IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received January 2, 2020, accepted January 15, 2020, date of publication January 21, 2020, date of current version January 28, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2968357

Bayesian Optimization-Based Global Optimal Rank Selection for Compression of Convolutional Neural Networks

TAEHYEON KIM, JIEUN LEE, AND YOONSIK CHOE, (Senior Member, IEEE)
School of Electrical and Electronic Engineering, Yonsei University, Seoul 120-749, South Korea
Corresponding author: Yoonsik Choe (yschoe@ yonsei.ac.kr)

ABSTRACT Recently, convolutional neural network (CNN) compression via low-rank decomposition has achieved remarkable performance. Finding the optimal rank is a crucial problem because rank is the only hyperparameter for controlling computational complexity and accuracy in compressed CNNs. In this paper, we propose a global optimal rank selection method based on Bayesian optimization (BayesOpt), which is a machine learning based global optimization technique. By utilizing both a simple objective function and a proper optimization scheme, the proposed method produces a global optimal rank that provides a good trade-off between computational complexity and accuracy degradation. In addition, our method also reflects the correlation of each rank in multi-rank selection, and is able to flexibly yield an optimal rank with a given fixed compression ratio. Experimental results indicate that the proposed algorithm can identify the global optimal rank regardless of the huge size of dataset or the various structural features of CNNs. In all experiments on multi-rank selection, the proposed method produces the rank with higher accuracy and lower computational complexity than the state-of-the-art rank selection method, variational Bayesian matrix factorization (VBMF).

INDEX TERMS Bayesian optimization, convolutional neural networks, neural network compression, low-rank decomposition, rank selection.

I. INTRODUCTION

Although convolutional neural networks (CNNs) with a large number of parameters and complex structures have achieved remarkable performance in various machine learning applications [1]–[5], there are serious implementation challenges caused by over-parameterized CNNs on resource constrained devices, such as mobile phones and embedded systems. To address these problems, CNN compression via low-rank decomposition has been proposed with various decomposition methods including singular value decomposition (SVD) [6], canonical polyadic decomposition (CPD) [7], tensor-train decomposition (TT decomposition) [8], [9], Tucker-2 decomposition [7], and block term decomposition [10]. The motivation behind low-rank decomposition for CNN compression is to search for a compact convolutional kernel that is close to the original by removing redundant components [11]. In CNN compression via low-rank decomposition, the rank is only a hyperparameter that controls the trade-off between compression rate and accuracy degradation. Thus, finding the optimal rank that provides a good trade-off on decomposed CNNs is the main purpose of the rank selection method. In this paper, the rank selection problem is separated into two categories-single- and multi-rank selection-depending on the number of elements that comprise the rank. For example, the rank selection of SVD and Tucker-2 decomposition are considered as single- and multi-rank selection, respectively, because SVD decomposes the data with one rank $\mathbf{r} \in \mathbb{R}$ and Tucker-2 decomposition decomposes the data with two ranks $\mathbf{r} \in \mathbb{R}^2$.

To find the optimal rank, several rank selection methods have been proposed. *Lebedev et al.* [7], who used CPD, proposed the heuristic rank selection method. However, because the value of the optimal rank is very responsive to changes in the dataset, structure of the CNNs, etc., this method is not universal and practical. *Jaderberg et al.* [6], who employed SVD, tried to find optimal rank by optimizing the accumulated principal component analysis (PCA) energy-based objective function without a definite optimization scheme. The resulting method is therefore time-consuming, compounding the

The associate editor coordinating the review of this manuscript and approving it for publication was Vicente Alarcon-Aquino.

multi-rank selection problem. *Saha et al.* [43] proposed a fitness based rank selection method with a heuristically defined optimization scheme. However, this method is only suitable for single-rank selection and may suffer from convergence problems because of their optimization schemes. *Kim et al.* [7] used Tucker-2 decomposition and proposed a variational Bayesian matrix factorization (VBMF) [23] as a tool to determine Tucker-2 rank. Because each rank in Tucker-2 decomposition is defined as the rank of the unfolded tensor that is a form of the matrix, VBMF can automatically find the noise variance and rank, and even provide a solid theoretical foundation for perfect rank recovery. Hence, VBMF has been used in other rank selection method to date [42]. However, as VBMF is a matrix-based algorithm, it is not considered a proper rank selection method in some low-rank tensor decompositions, such as CPD, wherein the CP rank is defined by the number of independent rank-one tensors. In addition, as this method does not reflect the correlation among ranks, it provides suboptimal rank in a multi-rank selection problem, such as Tucker-2 decomposition. Therefore, VBMF is limited in its applicability to multi-rank selection problems.

In this paper, we propose a novel global optimal rank selection algorithm, which can be used in all low-rank decomposition methods; in other words, our method can provide the global optimal rank in both single- and multi-rank selection problems. The proposed objective function contains computational complexity and reconstruction losses to represent the trade-off between compression rate and accuracy degradation, which is controlled by rank selection. To address devices with limited resource, we placed the computational complexity loss term in a nonlinear function (rectifier), which yields zero if the input is smaller than threshold; otherwise, it returns the input value. Furthermore, our method reflects the correlation among ranks in multi-rank selection problems. To obtain the global optimal rank, the loss function has been optimized using Bayesian optimization (BayesOpt), which is an efficient global optimization technique [12]–[15].

To employ BayesOpt, the loss function must be in the form of a continuous function but not necessary to be differentiable. Hence, we converted the proposed objective function, which is a discrete function due to the rank characteristic, to a continuous step function that is non-differentiable. The BayesOpt constructs a probabilistic model of the objective function to make decisions on where the feasible rank set is to next evaluate the objective function while factoring out uncertainty. Therefore, utilizing BayesOpt is not only a reasonable way to optimize our proposed objective function, but also an advanced way that produces a global optimal rank that balances computational complexity and accuracy degradation in every low-rank decomposition method. We performed CNN compression experiments based on SVD and Tucker-2 decomposition to confirm the single- and multi-rank selection performance of the proposed technique, respectively. Furthermore, to demonstrate the generalization performance of the proposed algorithm for various CNNs on various datasets, we conducted experiments on five representative models, namely, AlexNet [16], VGG-16 [17], ResNet18 [18], GoogLeNet [19], and SqueezeNet [20] on three benchmark datasets [21], [34].

In summary, the main contributions of this study are as follows:

- The proposed method produces the global optimal rank, which provides a good trade-off between computational complexity and accuracy degradation of compressed CNNs, by using both the simple objective function and a machine learning-based optimization algorithm.
- The global optimal rank, produced by the proposed algorithm, works for all low-rank decompositions of CNNs and all CNN structures. Furthermore, in multi-rank selection, the proposed method reflects the correlation between each rank.
- The proposed algorithm is a flexible rank selection method that can produce the global optimal rank for constrained resource devices, by just changing the hyperparameter in the proposed objective function.

The remainder of this paper is organized as follows. Section II briefly introduces SVD and Tucker-2 decomposition for CNN compression, VBMF, and BayesOpt. Section III discusses our global optimal rank selection algorithm, focusing on the proposed objective function and BayesOpt with both Màtern 5/2 covariance function and expected improvement acquisition function. Obtained experimental results are presented and analyzed in Section IV. Finally, Section V summarizes this paper.

II. PRELIMINARIES

A. LOW-RANK DECOMPOSITION FOR CNN COMPRESSION

In CNNs, the convolution kernel $\underline{\mathbf{W}} \in \mathbb{R}^{D\times D\times S\times T}$ is a four-way tensor, where D is the spatial window size, S and T are the depth values of the input and output feature maps, respectively. The inference operation of CNNs can be represented by linear mapping that input feature-maps $\underline{\mathbf{X}} \in \mathbb{R}^{H\times W\times S}$ into output feature maps $\underline{\mathbf{Y}} \in \mathbb{R}^{H'\times W'\times T}$, which is as follows:

$$\begin{aligned}\underline{\mathbf{Y}}(h', w', t) &= \sum_{i=1}^{D}\sum_{j=1}^{D}\sum_{s=1}^{S}\underline{\mathbf{W}}(i, j, s, t)\underline{\mathbf{X}}(h_i, w_i, s),\\ h_i &= (h'-1)\triangle + i - P,\\ w_j &= (w'-1)\triangle + j - P,\end{aligned} \tag{1}$$

where $\underline{\mathbf{X}}(a, b, c)$ is an element with coordinates (a, b, c) of the tensor $\underline{\mathbf{X}}$; $\triangle$ is stride; and P is size of zero-padding. The full convolution operation of the convolutional layer is illustrated in Fig. 1(a). Thus, the number of parameters (memory complexity) and multiplication-addition operations (computational complexity) of (1) are equivalent to D^2ST and $D^2STW'H'$, respectively. Following these definitions, we define the compression rate of memory complexity $c_m(\mathbf{r})$ and computational complexity $c_t(\mathbf{r})$ according to rank $\mathbf{r}$ as

