-16	Uncomp.	–	91.65	99.75	1	1
	Std. Tucker	Tucker	82.72 (−8.93)	99.17 (−0.58)	3.25	3.04
	ADMM-Tu2 (Yin et al., 2021)	TT	91.17 (−0.48)	99.72 (−0.03)	4.67	5.17
	PELA (Guo et al., 2024)	TSVD + KD	90.85 (−0.80)	99.65 (−0.10)	5.53	5.20
	TKD (Hajimolahoseini et al., 2022)	Tucker + KD	91.33 (−0.32)	99.70 (−0.05)	5.53	5.20
	TDKD (Dai et al., 2021)	Tucker + KD	91.37 (−0.28)	99.70 (−0.05)	5.53	5.20
	PKD-LRC(Ours)	ResTucker-2 + KD	91.55 (−0.10)	99.74 (−0.01)	5.53	5.20
CIFAR-100						
ResNet-20	Uncomp.	–	65.40	89.74	1	1
	PSTR-S (Li et al., 2021)	TR	66.13 (+0.73)	89.81 (+0.07)	2.30	2.14
	BATUDE (Yin et al., 2022)	Tucker	66.67 (+1.27)	90.18 (+0.44)	2.80	2.00
	ELRT (Sui et al., 2024a)	Tucker	65.69 (+0.29)	89.89 (+0.15)	1.97	1.90
	LROS (Yang et al., 2020)	TSVD + Sparsity	64.36 (−1.04)	85.33 (−4.41)	2.15	2.10
	EDP (Ruan et al., 2020)	TSVD + Sparsity	64.75 (−0.65)	88.84 (−0.90)	2.04	1.97
	PELA (Guo et al., 2024)	TSVD + KD	57.34 (−8.06)	84.77 (−4.97)	2.02	1.93
	TDKD (Dai et al., 2021)	Tucker + KD	66.27 (+0.87)	90.86 (+1.12)	2.02	1.93
	TKD (Hajimolahoseini et al., 2022)	Tucker + KD	65.97 (+0.57)	90.81 (+1.07)	2.02	1.93
	PKD-LRC(Ours)	ResTucker-2 + KD	66.75 (+1.35)	90.89 (+1.15)	3.27	2.96
ResNet-32	Uncomp.	–	68.10	90.31	1.00	1.00
	Std. Tucker	Tucker	59.03 (−9.07)	85.07 (−5.24)	2.50	2.34
	PSTR-S (Li et al., 2021)	TR	68.05 (−0.05)	90.56 (+0.25)	2.40	2.23
	EDP (Ruan et al., 2020)	TSVD	67.36 (−0.74)	90.25 (−0.06)	2.05	2.14
	BATUDE (Yin et al., 2022)	Tucker	68.95 (+0.85)	90.76 (+0.45)	2.60	2.38
	ELRT (Sui et al., 2024a)	Tucker	68.26 (+0.16)	90.75 (+0.44)	2.02	1.93
	LROS (Yang et al., 2020)	TSVD + Sparsity	67.28 (−0.82)	90.32 (+0.01)	2.13	2.02
	PELA (Guo et al., 2024)	TSVD + KD	62.61 (−5.49)	86.44 (−3.87)	2.02	1.93
	TDKD (Dai et al., 2021)	Tucker + KD	68.46 (+0.36)	91.83 (+1.52)	2.02	1.93
	TKD (Hajimolahoseini et al., 2022)	Tucker + KD	68.35 (+0.25)	91.68 (+1.37)	2.02	1.93
	PKD-LRC(Ours)	ResTucker-2 + KD	69.72 (+1.62)	91.90 (+1.59)	3.13	3.01
VGG-16	Uncomp.	–	68.40	89.89	1	1
	Std. Tucker	Tucker	62.85 (−5.55)	87.62 (−2.27)	3.15	3.04
	ADMM-Tu2 (Yin et al., 2021)	TT	66.39 (+2.01)	89.64 (−0.25)	4.67	5.17
	RR (Dai et al., 2025)	Tucker	67.00 (−1.40)	89.58 (−0.31)	7.67	3.19
	RR-Tu2 (Dai et al., 2025)	Tucker	67.83 (−0.57)	89.61 (−0.28)	6.46	6.27
	PELA (Guo et al., 2024)	TSVD + KD	61.93 (−6.47)	86.91 (−2.98)	1.99	2.00
	TDKD (Dai et al., 2021)	Tucker + KD	68.55 (+0.15)	90.23 (+0.34)	5.68	5.12
	TKD (Hajimolahoseini et al., 2022)	Tucker + KD	68.26 (−0.14)	89.86 (−0.04)	5.68	5.12
	PKD-LRC(Ours)	ResTucker-2 + KD	68.75 (+0.35)	90.34 (+0.45)	7.75	7.11

Table 5
Results on ImageNet. ResNetT-18/MobileNetV2 as the student and ResNet-50 as teacher.

Methods	Comp. Meth.	Top-1(%)	Top-5(%)	CR
ResNet-18				
Uncomp.	–	69.75	89.08	1
ALDS (Liebenwein et al., 2021)	Matrix	69.64	89.03	2.75
Stable EPC (Phan et al., 2020)	CP	69.17	89.08	2.75
MUSCO (Gusak et al., 2019)	Tucker	69.29	88.78	2.54
CHEX (Hou et al., 2022)	Sparsity	69.60	89.01	1.85
EE (Zhang et al., 2021)	Sparsity	68.27	88.44	1.95
SCOP (Tang et al., 2020)	Sparsity	69.18	88.89	1.65
PELA (Guo et al., 2024)	TSVD + KD	65.20	86.93	1.94
TDKD (Dai et al., 2021)	Tucker + KD	69.76	89.08	1.94
TKD (Hajimolahoseini et al., 2022)	Tucker + KD	69.85	89.10	1.94
PKD-LRC(Ours)	ResTucker-2 + KD	69.95	89.15	2.78
MobileNetV2				
Uncomp.	–	71.85	90.33	1
ALDS (Liebenwein et al., 2021)	Matrix	70.32	89.60	1.49
HOSA (Chatzikonstantinou et al., 2020)	Pruning	64.43	86.85	1.37
DCP (Zhuang et al., 2018)	Pruning	64.22	86.81	1.35
HALOC (Xiao et al., 2023)	Tucker-2	70.98	89.77	1.67
PELA (Guo et al., 2024)	TSVD + KD	66.09	87.63	1.94
TDKD (Dai et al., 2021)	Tucker + KD	71.85	90.35	1.94
TKD (Hajimolahoseini et al., 2022)	Tucker + KD	71.93	90.36	1.94
PKD-LRC(Ours)	ResTucker-2 + KD	72.07	90.40	1.94