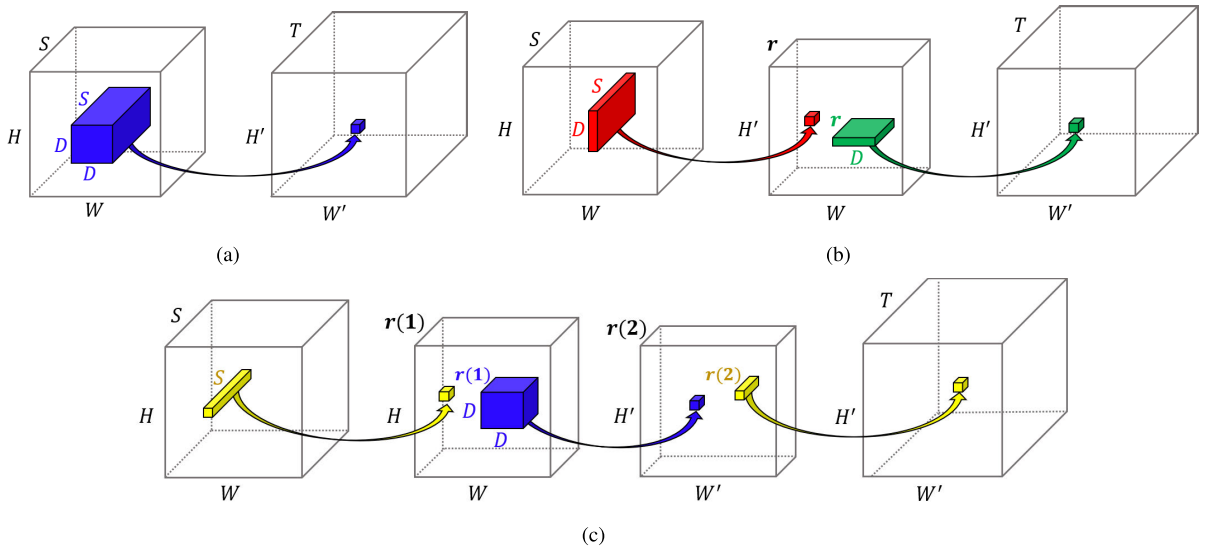


FIGURE 1. The transparent 3D tensors are feature maps in the convolutional layer and the arrows are linear mappings showing that input feature maps into the output feature maps. Blue 3D tensors are 2D convolution kernels whereas yellow 3D tensors are 1 × 1 convolution kernels. Red and green 3D tensors are 1D convolution kernels. Small-sized 3D tensors are single elements of the target feature-maps. (a) Full convolution in (1). (b) SVD on convolution kernel from (3) to (4) [6]. (c) Tucker-2 decomposition on convolution kernel from (8) to (10) [7].

follows:

$$c_m(\mathbf{r}) = \frac{\text{number of parameters of } \underline{\mathbf{W}} \text{ decomposed with } \mathbf{r}}{\text{number of parameters of } \underline{\mathbf{W}}},$$
$$c_t(\mathbf{r}) = \frac{\text{number of operations of } \underline{\mathbf{W}} \text{ decomposed with } \mathbf{r}}{\text{number of operations of } \underline{\mathbf{W}}}.$$

As previously mentioned, we divided the rank selection problem into two categories, according to how many components are associated with rank $\mathbf{r}$: single-rank selection problem and multi-rank selection problem. Therefore, to evaluate single- and multi-rank selection via our proposed method, we utilized SVD [6] and Tucker-2 decomposition [7] formulations, which require single-rank $\mathbf{r} \in \mathbb{R}$ and multi-rank $\mathbf{r} \in \mathbb{R}^2$, respectively.

1) SINGULAR VALUE DECOMPOSITION (SVD) ON CONVOLUTION KERNEL

SVD has been used as a simple tool to decompose convolution kernels $\underline{\mathbf{W}}$ into two components with a single rank:

$$\underline{\mathbf{W}}(i,j,s,t) \approx \hat{\underline{\mathbf{W}}}(i,j,s,t) = \sum_{r=1}^{\mathbf{r}} \underline{\mathbf{U}}(i,1,s,r)\underline{\mathbf{V}}(1,j,s,t), \quad (2)$$

where $\hat{\underline{\mathbf{W}}}$ is low-rank decomposed convolution kernels; and $\underline{\mathbf{U}} \in \mathbb{R}^{D\times 1\times S\times \mathbf{r}}$ and $\underline{\mathbf{V}} \in \mathbb{R}^{1\times D\times \mathbf{r}\times T}$ are factors from SVD. After substituting the decomposed kernels into the original kernel, we obtain the following two consecutive expressions for the convolution operation with a low-rank approximated kernel:

$$\underline{\mathbf{Z}}(h',w,r) = \sum_{i=1}^{D}\sum_{s=1}^{S} \underline{\mathbf{U}}(i,1,s,r)\underline{\mathbf{X}}(h_i,w,s), \quad (3)$$
$$\underline{\mathbf{Y}}(h',w',t) = \sum_{j=1}^{D}\sum_{r=1}^{\mathbf{r}} \underline{\mathbf{V}}(1,j,r,t)\underline{\mathbf{Z}}(h',w_i,r), \quad (4)$$

where $\underline{\mathbf{Z}}$ is intermediate feature maps of sizes $H' \times W \times \mathbf{r}$. The output tensor $\underline{\mathbf{Y}}$ can be expressed by substituting $\underline{\mathbf{V}}$ and $\underline{\mathbf{Z}}$.

From the above equations, we can see that the SVD separates the $D \times D$ convolution kernel into $D \times 1$ and $1 \times D$ convolution kernels. The Fig. 1(b) shows this SVD-based low-rank decomposition of convolution kernels. In CNN compression via SVD, the compression rates of memory complexity $c_m(\mathbf{r})$ and computational complexity $c_t(\mathbf{r})$ are as follows:

$$c_m(\mathbf{r}) = \frac{DS\mathbf{r} + DT\mathbf{r}}{D^2ST}, \quad (5)$$
$$c_t(\mathbf{r}) = \frac{DSWH'\mathbf{r} + DTW'H'\mathbf{r}}{D^2STW'H'}. \quad (6)$$

From (5) and (6), we demonstrated that rank $\mathbf{r}$ is the only hyperparameter that controls the computational complexity of the decomposed model because all the remaining parameters have fixed values.

2) TUCKER-2 DECOMPOSITION ON CONVOLUTION KERNEL

Tucker2-decomposition on convolution kernels can decompose the original convolution kernel into three factors with two ranks. Hence, the rank of Tucker-2 decomposition $\mathbf{r} \in \mathbb{R}^2$ is composed of both $\mathbf{r}(1)$ and $\mathbf{r}(2)$. Because the spatial size of recent convolution kernels has already become small, *e.g.*, 3×3 or 5×5, Tucker-2 decomposition may not be necessary to decompose the first and second axes of convolution kernels $\underline{\mathbf{W}}$, shown below:

$$\underline{\mathbf{W}}(i,j,s,t) \approx \hat{\underline{\mathbf{W}}}(i,j,s,t) = \sum_{r_1=1}^{\mathbf{r}(1)}\sum_{r_2=1}^{\mathbf{r}(2)} \underline{\mathbf{C}}(i,j,r_1,r_2)\underline{\mathbf{U}}^{(1)}(1,1,s,r_1)\underline{\mathbf{U}}^{(2)}(1,1,r_2,t), \quad (7)$$

where $\underline{\mathbf{C}}$ is the core tensor of size $D \times D \times \mathbf{r}(1) \times \mathbf{r}(2)$; and $\underline{\mathbf{U}}^{(1)} \in \mathbb{R}^{1\times 1\times S\times \mathbf{r}(1)}$ and $\underline{\mathbf{U}}^{(2)} \in \mathbb{R}^{1\times 1\times \mathbf{r}(2)\times T}$ are the projection factors of the third and fourth axis of $\underline{\mathbf{W}}$, respectively. $\mathbf{r}(1)$ and $\mathbf{r}(2)$ are indicated by the number of independent columns of mode-3 matricization (at size $S \times DDT$) and mode-4 matricization (at size $T \times DDS$) of the convolution kernel, respectively. After the Tucker-2 decomposition is conducted on the convolution kernel $\underline{\mathbf{W}}$, the original kernel in (1) can be replaced by low-rank approximated convolution kernel in (7) as follows:

$$\underline{\mathbf{Z}}^{(1)}(h, w, r_1) = \sum_{s=1}^{S} \underline{\mathbf{U}}^{(1)}(1, 1, s, r_1)\underline{\mathbf{X}}(h, w, s), \tag{8}$$

$$\underline{\mathbf{Z}}^{(2)}(h', w', r_2) = \sum_{i=1}^{D}\sum_{j=1}^{D}\sum_{r_1=1}^{\mathbf{r}(1)} \underline{\mathbf{C}}(1, j, r_1, r_2)\underline{\mathbf{Z}}^{(1)}(h_i, w_i, r_1), \tag{9}$$

$$\underline{\mathbf{Y}}(h', w', t) = \sum_{r_2=1}^{\mathbf{r}(2)} \underline{\mathbf{U}}^{(2)}(1, 1, r_2, t)\underline{\mathbf{Z}}^{(2)}(h', w', r_2), \tag{10}$$

where $\underline{\mathbf{Z}}^{(1)}$ and $\underline{\mathbf{Z}}^{(2)}$ are intermediate tensors of sizes $H \times W \times \mathbf{r}(1)$ and $H' \times W' \times \mathbf{r}(2)$, respectively. The Fig. 1(c) shows the overall process of consecutive operations from (8) to (10). We can see that Tucker-2 decomposition decomposes the $D \times D$ convolution kernel into 1×1 convolution kernel for dimension reduction, $D \times D$ convolution kernel and 1×1 convolution kernel for dimension expansion, sequentially. Analogous to both (5) and (6), the memory compression rates of memory complexity and computational complexity are as follows:

$$c_m(\mathbf{r}) = \frac{S\mathbf{r}(1) + D^2\mathbf{r}(1)\mathbf{r}(2) + T\mathbf{r}(2)}{D^2ST}, \tag{11}$$

$$c_t(\mathbf{r}) = \frac{S\mathbf{r}(1)WH + D^2\mathbf{r}(1)\mathbf{r}(2)W'H' + T\mathbf{r}(2)W'H'}{D^2STW'H'}. \tag{12}$$

Equations (11) and (12) demonstrate that the correlation among ranks must be considered to ensure efficient compression performance in multi-rank selection.