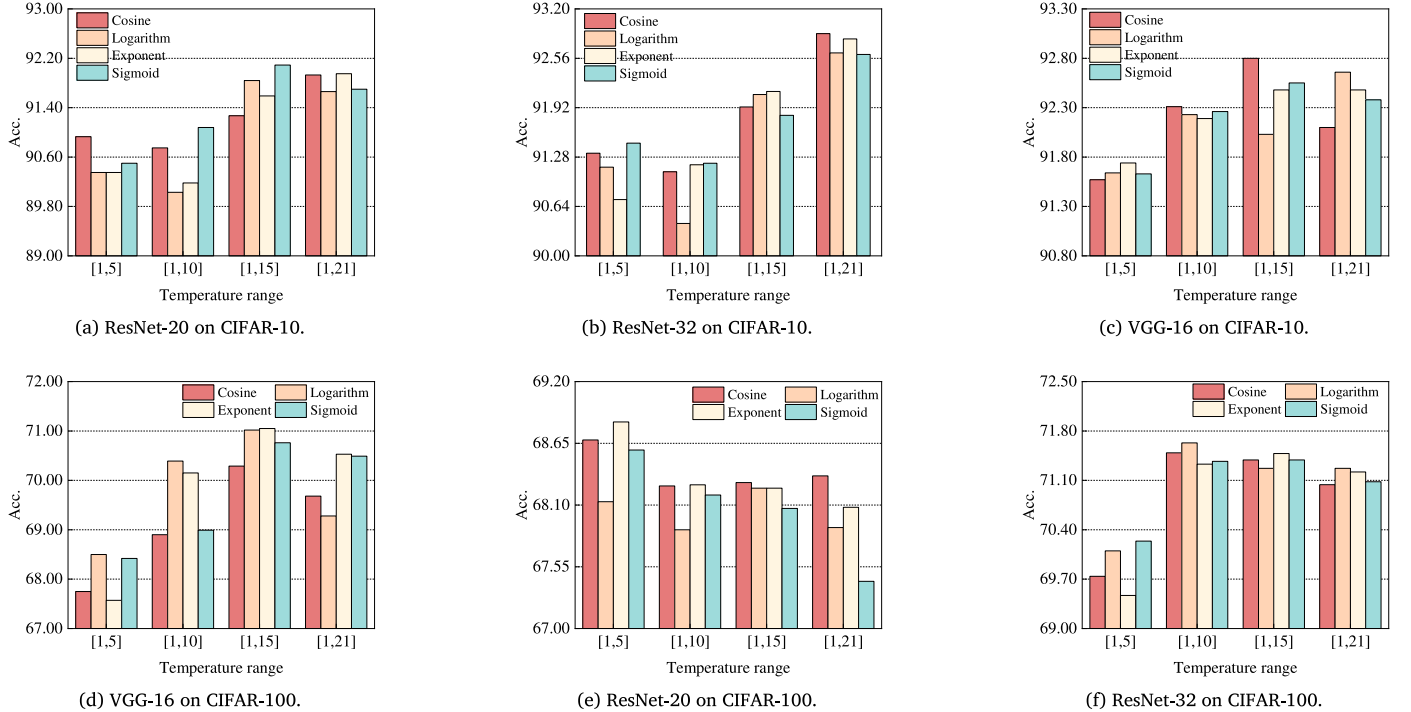


Fig. 2. The impact of different dimension search ranges and growth functions on accuracy. All models are controlled with a compression ratio of 2.

Table 6
Comparison with exiting additive compression methods on CIFAR-10.

Methods	Comp. Meth.	Top-1 (Δ)	CR	SR
Uncomp.	–	91.25	1	1
DTD (Wen et al., 2021)	Tucker-2 + KD	86.88 (−4.37)	2.02	1.93
DTKD (Wei & Bai, 2024)	Tucker-2 + KD	91.37 (+0.12)	2.02	1.93
CTKD (Li et al., 2023b)	Tucker-2 + KD	91.85 (+0.60)	2.02	1.93
PKD-LRC(Ours)	Tucker-2 + KD	92.05 (+0.80)	2.02	1.93
	Tucker-2 + KD	91.73 (+0.48)	3.05	2.63

logarithmic strategy rises quickly, enabling rapid label softening, while the sigmoid strategy increases more gradually, supporting smoother and more stable knowledge transfer; the other two strategies fall in between,