B. RANK SELECTION VIA VBMF

Recently, VBMF has become a promising tool used in layerwise rank selection, which was first employed in [7]. This is because the VBMF can automatically find noise variance, rank, and even provide theoretical condition for rank recovery [22], [23]. However, because the VBMF is based on matrix format data, it cannot produce optimal rank on all low-rank decomposition on CNNs. For example, in [7], ranks $\mathbf{r}(1)$ and $\mathbf{r}(2)$ were determined from (7) by applying this algorithm on mode-3 matricization $\in \mathbb{R}^{S\times D^2T}$ and mode-4 matricization $\in \mathbb{R}^{T\times D^2S}$ of convolution kernel $\underline{\mathbf{W}}$, respectively. However, as indicated in (11) and (12), because the computational complexity and accuracy degradation of the decomposed convolutional layer are simultaneously related with $\mathbf{r}(1)$ and $\mathbf{r}(2)$, the rank selection method must consider the correlation of each rank in multi-rank selection. Therefore, multi-rank selection via VBMF cannot be an optimal rank. The importance of correlation in multi-rank selection will be discussed in Section IV as we compare the performance of the proposed method with VBMF through Tucker-2 decomposition experiments.

C. BAYESIAN OPTIMIZATION (BAYESOPT)

BayesOpt is a class of machine learning-based optimization methods. BayesOpt consists of two major components: a Bayesian statistical model for modeling the objective function and an acquisition function for deciding the next sampling points [12]–[15]. The Bayesian statistical model, which is a Gaussian Process (GP), provides quantified uncertainty of objective function values at an unobserved data [24]. The acquisition function measures the predictive enhancement at an unobserved data, to determine the next sampling point [25], [33]. Therefore, BayesOpt is designed for black-box derivative-free global optimization because 1) it does not require the structural information of objective function (black-box); 2) it does not observe the derivatives of objective function (derivative-free); and 3) it finds the global optimum by calculating the uncertainty of the objective function at unobserved points (global optimization). These properties not only simplify the design complexity of the objective function for rank selection, but also enable global optimal rank selection through the objective function. By utilizing BayesOpt, our proposed algorithm can produce the global optimal rank on all decompositions using the designed objective function.

III. PROPOSED METHOD

In this section, we explain the global optimal rank selection method, which applies to all low-rank decompositions, including single- and multi-rank selection via BayesOpt. In addition, we show that the proposed method is able to select the optimal rank flexibly with a given fixed compression ratio. The overall process of CNN compression via the proposed algorithm is summarized as follows:

1) Train the CNN with a dataset and select any low-rank decomposition method for CNN compression.
2) Compute the global optimal rank for all convolutional layers via proposed method. More details are given in Algorithm 1.
3) Compress the trained CNN using the chosen low-rank decomposition with the computed global optimal rank.
4) To compensate for the errors caused by the low-rank decomposition, fine-tune the compressed CNN using the dataset.

A. PROPOSED OBJECTIVE FUNCTION

The proposed algorithm aims to find a global optimal rank that guarantees both low computational complexity and low accuracy degradation in decomposed neural networks. Therefore, the proposed objective function f is simply formulated through the superposition between computational complexity loss c_t and reconstruction loss c_r, which are easily

Algorithm 1 Basic Pseudo-Code for Proposed Global Optimal Rank Selection Method

Perform the initial space-filling experiment by randomly sampling the objective function f on n_0 ranks.
Assume the prior multi-variate normal distribution using GP with Màtern 5/2 covariance function in (16).
Estimate the hyperparameter of Màtern 5/2 covariance function via maximum a posteriori in (18).
Set n as n_0.
Initialize maximum iteration number M and mimimum Euclidean distance F.
while $n <= M$ **and** Euclidean distance with previous observed rank $> F$ **do**
  Calculate the posterior distribution of objective function f on unobserved rank $\mathbf{r}^*$ by using observed rank $f(\mathbf{r_{1:n}})$ according to the Bayesian rule in (19)
  Estimate the hyperparameters of Màtern 5/2 covariance function via maximum a posteriori in (18)
  Initialize the $\mathbf{r_{best}}$ as $\arg\min_{\mathbf{r_{1:n}}} f(\mathbf{r_{1:n}})$
  Compute expected improvement acquisition function using current posterior distribution in (21)
  Set $\mathbf{r_{n+1}}$ as a maximizer of the expected improvement acquisition function in (22)
  Observe $f(\mathbf{r_{n+1}})$.
  $n = n + 1$
end while
Return a solution: $\mathbf{r_{best}}$

computable in all low-rank decomposition as below:

$$\begin{aligned} \tilde{\mathbf{r}} &= \arg\min f(\mathbf{r}), \\ f(\mathbf{r}) &= c_r(\mathbf{r}) + c_t(\mathbf{r}), \\ c_r(\mathbf{r} &= \frac{||\underline{\mathbf{W}} - \underline{\hat{\mathbf{W}}}||_F^2}{||\underline{\mathbf{W}}||_F^2}, \end{aligned} \tag{13}$$

where $\tilde{\mathbf{r}}$ means global optimal rank, and reconstruction loss c_r is measured by the squared Frobenius norm. The squared Frobenius norm of a convolution kernel $\underline{\mathbf{W}} \in \mathbb{R}^{D\times D\times S\times T}$ is defined as the sum of the squares of its elements:

$$||\underline{\mathbf{W}}||_F^2 = \sum_{i=1}^{D}\sum_{j=1}^{D}\sum_{s=1}^{S}\sum_{t=1}^{T}\underline{\mathbf{W}}(i,j,s,t). \tag{14}$$

As reconstruction loss c_r is linear with accuracy degradation, the proposed objective function (13) represents the trade-off relationship between computational complexity and accuracy degradation based on rank selection. In addition, the objective function in a multi-rank selection problem has a form of the multimodal function to reflect the correlation of each rank. Therefore, the proposed rank selection method based on the objective function is available in all low-rank decompositions, including both single- and multi-rank selections.

However, as rank $\mathbf{r}$ is discrete data, the objective function has the form of a discrete function which is not applicable to BayesOpt. To address this problem, we simply change the proposed objective function to a continuous step function. Although the changed objective function with the form of a step function is not differentiable, BayesOpt is derivative-free; hence, it is possible to find the global optimal rank via the proposed objective function.

For devices with limited resources, practical CNN compression techniques require maximum performance under the condition of fixed computational complexity. To handle this demand, we transform (13) into its final objective function:

$$\begin{aligned} \tilde{\mathbf{r}} &= \arg\min f(\mathbf{r}), \\ f(\mathbf{r}) &= c_r(\mathbf{r}) + g(c_t(\mathbf{r}), \alpha), \\ g(x,\alpha) &= \begin{cases} x & \text{if } x > \alpha \\ 0 & \text{otherwise} \end{cases}, \end{aligned} \tag{15}$$

where $g(x, \alpha)$ is a rectifier function that either provides an input value, if the input value is greater than the threshold α, or zero. By setting α to zero, there is no condition for computational complexity. Therefore, we can use the objective function by controlling the computational complexity threshold α at the rectifier.

In summary, the proposed objective function has four properties: 1) It represents the relationship between computational complexity and accuracy degradation for a low-rank approximated convolution kernel according to rank selection. 2) Because the objective function involves the dependencies of each rank in a multi-rank selection, and each loss term is readily computable on any low-rank decompositions, it can be used for all low-rank decomposition, including both single- and multi-rank selections. 3) It can be applied to rank selection problems with fixed computational complexity by using the computational complexity loss rectifier with threshold α. 4) It has a continuous function form that can be globally optimized via BayesOpt.

To verify the properties of the objective function, we performed Tucker-2 decomposition on the convolution kernel of the third convolutional layer of AlexNet, which is trained on ImageNet classification. This convolution kernel has a size of $3 \times 3 \times 192 \times 382$. Figs. 2(a), 2(b), and 2(c) show the surfaces of computational complexity loss, reconstruction loss, and objective function, respectively. It can be confirmed that the computational complexity loss follows an increasing function with respect to rank values whereas the reconstruction loss does not. In addition, the objective

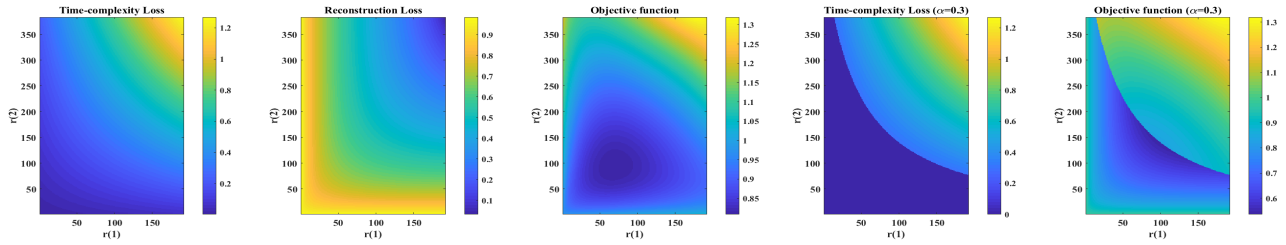


FIGURE 2. All sub-figures cover Tucker-2 decomposition on the third convolutional layer of AlexNet, which is trained on ImageNet classification. The x- and y-axes of all sub-figures are r(1) and r(2), respectively. (a) Surface of computational complexity loss $c_t(\mathbf{r})$. (b) Surface of reconstruction loss $c_r(\mathbf{r})$. (c) Surface of objective function without rectifier ($\alpha = 0$) in (15). (d) Surface of computational complexity loss with rectifier ($\alpha = 0.3$). (e) Surface of objective function with rectifier ($\alpha = 0.3$) in (15).

function represents the trade-off between the two loss terms. Figs. 2(d) and 2(e) show the surfaces of computational complexity loss and objective function, respectively when α is 0.3, suggesting that the computational complexity is zero when the threshold value is less than 0.3. It creates a steep cliff on the surface of the objective function and moves the optimal rank point near this cliff. All sub-figures in Fig. 2 show the dependencies between $\mathbf{r}(1)$ and $\mathbf{r}(2)$ via multimodal objective function.