highlighting the influence of mathematical formulations on temperature adaptation. Furthermore, when comparing CIFAR-10 and CIFAR-100, the latter shows a smaller fluctuation range and a lower convergence value, suggesting that in more complex classification tasks, dynamic temperature strategies require finer adjustments to avoid excessive softening or hardening, thereby enhancing adaptability across different datasets.

5.8. Qualitative visualization

1. *t-SNE Visualization Analysis.* To further validate the effectiveness of our approach, we utilize the t-SNE tool to visualize feature distributions, as shown in Fig. 4. The comparison reveals that samples

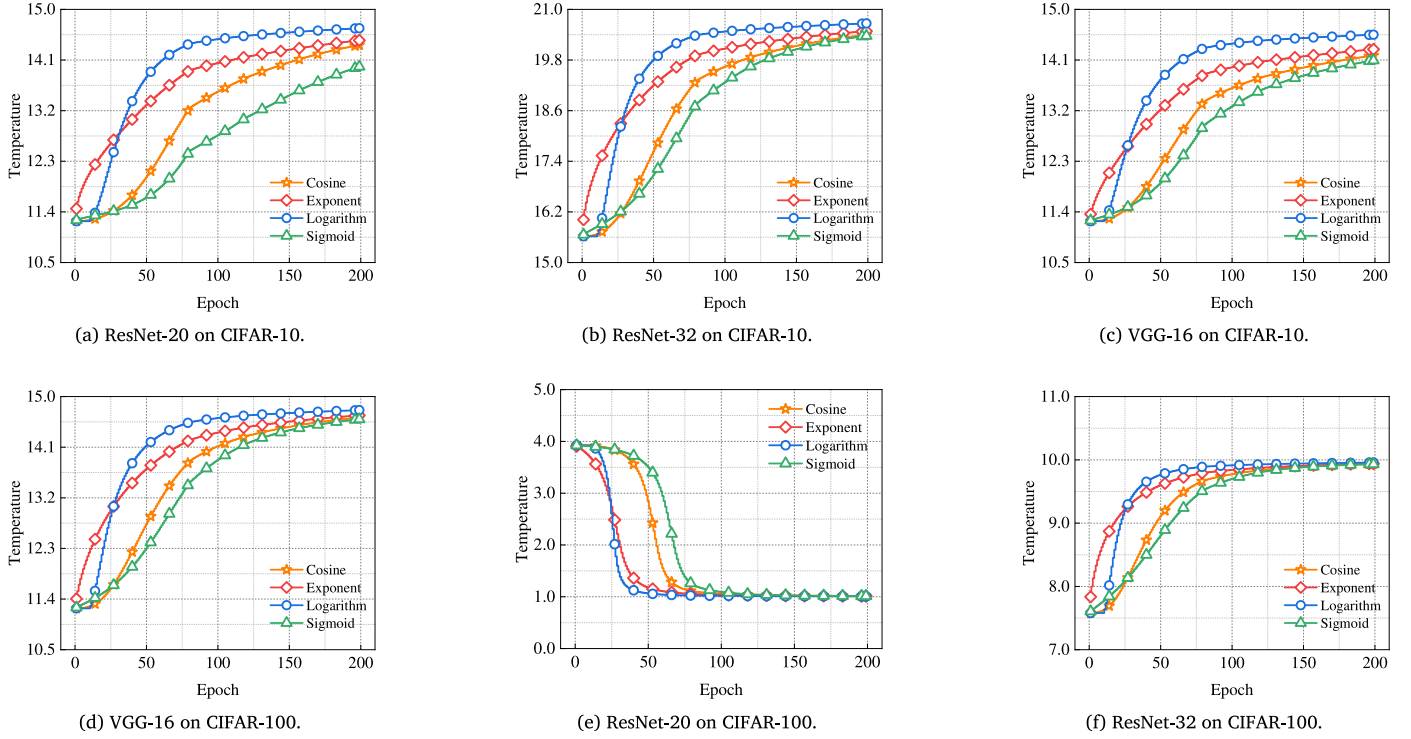
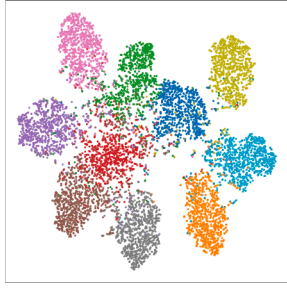