B. GAUSSIAN PROCESS (GP) REGRESSION ON PROPOSED OBJECTIVE FUNCTION

In the proposed method, GP regression provides the posterior distribution that represents the potential values of the proposed objective function $f(\mathbf{r}^*)$ at unevaluated ranks $\mathbf{r}^*$, based on a prior distribution taken from previously observed k ranks $\mathbf{r_1}, \ldots, \mathbf{r_k} \in \mathbb{R}^D$, where the D denotes the value of rank dimensionality; therefore, if $\mathbf{r}$ is the rank of Tucker-2 decomposition than D is two. In other words, this algorithm quantitatively estimates the uncertainty of unconfirmed loss values $f(\mathbf{r}^*)$ based on revealed objective function values $[f(\mathbf{r_1}), \ldots, f(\mathbf{r_k})]$ obtained from either previous optimization iterations or initial space-filling experiments. For more compact notation, we describe this finite collection of k ranks as $\mathbf{r_{1:k}}$, and denote the loss value vector of $\mathbf{r_{1:k}}$ as $f(\mathbf{r_{1:k}}) = [f(\mathbf{r_1}), \ldots, f(\mathbf{r_k})] \in \mathbb{R}^k$. Following the Bayesian rule, we must define the prior distribution of $f(\mathbf{r_{1:k}})$ to obtain the conditional distribution $P(f(\mathbf{r}^*)|f(\mathbf{r_{1:k}}))$. Therefore, we assume the prior distribution of $f(\mathbf{r_{1:k}})$ to be a multivariate normal distribution by using the mean function μ_0 and Màtern 5/2 covariance function Σ_0. The equations are as follows:

$$P(f(\mathbf{r_{1:k}})) \sim \text{Normal}(\mu_0(\mathbf{r_{1:k}}), \Sigma_0(\mathbf{r_{1:k}}, \mathbf{r_{1:k}})), \tag{16}$$

$$\Sigma_0(\mathbf{r_i}, \mathbf{r_j}) = \theta_0\Big(1 + \sqrt{5v^2(\mathbf{r_i}, \mathbf{r_j})} + \frac{5}{3}v^2(\mathbf{r_i}, \mathbf{r_j})\Big) \times \exp\Big\{-\sqrt{5\, v^2(\mathbf{r_i}, \mathbf{r_j})}\Big\},$$

$$v^2(\mathbf{r_i}, \mathbf{r_j}) = \sum_{d=1}^{D} \frac{(\mathbf{r_i}(d) - \mathbf{r_j}(d))^2}{\theta_d^2}, \tag{17}$$

where $\theta_{0:D} = [\theta_0, .., \theta_D]$ is the hyper-parameter of Σ_0. In particular, the mean vector $\mu_0(\mathbf{r_{1:k}})$ and Màtern 5/2 covariance matrix $\Sigma_0(\mathbf{r_{1:k}}, \mathbf{r_{1:k}})$ are:

$$\mu_0(\mathbf{r_{1:k}}) = [\mu_0(\mathbf{r_1}), \ldots, \mu_0(\mathbf{r_k})] \in \mathbb{R}^k,$$

$$\Sigma_0(\mathbf{r_{1:k}}, \mathbf{r_{1:k}}) = \begin{bmatrix} \Sigma_0(\mathbf{r_1}, \mathbf{r_1}) & \cdots & \Sigma_0(\mathbf{r_1}, \mathbf{r_k}) \\ \vdots & \ddots & \vdots \\ \Sigma_0(\mathbf{r_k}, \mathbf{r_1}) & \cdots & \Sigma_0(\mathbf{r_k}, \mathbf{r_k}) \end{bmatrix} \in \mathbb{R}^{k \times k}.$$

The Màtern 5/2 covariance function is commonly applied for the BayesOpt problem because GP regression with this covariance function is smooth in effect under practical optimization problems [14]. The hyper-parameters $\theta_{0:D}$ in Màtern 5/2 covariance function play a role in the design of prior distribution on $f(\mathbf{r_{1:k}})$, and they can be estimated using maximum a posteriori (MAP) estimation, as follows:

$$\begin{aligned} \hat{\theta}_{0:D} &= \arg\max_{\theta_{0:D}} P(\theta_{0:D}|f(\mathbf{r_{1:k}})) \\ &= \arg\max_{\theta_{0:D}} P(f(\mathbf{r_{1:k}})|\theta_{0:D})P(\theta_{0:D}). \end{aligned} \tag{18}$$

By using the prior distribution given by (16), we can calculate the conditional distribution of $f(\mathbf{r}^*)$:

$$\begin{aligned} &P(f(\mathbf{r}^*)|f(\mathbf{r_{1:k}})) \sim Normal(\mu_k(\mathbf{r}^*), \sigma_k^2(\mathbf{r}^*)), \\ &\mu_k(\mathbf{r}^*) \\ &= \Sigma_0(\mathbf{r}^*, \mathbf{r_{1:k}})\Sigma_0(\mathbf{r}^*, \mathbf{r_{1:k}})^{-1}(f(\mathbf{r_{1:k}}) - \mu_0(\mathbf{r_{1:k}})) + \mu_0(\mathbf{r_{1:k}}), \\ &\sigma_k^2(\mathbf{r}^*) \\ &= \Sigma_0(\mathbf{r}^*, \mathbf{r}^*) - \Sigma_0(\mathbf{r}^*, \mathbf{r_{1:k}})\Sigma_0(\mathbf{r_{1:k}}, \mathbf{r_{1:k}})^{-1}\Sigma_0(\mathbf{r_{1:k}}, \mathbf{r}^*). \end{aligned} \tag{19}$$

Therefore, we can define the potential values of the proposed objective function in a rank that has not been evaluated from a conditional distribution, typically associated with the prior distribution of k observed ranks. Fig. 3 shows the results of the proposed algorithm for Tucker-2 decomposition on the third convolutional layer of AlexNet. In particular, the figures at the bottom establish the consequences of the proposed algorithm with α of 0.3. Figs. 3(a) and (b) represent posterior mean and posterior standard deviation, respectively, which are obtained from GP with Màtern 5/2 covariance function.

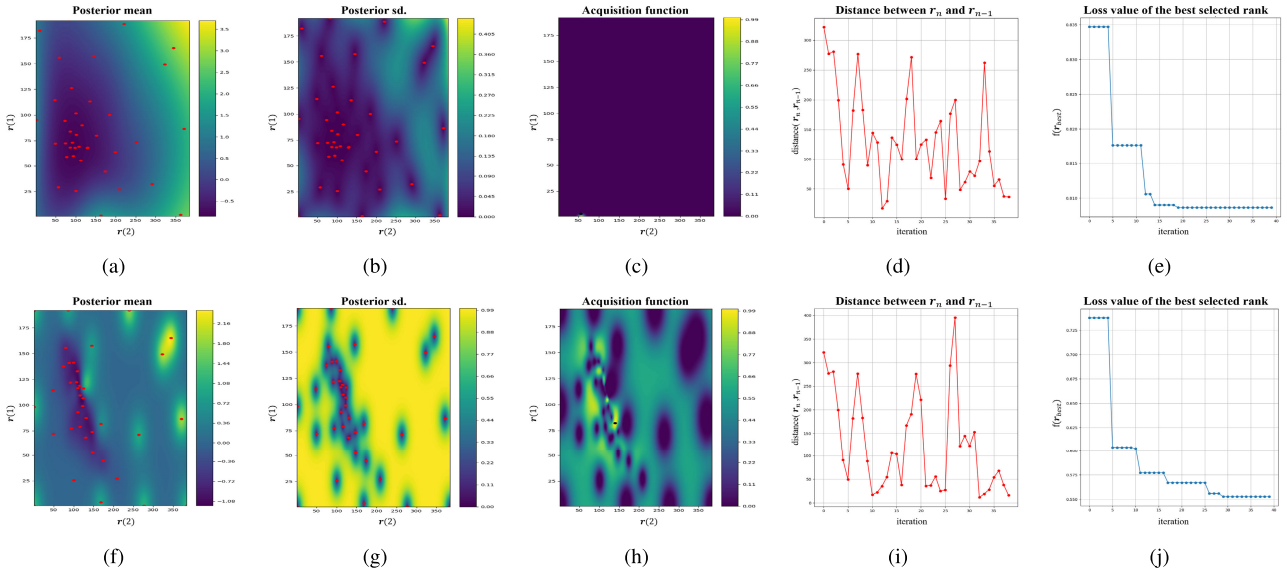


FIGURE 3. All sub-figures cover Tucker-2 decomposition on the third convolutional layer of AlexNet, which is trained on ImageNet classification. Top BayesOpt on objective function without rectifier ($\alpha = 0$) in (15). Bottom BayesOpt on objective function with rectifier ($\alpha = 0.3$) in (15). (a) and (f) Mean of posterior distribution $\mu_k(\mathbf{r}^*)$ in (19). The x- and y-axes are r(2) and r(1), respectively. (b) and (g) Standard deviations of posterior distribution $\sigma_k^2(\mathbf{r}^*)$ in (19). The x- and y-axes are r(2) and r(1), respectively. (c) and (h) Expected improvement acquisition function $\mathrm{EI}_k(\mathbf{r}^*)$ in (21). The x- and y-axes are r(2) and r(1), respectively. (d) and (i) Distance between consecutive observed ranks in all iterations. The x- and y-axes are iteration and the Euclidean distance respectively. (e) and (j) Objective function value of the best rank $f(\mathbf{r_{best}})$ in all iterations. The x- and y-axes are iteration and $f(\mathbf{r_{best}})$, respectively.