Fig. 3. Temperature learning curves for different datasets and models.

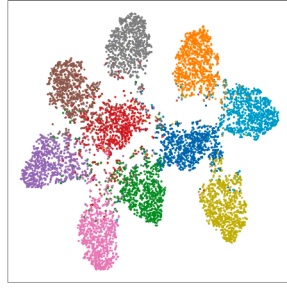
of the same class exhibit higher compactness, while different classes are more distinctly separated, leading to improved classification performance. Moreover, for the same model applied to other datasets, Fig. 4h and f show similar clustering quality, indicating that our method is robust for high-category datasets. Additionally, for different models on the same dataset, Fig. 4b, d, and 4 exhibit clearer feature separation compared to the original model, suggesting that our method generalizes well across deep models of varying architectures.

2. *Correlation Analysis.* Fig. 5 presents the correlation matrix of different deep models on the CIFAR-100 dataset. It can be observed

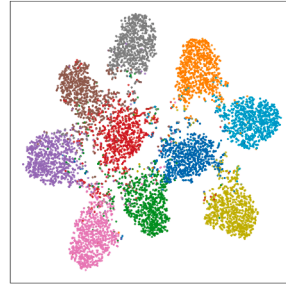
that the diagonal elements of the feature matrix obtained by our method are more pronounced, while the values of the off-diagonal elements are significantly reduced, with noticeably lighter colors. This result demonstrates that the combination of ResTucker-2 decomposition, dynamic temperature, and progressive learning strategy effectively reduces redundancy in features while improving both feature independence and correlation, further validating the effectiveness of our proposed method.



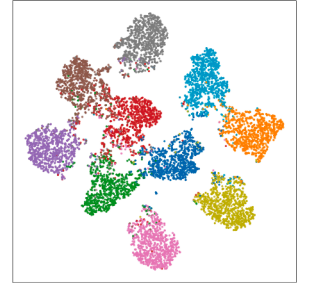
(a) ResNet-20 on CIFAR-10 (Original).



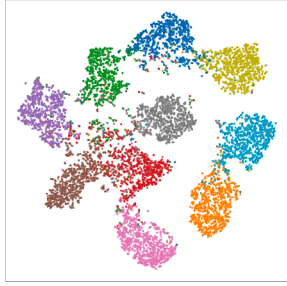
(b) ResNet-20 on CIFAR-10 (Ours).



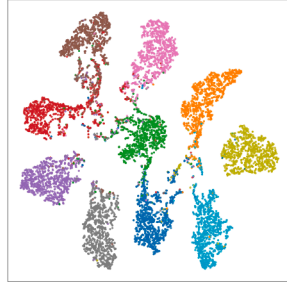
(c) ResNet-32 on CIFAR-10 (Original).



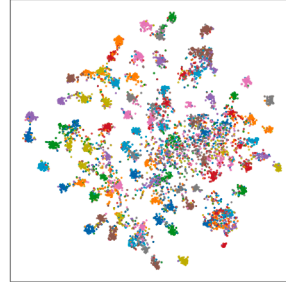
(d) ResNet-32 on CIFAR-10 (Ours).



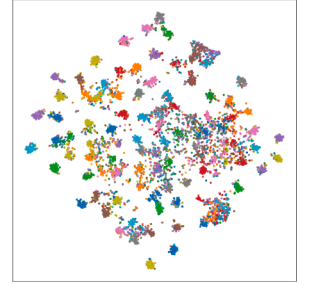
(e) VGG-16 on CIFAR-10 (Original).



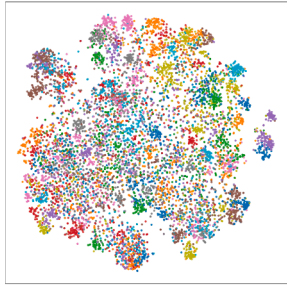
(f) VGG-16 on CIFAR-10 (Ours).



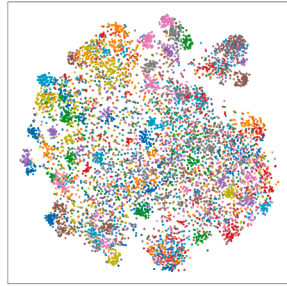
(g) VGG-16 on CIFAR-100 (Original).



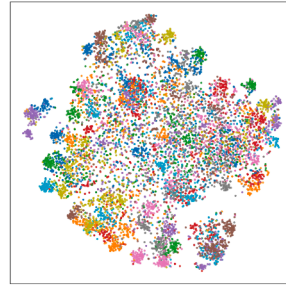
(h) VGG-16 on CIFAR-100 (Ours).



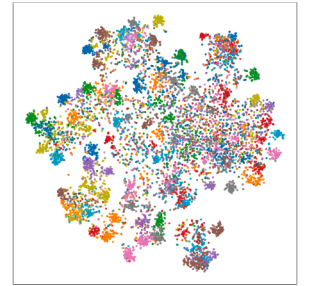
(i) ResNet-20 on CIFAR-100 (Original).



(j) ResNet-20 on CIFAR-100 (Ours).



(k) ResNet-32 on CIFAR-100 (Original).



(l) ResNet-32 on CIFAR-100 (Ours).

Fig. 4. t-SNE of features learned by original and our method.

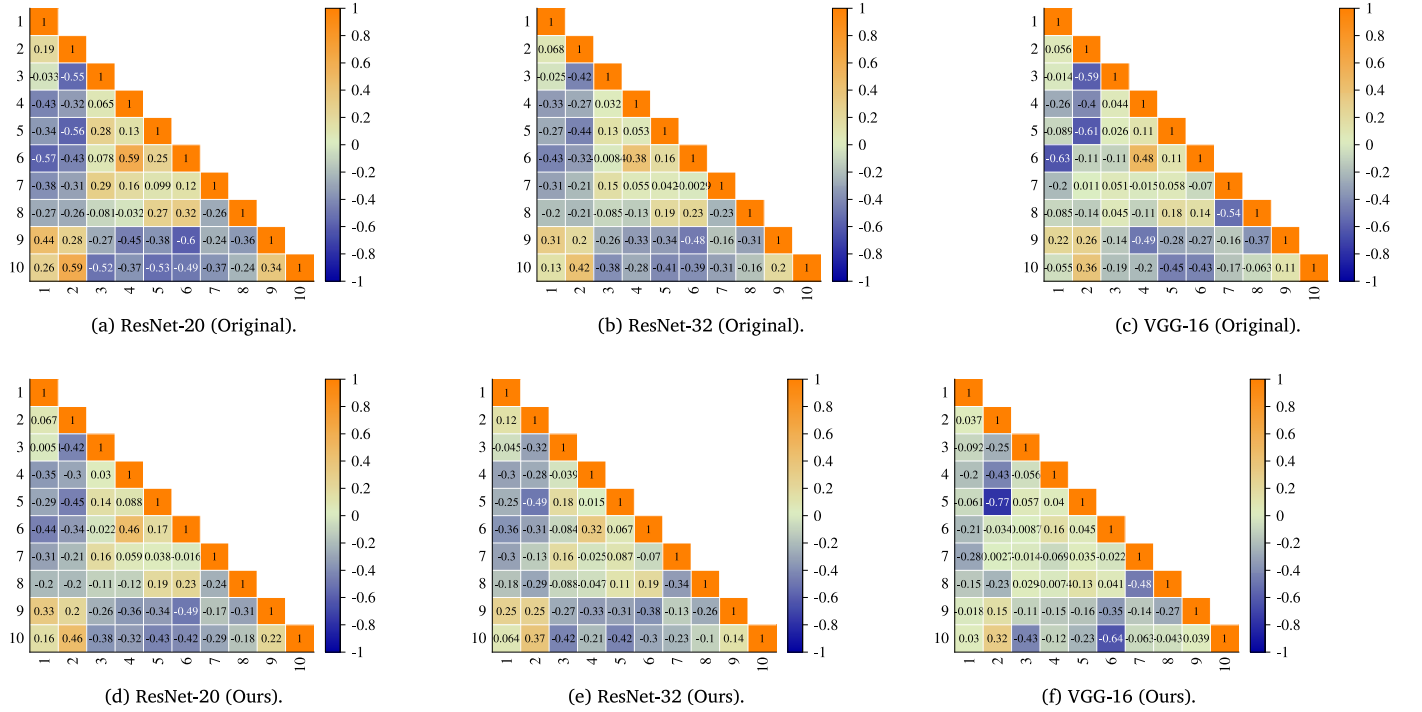


Fig. 5. Correlation matrix of various models on the CIFAR-10 dataset.