The red dots in the Figs. 3(a) and (b) denote the 40 observed rank points. As the Màtern 5/2 covariance function has the property that points close in the input space are more strongly correlated, it can be deemed that the standard deviation of the unobserved points close to the observation point has a small value and vice versa. Figs. 3(f) and (g) also show similar tendencies. However, owing to the large distortion by the computational complexity rectifier, the standard deviation of the unobserved data points far from the observed data points is high.

C. EXPECTED IMPROVEMENT (EI) FOR GLOBAL OPTIMAL RANK

In our proposed algorithm, we utilize the expected improvement acquisition function to determine the next sampling rank point [25]. The key idea of this acquisition function is to return the best rank that yields the lowest loss among the already observed ranks. As the proposed objective function generates the loss of observed rank without noise, there is no doubt that the best current rank $\mathbf{r_{best}}$ is the rank that has the least error among those previously evaluated. Hence, it is appropriate to use the expected improvement (EI) acquisition function in BayesOpt framework to optimize the proposed objective function. In this section, we denote the best current rank as $\mathbf{r_{best}} = \arg\min_{\mathbf{r_{1:k}}} f(\mathbf{r_{1:k}})$, the cumulative distribution function of the normal distribution as Φ, and the probability density function of normal distribution as ϕ.

To quantify the improvement at candidate rank $\mathbf{r}^*$, the expected improvement acquisition function compares the loss value of current best rank $f(\mathbf{r_{best}})$ and the approximated objective error of candidate ranks $f(\mathbf{r}^*)$. Therefore, the expected improvement can be defined as follows:

$$\mathrm{EI}_k(\mathbf{r}^*) := E_k[\max(f(\mathbf{r_{best}}) - f(\mathbf{r}^*), 0)], \tag{20}$$

Therefore the EI acquisition function in proposed method computes the expectation taken under the conditional distribution $P(f(\mathbf{r}^*)|f(\mathbf{r_{1:k}}))$ of utility function which returns a positive value if $f(\mathbf{r}^*) \leq f(\mathbf{r_{best}})$ or zero if $f(\mathbf{r}^*) > f(\mathbf{r_{best}})$ [25]. Another main reason of utilizing the EI acquisition function is that it has a closed form under GP regression [12], [14], [33]. Thus (20) can be changed into a closed form, as follows:

$$\begin{aligned} \mathrm{EI}_k(\mathbf{r}^*) &= \sigma_k(\mathbf{r}^*)(\gamma_k(\mathbf{r}^*)\Phi(\gamma_k(\mathbf{r}^*)) + \phi(\gamma_k(\mathbf{r}^*)), \\ \gamma_k(\mathbf{r}^*) &= \frac{f(\mathbf{r_{best}}) - \mu_k(\mathbf{r}^*)}{\sigma_k(\mathbf{r}^*)}. \end{aligned} \tag{21}$$

To determine the next rank point $\mathbf{r_{k+1}}$, which is expected to be closer to the global optimal rank $\tilde{\mathbf{r}}$ than the current best rank $\mathbf{r_{best}}$, the expected improvement acquisition function evaluates the candidate rank $\mathbf{r}^*$ with the largest expected improvement:

$$\mathbf{r_{k+1}} = \arg\max \mathrm{EI}_k(\mathbf{r}^*). \tag{22}$$

After the EI, if the BayesOpt termination condition is satisfied, then the obtained rank from the expected improvement function $\mathbf{r_{k+1}}$ becomes the solution of the proposed algorithm; otherwise, it will be used to construct the posterior distribution through GP regression. Figs. 3(c) and (h) show the EI acquisition function value of each experiment. In Fig. 3(c), almost all values of the acquisition function are zero, *i.e.*, the proposed algorithm converges to the

TABLE 1. Performance comparison of SVD for CNN compression using VBMF and the proposed method.

Dataset	Model	Rank Selection	Top-1 Accuracy (%)	Top-5 Accuracy (%)	# of Parameters (Mega)	FLOPS (Giga)
ImageNet	AlexNet	Base	56.55	75.09	2.47 (×1.0)	1.31 (×1.0)
		Ours	48.88±0.05 (-7.67)	72.82±0.07 (-2.27)	0.51 (×4.84)	**0.30 (×4.36)**
		VBMF	**49.20±0.02 (-7.35)**	**73.13±0.05 (-1.96)**	**0.43 (×5.74)**	0.33 (×3.96)
CIFAR-100	AlexNet	Base	54.85	80.28	2.25 (×1.0)	0.05 (×1.0)
		Ours	**36.30±0.25 (-18.55)**	**67.70±0.15 (-12.58)**	0.42 (×5.35)	0.02 (×2.50)
		VBMF	35.68±0.35 (-19.17)	67.57±0.25 (-12.71)	**0.28 (×8.04)**	**0.01 (×5.0)**
CIFAR-10	AlexNet	Base	78.30	-	2.25 (×1.0)	0.05 (×1.0)
		Ours	**67.56±0.42 (-10.74)**	-	**0.30 (×7.50)**	**0.01 (×5.0)**
		VBMF	67.36±0.21 (-10.94)	-	0.32 (×7.03)	**0.01 (×5.0)**

global optimal rank. Fig, 3(d) shows the Euclidean distance between two consecutive ranks observed in all iterations. As the expected improvement acquisition function picks the next sampling point based on uncertainty, the distance between the observed consecutive ranks has irregular fluctuations in all iterations, verifying that BayesOpt globally optimizes the proposed objective function. Fig. 3(e) shows the loss of the current best rank $f(\mathbf{r_{best}})$ in all iterations and confirms that BayesOpt converge to the global optimal rank. The proposed method with the $\alpha = 0.3$ case is also represented in Figs. 3(h-j), which have similar trends with Figs. 3(a-e). Therefore, we can verify that the proposed method converges to the global optimal rank.

IV. EXPERIMENTAL RESULTS

In this section, both SVD and Tucker-2 decomposition-based CNN compression experiments were performed to verify the single- and multi-rank selection performance of the proposed method, respectively. In addition, to confirm the generalization performance of our proposed method, we used five representative CNNs, namely AlexNet, VGG-16, ResNet18, GoogLeNet, and SqueezeNet, which have different structural properties. The experiments are conducted on three representative datasets; ImageNet [21], CIFAR-100 [34], and CIFAR-10 [34]. In this study, we evaluated the performance of rank selection in terms of top-1 accuracy, top-5 accuracy, computational complexity, and memory complexity in the decomposed CNNs. Note that, the computational complexity is expressed in floating point operations per second (FLOPS). In addition, to validate the experimental results with respect to statistical methods, we conducted all experiments five times and then acquired the averages and standard deviations.

The experiment consists of three steps: 1) rank selection via proposed method, 2) low-rank decomposition on convolutional layer, and 3) fine-tuning the low-rank decomposed CNNs to recover the accumulated accuracy loss.

A. IMPLEMENTATION DETAILS

CNNs and BayesOpt tools were actualized using PyTorch in a GPU implementation [26] and GPyOpt in a CPU implementation [27], respectively. Tensorly API was utilized for SVD and Tucker-2 decomposition constructions [28]. As previously noted, our CNNs model was pre-trained on ImageNet classification and downloaded from torchvision [29]. We performed all experiments on GTX 1080 8-GB GPU and an Intel core i7 CPU. In Algorithm 1, the number of initial space-filling experiments n_0 is 10; the number of maximum iterations is 30; and the threshold of the distance between two consecutive observed ranks F is fixed at 10^{-3}. To fairly compare our method and VBMF, we initialized the learning rate at 10^{-3} and reduced it by a factor of 10 every five epochs. In addition, we only decomposed convolutional layers to accelerate the convergence speed in fine-tuning.

B. SINGLE-RANK SELECTION OF SVD FOR CNN COMPRESSION

To evaluate the performance of the proposed method on single-rank selection, we performed experiments with four-way tensor convolution kernel decomposition using SVD. Here, the rank is chosen by the number of independent columns of reshaped convolution kernel, which has a $DS \times DT$ size. The target model is AlexNet with the window sizes of the convolution kernels being 11×11, 5×5, and 3×3. In image classification on ImageNet, AlexNet achieved 56.66% top-1 accuracy and 70.09% top-5 accuracy. In CIFAR-10 and CIFAR-100 experiments, we changed 11×11, 5×5 convolution kernels to 3×3 convolution kernels to fit the AlexNet's receptive field in size of input images.