6. Conclusions

In this paper, we have proposed a progressive knowledge distillation method with low-rank composition for model compression. The ResTucker-2 decomposition has been employed to reduce performance degradation in low-rank compressed models, while a dynamic temperature strategy and progressive learning rules have enhanced the adaptability and performance of the student model. Experiments on benchmark datasets have demonstrated that PKD-LRC has outperformed both pure low-rank and knowledge distillation methods, providing an effective solution for resource-constrained environments. Future work will explore extending PKD-LRC to Transformer-based architectures, as well as integrating it with quantization-aware training and hardware-aware neural architecture search for broader deployment.

CRedit authorship contribution statement

Yaping He: Writing – original draft, Visualization, Software, Methodology, Investigation, Formal analysis; **Hao Wu:** Writing – review & editing, Validation, Methodology, Formal analysis, Conceptualization; **Weibo Liu:** Writing – review & editing, Validation, Methodology, Formal analysis, Conceptualization; **Xin Luo:** Writing – review & editing, Supervision, Resources, Methodology, Conceptualization.

Declaration of competing interest

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

- An, S., Ma, Z., Lin, Z., Zheng, N., Lou, J. G., & Chen, W. (2024). Make your LLM fully utilize the context. In *Adv. Neural Inf. Process. Syst.* (pp. 62160–62188). (vol. 37).
- An, Y., Zhao, X., Yu, T., Tang, M., & Wang, J. (2024b). Fluctuation-based adaptive structured pruning for large language models. In *Proc. AAAI Conf. Artif. Intell.* (pp. 10865–10873).

- Bai, H., Jian, S., Liang, T., Yin, Y., & Wang, H. (2025). ResSVD: Residual compensated SVD for large language model compression, arXiv preprint arXiv:2505.20112.
- Chan, S., Meng, W., Bai, C., Hu, J., & Chen, S. (2024). Diverse-feature collaborative progressive learning for visible-infrared person re-identification. *IEEE Transactions on Industrial Informatics*, 20, 7754–7763.
- Chatzikonstantinou, C., Papadopoulos, G. T., Dimitropoulos, K., & Daras, P. (2020). Neural network compression using higher-order statistics and auxiliary reconstruction losses. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops* (pp. 716–717).
- Chen, L., Wu, P., Tan, W., Li, H., Chen, H., & Zeng, N. (2025). A novel UAV-based road damage detection algorithm with lightweight convolution and attention mechanism. *International Journal of Network Dynamics and Intelligence*, 4(4), 100025.
- Chen, Z., Zhang, L., Tang, J., Mao, J., & Sheng, W. (2024). Conditional generative adversarial net based feature extraction along with scalable weakly supervised clustering for facial expression classification. *International Journal of Network Dynamics and Intelligence*, 3(4), 100024.
- Cheng, H., Zhang, M., & Shi, J. Q. (2024). A survey on deep neural network pruning: Taxonomy, comparison, analysis, and recommendations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46, 10558–10578.
- Dai, C., Liu, X., Li, Z., & Chen, M. Y. (2021). A Tucker decomposition based knowledge distillation for intelligent edge applications. *Applied Soft Computing*, 101, 107051.
- Dai, W., Fan, J., Miao, Y., & Hwang, K. (2025). Deep learning model compression with rank reduction in tensor decomposition. *IEEE Transactions on Neural Networks and Learning Systems*, 36, 1315–1328.
- Fang, W., Shen, B., Pan, A., Zou, L., & Song, B. (2024). A cooperative stochastic configuration network based on differential evolutionary sparrow search algorithm for prediction. *Systems Science & Control Engineering*, 12(1), 2314481.
- Gao, S., Li, J., Zhang, Z., Zhang, Y., Cai, W., & Huang, H. (2024). Device-wise federated network pruning. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.* (pp. 12342–12352).
- Guo, Y., Wang, G., & Kankanhalli, M. (2024). PELA: Learning parameter-efficient models with low-rank approximation. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.* (pp. 15699–15709).
- Gusak, J., Kholiavchenko, M., Ponomarev, E., Markeeva, L., Blagoveschensky, P., Cichocki, A., & Oseledets, I. (2019). Automated multi-stage compression of neural networks. In *Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshops* (pp. 1–8).
- Hajimolahoseini, H., Ahmed, W., Rezagholizadeh, M., Partovinia, V., & Liu, Y. (2022). Strategies for applying low rank decomposition to transformer-based models. In *Conf. Neural Inf. Process. Syst.* (pp. 1–6).
- He, S., Karimi, H. R., Li, Y., & Wang, Y. (2024). Memory synchronisation of delayed neural networks via variable sampling control. *International Journal of Systems Science*, 55(16), 3412–3424.
- Hinton, G., Vinyals, O., & Dean, J. (2015). Distilling the knowledge in a neural network, arXiv preprint arXiv:1503.02531.
- Hou, Z., Qin, M., Sun, F., Ma, X., Yuan, K., Xu, Y., Chen, Y. K., Jin, R., Xie, Y., & Kung, S. Y. (2022). CHEX: Channel exploration for CNN model compression. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.* (pp. 12287–12298).