1) EXPERIMENTS WITH ALEXNET

Table. 1 shows that the proposed method achieved 1.1 times higher compression rate on computational complexity than VBMF with -0.32% top-1 accuracy and -0.31% top-5 accuracy in ImageNet experiment. In contrast, the proposed method achieved 2.5 times lower compression rate on computational complexity than VBMF with $+0.62\%$ top-1 accuracy and $+0.13\%$ top-5 accuracy in CIFAR-100 experiment. From these results verify that the accuracy of the decomposed model is linear with its computational complexity; hence, these experiments cannot determine the superior method in a single-rank selection problem. If the experiments performed were single-rank tensor decomposition such as CP-decomposition [7], VBMF would be unusable because it is a matrix-based rank selection method,

TABLE 2. Performance comparison of Tucker-2 decomposition for CNN compression using VBMF and the proposed method.

Dataset	Model	Rank Selection	Top-1 Accuracy (%)	Top-5 Accuracy (%)	# of Parameters (Mega)	FLOPS (Giga)
ImageNet	VGG-16	Base	71.59	90.38	14.71 (×1.0)	30.73 (×1.0)
		Ours	**67.93±0.04 (-3.66)**	**88.60±0.03 (-1.78)**	2.25 (×6.54)	**5.06 (×6.07)**
		VBMF	67.36±0.06 (-4.23)	88.04±0.01 (-2.34)	**2.21 (×6.66)**	5.57 (×5.52)
CIFAR-100	VGG-16	Base	72.93	91.16	14.71 (×1.0)	0.63 (×1.0)
		Ours	**68.07±0.22 (-4.86)**	**89.20±0.10 (-1.96)**	2.13 (×6.91)	**0.11 (×5.73)**
		VBMF	58.45±0.31 (-14.48)	83.02±0.13 (-8.14)	**1.96 (×7.51)**	0.12 (×5.25)
CIFAR-10	VGG-16	Base	91.29	-	14.71 (×1.0)	0.63 (×1.0)
		Ours	**88.52±0.21 (-2.77)**	-	**1.64 (×8.97)**	**0.10 (×6.30)**
		VBMF	87.18±0.27 (-4.08)	-	3.67 (×4.01)	0.13 (×4.85)
ImageNet	ResNet18	Base	69.75	89.07	11.18 (×1.0)	3.64 (×1.0)
		Ours	**63.53±0.11 (-6.22)**	**85.33±0.01 (-3.74)**	**2.12 (×5.27)**	**0.74 (×4.92)**
		VBMF	62.49±0.07 (-7.26)	84.50±0.03 (-4.57)	2.31 (×4.84)	0.81 (×4.49)
CIFAR-100	ResNet18	Base	76.14	92.70	11.18 (×1.0)	1.11 (×1.0)
		Ours	**72.71±0.14 (-3.43)**	**92.02±0.05 (-0.68)**	**1.86 (×6.01)**	**0.22 (×5.05)**
		VBMF	70.96±0.22 (-5.18)	91.32±0.19 (-1.38)	1.90 (×5.88)	0.34 (×3.26)
CIFAR-10	ResNet18	Base	93.57	-	11.18 (×1.0)	1.11 (×1.0)
		Ours	**91.48±0.11 (-2.09)**	-	**1.65 (×6.78)**	**0.20 (×5.55)**
		VBMF	88.43±0.26 (-5.14)	-	2.55 (×6.37)	0.24 (×4.63)
ImageNet	GoogLeNet	Base	69.77	89.53	5.60 (×1.0)	3.01 (×1.0)
		Ours	**67.95±0.07 (-1.82)**	**88.34±0.05 (-1.19)**	2.94 (×1.90)	**1.18 (×2.55)**
		VBMF	66.68±0.05 (-3.09)	87.74±0.03 (-1.79)	**2.90 (×1.93)**	1.22 (×2.47)
CIFAR-100	GoogLeNet	Base	74.50	92.72	6.16 (×1.0)	4.08 (×1.0)
		Ours	**74.47±0.12 (-0.03)**	**93.16±0.11 (+0.44)**	**3.35 (×1.84)**	**2.07 (×1.97)**
		VBMF	72.05±0.07 (-2.45)	91.66±0.09 (-1.06)	3.74 (×1.65)	2.33 (×1.75)
CIFAR-10	GoogLeNet	Base	93.56	-	6.16 (×1.0)	4.08 (×1.0)
		Ours	**93.27±0.14 (-0.29)**	-	**3.27 (×1.88)**	**1.51 (×2.70)**
		VBMF	93.14±0.12 (-0.42)	-	3.71 (×1.66)	1.67 (×2.44)
ImageNet	SqueezeNet	Base	58.09	80.42	0.74 (×1.0)	1.48 (×1.0)
		Ours	**55.39±0.02 (-2.70)**	**78.18±0.06 (-2.24)**	**0.30 (×2.47)**	**0.59 (×2.51)**
		VBMF	53.85±0.04 (-4.24)	76.83±0.03 (-3.59)	**0.30 (×2.47)**	0.62 (×2.39)
CIFAR-100	SqueezeNet	Base	69.71	91.23	0.74 (×1.0)	0.11 (×1.0)
		Ours	**67.58±0.30 (-2.13)**	**90.57±0.10 (-0.66)**	**0.34 (×2.18)**	**0.05 (×2.20)**
		VBMF	60.69±0.24 (-9.02)	86.96±0.16 (-4.27)	0.35 (×2.11)	**0.05 (×1.38)**
CIFAR-10	SqueezeNet	Base	91.04	-	0.74 (×1.0)	0.11 (×1.0)
		Ours	**88.23±0.23 (-2.81)**	-	**0.33 (×2.24)**	**0.04 (×2.75)**
		VBMF	87.80±0.16 (-3.24)	-	**0.33 (×2.24)**	**0.04 (×2.38)**

whereas the proposed method would be usable because it only takes rank value as an input.

C. MULTI-RANK SELECTION OF TUCKER-2 DECOMPOSITION FOR CNN COMPRESSION

In Tucker-2 decomposition of the convolution kernel, $\mathbf{r}(1)$ and $\mathbf{r}(2)$ are indicated by the number of independent columns of mode-3 matricization (size of $S \times DDT$) and mode-4 matricization (size of $T \times DDS$) of the convolution kernel, respectively. To obtain the Tucker-2 rank using VBMF, [7] entered both mode-3 and mode-4 matricization of the convolution kernel into VBMF, which only accepts data in a matrix format. When calculating the Tucker-2 rank via VBMF, each rank is acquired individually without considering the correlation between $\mathbf{r}(1)$ and $\mathbf{r}(2)$. In contrast, the proposed method considers not only the trade-off between reconstruction loss and computational complexity loss, but also the dependencies between the two ranks.

Four representative pretrained CNN models with different structural characteristics were utilized in our experiments to show that the proposed method can produce global optimal Tucker-2 rank independent of the CNN structure. The VGG-16 model achieved 71.592% top-1 accuracy and 90.382% top-5 accuracy; consisting of only 3×3 convolutional layers, it has the simplest structure among all models [17]. The second model, ResNet18, achieved 69.758% top-1 accuracy and 89.076% top-5 accuracy. It uses skip-connection, which provides a new solution for the

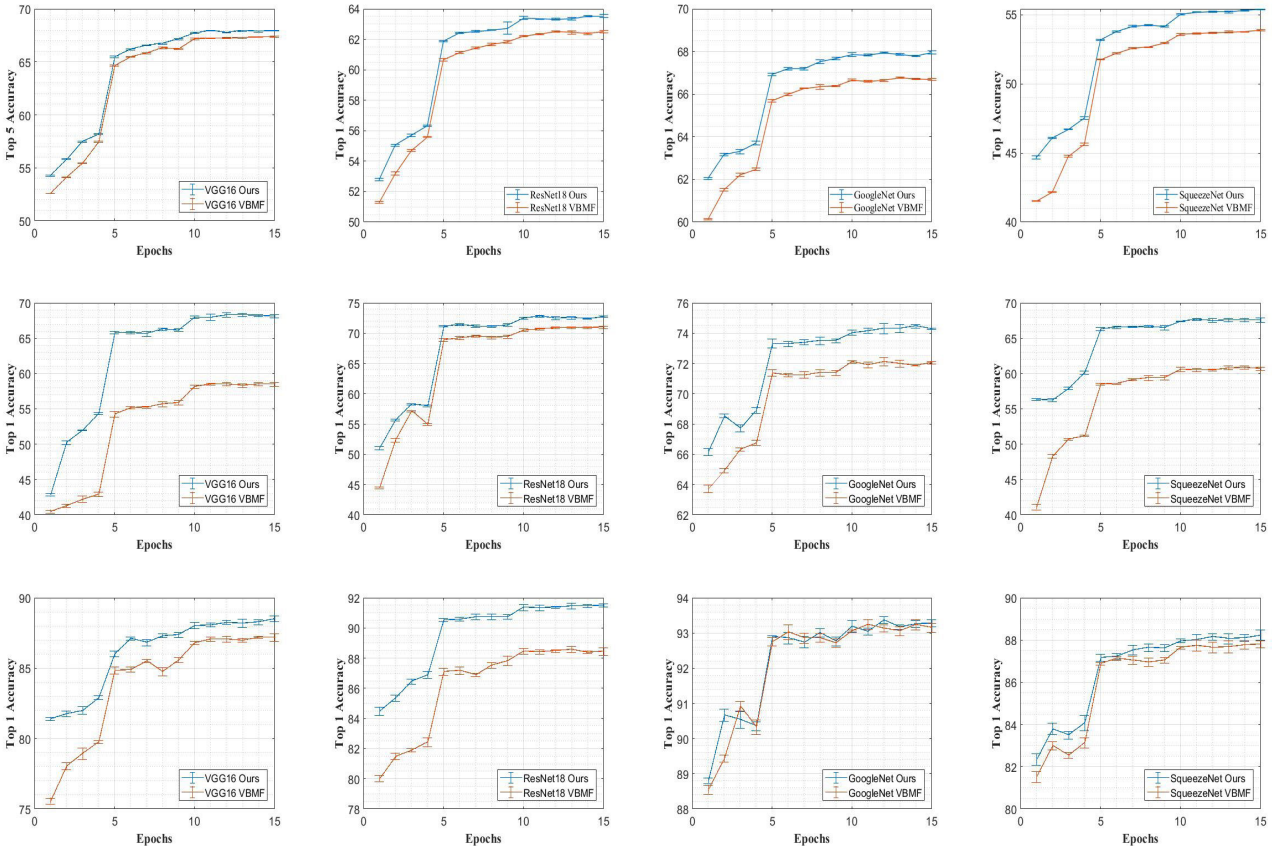


FIGURE 4. Top-1 Accuracy of low-rank decomposed CNNs using Tucker-2 decomposition in fine-tuning. The x- and y-axes of all sub-figures are finetuning epochs and top-1 accuracy, respectively. first row ImageNet, second row CIFAR-100, third row CIFAR-10, first column VGG-16, second column ResNet18, third column GoogLeNet, and fourth column SqueezeNet.

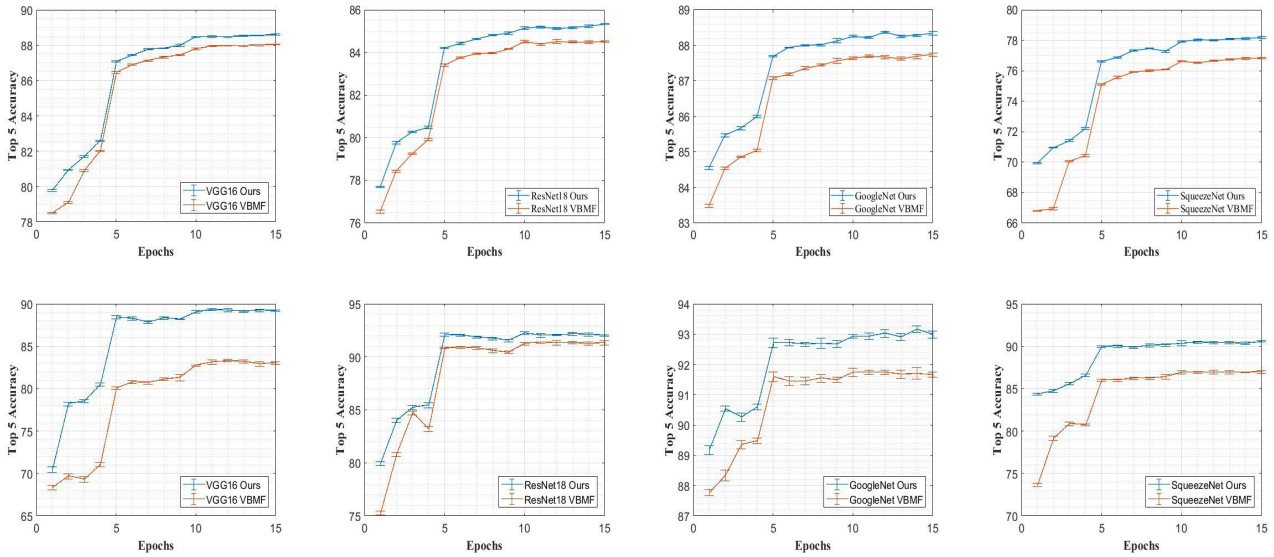


FIGURE 5. Top-5 Accuracy of low-rank decomposed CNNs using Tucker-2 decomposition in fine-tuning. The x- and y-axes of all sub-figures are finetuning epochs and top-5 accuracy, respectively. first row ImageNet, second row CIFAR-100, first column VGG-16, second column ResNet18, third column GoogLeNet, and fourth column SqueezeNet.

gradient vanishing problem [18]. Adopting an inception module, which enables parallel processing of various window size convolution filters in specific convolutional layer [19], GoogLeNet achieved 69.778% top-1 accuracy and 89.530% top-5 accuracy. Finally, the SqueezeNet model achieved 58.092% top-1 accuracy and 80.422% top-5 accuracy. It has the CNNs baseline and been developed for resource-constrained devices. Thus, this model's building block is called the fire module, which has dimension reduction effects [20].

TABLE 3. Performance comparison of Tucker-2 decomposition for CNN compression using the proposed method with a computational complexity rectifier for various α. In particular, $c_t(\mathbf{r}_{\mathbf{VBMF}})$ denotes the proposed method with α as the layerwise computational complexity loss of VBMF.

Dataset	Model		0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	$c_t(\mathbf{r}_{\mathbf{VBMF}})$
ImageNet	VGG16	Top-1 Accuracy (%)	66.45±0.07	68.19±0.03	69.00±0.08	70.39±0.06	70.61±0.05	70.77±0.03	71.04±0.02	71.03±0.06	71.15±0.01	68.08±0.05
		Top-5 Accuracy (%)	87.64±0.04	88.50±0.03	89.12±0.03	89.80±0.05	89.94±0.05	90.03±0.02	90.16±0.04	90.16±0.04	90.31±0.03	88.48±0.03
		# of Parameters (Mega)	1.42	2.89	4.16	5.69	7.17	8.50	9.49	11.03	12.49	2.17
		FLOPS (Giga)	3.19	6.05	8.81	11.77	15.01	17.73	19.93	23.53	26.2	5.46
ImageNet	ResNet18	Top-1 Accuracy (%)	55.00±0.04	60.51±0.03	65.59±0.06	67.75±0.06	68.24±0.02	68.71±0.08	68.79±0.02	69.00±0.05	69.41±0.05	63.27±0.02
		Top-5 Accuracy (%)	79.54±0.05	83.47±0.04	86.71±0.04	88.01±0.04	88.38±0.04	88.57±0.03	88.83±0.03	88.83±0.04	89.11±0.04	85.12±0.03
		# of Parameters (Mega)	1.22	2.33	3.35	4.43	5.51	6.58	7.12	8.85	9.55	2.18
		FLOPS (Giga)	0.4	0.77	1.08	1.47	1.83	2.16	2.40	2.89	3.18	0.77
ImageNet	GoogLeNet	Top-1 Accuracy (%)	63.24±0.07	65.32±0.07	69.36±0.03	70.34±0.03	70.64±0.03	70.85±0.07	71.14±0.05	71.36±0.04	71.27±0.03	67.68±0.03
		Top-5 Accuracy (%)	85.41±0.04	86.79±0.03	89.27±0.02	89.84±0.01	90.14±0.02	90.24±0.04	90.36±0.03	90.50±0.02	90.47±0.03	88.31±0.05
		# of Parameters (Mega)	2.84	2.90	3.25	3.53	3.87	4.22	4.54	4.88	5.18	2.87
		FLOPS (Giga)	1.07	1.18	1.41	1.63	1.85	2.08	2.28	2.50	2.72	1.18
ImageNet	SqueezeNet	Top-1 Accuracy (%)	52.91±0.03	54.33±0.07	56.12±0.07	56.83±0.03	57.29±0.06	57.49±0.01	57.94±0.07	57.97±0.05	58.15±0.03	55.43±0.04
		Top-5 Accuracy (%)	76.15±0.06	77.34±0.03	78.72±0.02	79.38±0.02	79.88±0.05	79.99±0.04	80.24±0.04	80.37±0.05	80.49±0.01	78.25±0.04
		# of Parameters (Mega)	0.26	0.27	0.33	0.38	0.45	0.48	0.55	0.60	0.64	0.30
		FLOPS (Giga)	0.51	0.55	0.62	0.73	0.85	0.94	1.08	1.18	1.29	0.59
CIFAR-100	VGG16	Top-1 Accuracy (%)	63.44±0.26	64.08±0.22	66.24±0.23	66.31±0.31	66.96±0.16	66.35±0.13	66.18±0.22	65.90±0.16	66.29±0.13	67.93±0.18
		Top-5 Accuracy (%)	87.17±0.25	88.43±0.05	88.81±0.14	89.10±0.18	89.45±0.10	89.59±0.11	89.42±0.07	89.24±0.12	89.32±0.17	89.12±0.23
		# of Parameters (Mega)	1.37	2.90	4.21	5.81	7.01	8.42	9.45	10.96	12.65	1.99
		FLOPS (Giga)	0.08	0.12	0.18	0.25	0.30	0.36	0.42	0.49	0.54	0.12
CIFAR-100	ResNet-18	Top-1 Accuracy (%)	71.42±0.18	72.82±0.18	72.93±0.15	72.95±0.12	72.49±0.12	73.18±0.12	72.65±0.25	72.48±0.25	72.49±0.13	72.28±0.24
		Top-5 Accuracy (%)	91.51±0.07	91.83±0.18	92.56±0.07	92.71±0.11	92.66±0.11	92.87±0.05	92.70±0.14	92.80±0.14	92.92±0.11	92.19±0.11
		# of Parameters (Mega)	1.27	2.39	3.40	4.46	5.62	6.59	7.64	8.70	9.75	1.93
		FLOPS (Giga)	0.12	0.24	0.34	0.45	0.56	0.66	0.75	0.87	0.96	0.33
CIFAR-100	GoogLeNet	Top-1 Accuracy (%)	65.01±0.11	71.28±0.21	72.07±0.15	74.58±0.06	74.86±0.24	74.54±0.23	74.70±0.11	74.70±0.18	74.80±0.11	72.92±0.25
		Top-5 Accuracy (%)	87.66±0.18	91.26±0.17	91.84±0.18	93.04±0.17	93.19±0.09	93.05±0.11	93.12±0.12	93.18±0.10	93.20±0.06	92.34±0.09
		# of Parameters (Mega)	3.16	3.48	3.80	4.03	4.37	4.68	4.93	5.32	5.70	3.71
		FLOPS (Giga)	1.36	1.54	1.72	2.55	2.75	3.02	3.18	3.46	3.74	2.30
CIFAR-100	SqueezeNet	Top-1 Accuracy (%)	64.92±0.16	66.69±0.11	67.08±0.20	67.66±0.19	68.29±0.12	68.45±0.10	68.83±0.04	68.79±0.11	69.30±0.12	66.78±0.27
		Top-5 Accuracy (%)	89.33±0.06	90.32±0.15	90.47±0.09	90.83±0.09	91.22±0.07	91.31±0.04	90.92±0.08	91.18±0.09	91.37±0.05	90.28±0.24
		# of Parameters (Mega)	0.35	0.37	0.39	0.44	0.50	0.53	0.59	0.65	0.70	0.34
		FLOPS (Giga)	0.04	0.05	0.05	0.06	0.07	0.07	0.08	0.09	0.10	0.05