- Hu, L., Liu, Z., Zhao, Z., Hou, L., Nie, L., & Li, J. (2024). A survey of knowledge enhanced pre-trained language models. *IEEE Transactions on Knowledge and Data Engineering*, 36, 1413–1430.
- Li, N., Pan, Y., Chen, Y., Ding, Z., Zhao, D., & Xu, Z. (2021). Heuristic rank selection with progressively searching tensor ring network. *Complex & Intelligent Systems*, 8, 771–785.
- Li, Z., Li, X., Yang, L., Zhao, B., Song, R., Luo, L., Li, J., & Yang, J. (2023b). Curriculum temperature for knowledge distillation. In *Proc. AAAI Conf. Artif. Intell.* (pp. 1504–1512).
- Liebenwein, L., Maalouf, A., Feldman, D., & Rus, D. (2021). Compressing neural networks: Towards determining the optimal layer-wise decomposition. *Advances in Neural Information Processing Systems*, 34, 5328–5344.
- Phan, A. H., Sobolev, K., Sozykin, K., Ermilov, D., Gusak, J., Tichavský, P., Glukhov, V., Oseledets, I., & Cichocki, A. (2020). Stable low-rank tensor decomposition for compression of convolutional neural network. In *Proc. Eur. Conf. Comput. Vis.* (pp. 522–539). Springer.
- Ruan, X., Liu, Y., Yuan, C., Li, B., Hu, W., Li, Y., & Maybank, S. (2020). EDP: An efficient decomposition and pruning scheme for convolutional neural network compression. *IEEE Transactions on Neural Networks and Learning Systems*, 32, 4499–4513.
- Senderovich, A., Bulatova, E., Obukhov, A., & Rakhuba, M. (2022). Towards practical control of singular values of convolutional layers. *Advances in Neural Information Processing Systems*, 35, 10918–10930.
- Sepahvand, M., Abdali-Mohammadi, F., & Taherkordi, A. (2022). Teacher–student knowledge distillation based on decomposed deep feature representation for intelligent mobile applications. *Expert Systems with Applications*, 202, 117474.
- Shao, X., Zhang, J., Lyu, M., & Lu, Y. (2025). Event-based nonfragile state estimation for memristive neural networks with multiple time-delays and sensor saturations. *International Journal of Systems Science*, 56(3), 618–637.
- Shuvo, M. M. H., Islam, S. K., Cheng, J., & Morshed, B. I. (2022). Efficient acceleration of deep learning inference on resource-constrained edge devices: A review. *Proceedings of the IEEE*, 111, 42–91.
- Song, B., Zhao, S., Dang, L., Wang, H., & Xu, L. (2025). A survey on learning from data with label noise via deep neural networks. *Systems Science and Control Engineering*, 13(1), 2488120.
- Su, Q., He, W., Wei, X., Xu, B., & Li, G. (2024). Multi-scale full spike pattern for semantic segmentation. *Neural Networks*, 176, 106330.
- Sui, Y., Yin, M., Gong, Y., Xiao, J., Phan, H., & Yuan, B. (2024a). ELRT: Efficient low-rank training for compact convolutional neural networks, arXiv preprint arXiv:2401.10341.
- Sui, Y., Yin, M., Gong, Y., & Yuan, B. (2024). Co-exploring structured sparsification and low-rank tensor decomposition for compact dnns. *IEEE Transactions on Neural Networks and Learning Systems*, 36, 6642–6654.
- Tang, Y., Wang, Y., Xu, Y., Tao, D., Xu, C., Xu, C., & Xu, C. (2020). SCOP: Scientific control for reliable neural network pruning. In *Adv. Neural Inf. Process. Syst.* (pp. 10936–10947). (vol. 33).
- Wang, N., Shi, Y., Ni, J., Fang, J., & Yu, B. (2024). Enhanced seismic attenuation compensation: Integrating attention mechanisms with residual learning in neural networks. *IEEE Transactions on Geoscience and Remote Sensing*, 62, 1–11.
- Wei, S., Luo, C., & Luo, Y. (2024). Scaled decoupled distillation. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.* (pp. 15975–15983).
- Wei, Y., & Bai, Y. (2024). Dynamic temperature knowledge distillation. arXiv preprint arXiv:2404.12711.
- Wen, T., Lai, S., & Qian, X. (2021). Preparing lessons: Improve knowledge distillation with better supervision. *Neurocomputing*, 454, 25–33.
- Xiang, Q., Zhang, M., Shang, Y., Wu, J., Yan, Y., & Nie, L. (2025). DKDM: Data-free knowledge distillation for diffusion models with any architecture. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.* (pp. 2955–2965).
- 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 *Proc. AAAI Conf. Artif. Intell.* (pp. 10464–10472).
- Xie, K., Liu, C., Wang, X., Li, X., Xie, G., Wen, J., & Li, K. (2024). Neural network compression based on tensor ring decomposition. *IEEE Transactions on Neural Networks and Learning Systems*, 36, 5388–5402.
- Xu, Y., Li, Y., Zhang, S., Wen, W., Wang, B., Qi, Y., Chen, Y., Lin, W., & Xiong, H. (2020). TRP: Trained rank pruning for efficient deep neural networks. arXiv preprint arXiv:2004.14566.
- Xue, J., & Shen, B. (2024). A survey on sparrow search algorithms and their applications. *International Journal of Systems Science*, 55(4), 814–832.
- Xue, Y., Li, M., Arabnejad, H., Suleimenova, D., Jahani, A., Geiger, B. C., Boesjes, F., Anagnostou, A., Taylor, S. J. E., & Liu, X. (2024). Many-objective simulation optimization for camp location problems in humanitarian logistics. *International Journal of Network Dynamics and Intelligence*, 3(3), 100017.
- Yang, H., Tang, M., Wen, W., Yan, F., Hu, D., Li, A., Li, H., & Chen, Y. (2020). Learning low-rank deep neural networks via singular vector orthogonality regularization and singular value sparsification. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops* (pp. 678–679).
- Yang, X., Yu, H., Gao, X., Wang, H., Zhang, J., & Li, T. (2024). Federated continual learning via knowledge fusion: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 36, 3832–3850.
- Yin, M., Sui, Y., Liao, S., & Yuan, B. (2022). BATUDE: Budget-aware neural network compression based on tucker decomposition. In *Proc. AAAI Conf. Artif. Intell.* (pp. 8874–8882).
- Yin, M., Sui, Y., Liao, S., & Yuan, B. (2021). Towards efficient tensor decomposition-based DNN model compression with optimization framework. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.* (pp. 10674–10683).
- Yu, W., Zhou, P., Yan, S., & Wang, X. (2024). InceptionNeXt: When inception meets ConvNeXt. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.* (pp. 5672–5683).
- Yuan, M., Zhang, C., Wang, Z., Liu, H., Pan, G., & Tang, H. (2024). Trainable spiking-YOLO for low-latency and high-performance object detection. *Neural Networks*, 172, 106092.
- Zayed, A., Mordido, G., Shabanian, S., Baldini, I., & Chandar, S. (2024). Fairness-aware structured pruning in transformers. In *Proc. AAAI Conf. Artif. Intell.* (pp. 22484–22492).
- Zeng, N., Li, X., Wu, P., Li, H., & Luo, X. (2024). A novel tensor decomposition-based efficient detector for low-altitude aerial objects with knowledge distillation scheme. *IEEE/CAA Journal of Automatica Sinica*, 11, 487–501.
- Zhan, Y., Yang, R., You, J., Huang, M., Liu, W., & Liu, X. (2025). A systematic literature review on incomplete multimodal learning: Techniques and challenges. *Systems Science and Control Engineering*, 13(1), 2467083.
- Zhang, Y., Gao, S., & Huang, H. (2021). Exploration and estimation for model compression. In *Proc. IEEE/CVF Int. Conf. Comput. Vis.* (pp. 487–496).
- Zhao, B., Cui, Q., Song, R., Qiu, Y., & Liang, J. (2022). Decoupled knowledge distillation. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.* (pp. 11953–11962).
- Zhuang, Z., Tan, M., Zhuang, B., Liu, J., Guo, Y., Wu, Q., Huang, J., & Zhu, J. (2018). Discrimination-aware channel pruning for deep neural networks. *Advances in Neural Information Processing Systems*, 31, 1–12.