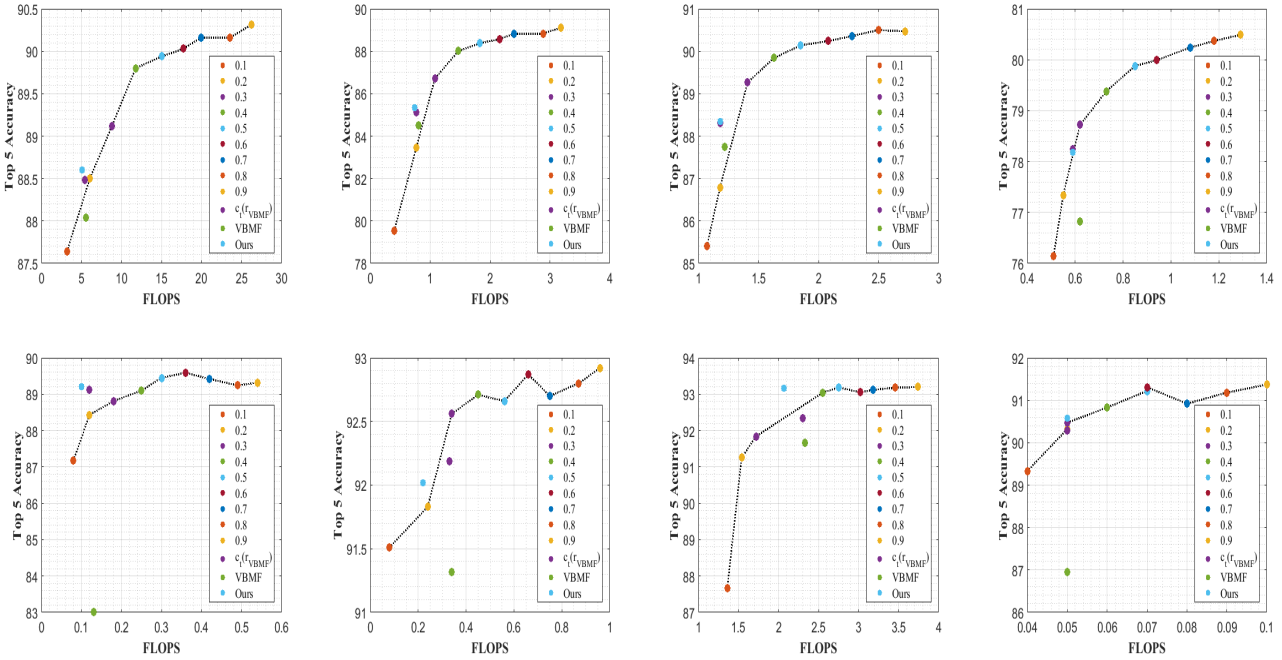


FIGURE 6. Top-5 accuracy-FLOPS curves of low-rank decomposed CNNs using proposed method with computational complexity rectifier on various α. $c_t(\mathbf{r}_{\mathbf{VBMF}})$ denotes the proposed method with the α as layerwise computational complexity loss of VBMF, VBMF denotes the performance of VBMF, and Ours means the performance of proposed method without computational complexity rectifier. The x- and y-axes of all sub-figures are FLOPS and top-5 accuracy, respectively. first row ImageNet, second row CIFAR-100, first column VGG-16, second column ResNet18, third column GoogLeNet, and fourth column SqueezeNet.

1) EXPERIMENTS WITH VGG-16, ResNet18, GoogLeNet, AND SqueezeNet

Table. 2 shows that the proposed method based decomposed CNN has both higher accuracy and lower computational complexity than a VBMF based decomposed CNN in all experimental results. Figs. 4 and 5 illustrate the top-1 and top-5 accuracy curves of the decomposed models using VBMF and the proposed method in every epoch, respectively.

TABLE 4. Layerwise analysis of Tucker-2 decomposed VGG-16 using VBMF and the proposed method with and without a computational complexity rectifier. $c_t(\mathbf{r}_{\mathbf{VBMF}})$ denotes the proposed method with α as the layerwise computational complexity loss of VBMF.

Layer	Kernel size	Channels	Filters	Output size	FLOPS (Giga)	VBMF			Proposed Method			Proposed Method ($\alpha = c_t(\mathbf{r}_{\mathbf{VBMF}})$)		
						$\mathbf{r}(1)$	$\mathbf{r}(2)$	FLOPS (Giga)	$\mathbf{r}(1)$	$\mathbf{r}(2)$	FLOPS (Giga)	$\mathbf{r}(1)$	$\mathbf{r}(2)$	FLOPS (Giga)
Conv1	3 ×3	3	64	224×224	0.176	2	14	0.119 (×1.48)	3	7	**0.068 (×2.59)**	3	11	0.104 (×1.68)
Conv2	3 ×3	64	64	224×224	3.702	29	26	1.037 (×3.56)	17	25	**0.656 (×5.63)**	21	35	1.026 (×3.6)
Maxpool1	2×2			112×112										
Conv3	3 ×3	64	128	112×112	1.851	32	39	0.46 (×4.02)	23	37	**0.349 (×5.29)**	27	42	0.435 (×4.24)
Conv4	3 ×3	128	128	112×112	3.7	43	42	0.682 (×5.42)	37	45	**0.640 (×5.77)**	30	58	0.677 (×5.46)
Maxpool2	2×2			56×56										
Conv5	3 ×3	128	256	56×56	1.85	59	71	0.398 (×4.64)	46	71	**0.336 (×5.5)**	54	75	0.393 (×4.7)
Conv6	3 ×3	256	256	56×56	3.7	81	78	0.612 (×6.03)	74	85	**0.611 (×6.05)**	69	91	0.612 (×6.04)
Conv7	3 ×3	256	256	56×56	3.7	73	70	0.518 (×7.13)	69	83	0.568 (×6.51)	67	73	**0.501 (×7.37)**
Maxpool3	2×2			28×28										
Conv8	3 ×3	256	512	28×28	1.85	110	131	0.353 (×5.23)	93	137	**0.327 (×5.64)**	92	144	0.339 (×5.44)
Conv9	3 ×3	512	512	28×28	3.699	148	138	0.518 (×7.13)	142	153	0.543 (×6.8)	153	132	**0.514 (×7.19)**
Conv10	3 ×3	512	512	28×28	3.699	132	121	0.428 (×8.62)	138	146	0.512 (×7.21)	122	129	**0.424 (×8.72)**
Maxpool4	2×2			14×14										
Conv11	3 ×3	512	512	14×14	0.924	157	144	0.14 (×6.59)	144	171	0.150 (×6.15)	138	161	**0.138 (×6.67)**
Conv12	3 ×3	512	512	14×14	0.924	149	147	0.136 (×6.76)	152	152	0.142 (×6.48)	147	147	**0.135 (×6.83)**
Conv13	3 ×3	512	512	14×14	0.924	155	152	0.144 (×6.38)	147	141	**0.131 (× 7.05)**	158	145	0.141 (×6.52)

From these observations, it can be confirmed that in every epoch, the decomposed CNNs using the proposed method achieved higher top-1 and top-5 accuracies than the model via VBMF. In addition, the proposed method provides a global optimal multi-rank irrespective of CNNs structure.

In multi-rank selection, the major reason the proposed method performs better than VBMF is that the proposed method calculates the multiple ranks simultaneously through constructing the multi-modal objective function. However, VBMF computes each rank independently. In other words, from the point of view of optimization theory, VBMF returns a locally optimal rank solution in the multi-rank selection problem. Thus, the experimental results on Tucker-2 rank selection, confirm that considering the correlation among ranks in multi-rank selection is an important key to good compression performance.

2) EXPERIMENTS WITH PROPOSED METHOD WITH TIME-COMPLEXITY RECTIFIER ON VARIOUS α

From Table. 3, it can be verified that using a computational complexity rectifier results in the compressed CNNs satisfying the restricted computational complexity condition and makes the proposed method flexible. As an example, the proposed method with $\alpha = 0.1$ means that all convolutional layers in original CNNs are decomposed with the same 0.1 computational complexity condition. Thus, the dotted curves in Fig. 6 can be considered as the approximated minimum boundary of the proposed method. All experiments in Fig. 6 show that VBMF produces the rank with the decomposed CNNs having higher computational complexity and lower accuracy than the proposed method. Therefore, it is verified that the proposed method produces the rank with more effective decomposed CNNs than VBMF because the former considers the correlation between ranks in multi-rank selection.

3) LAYERWISE ANALYSIS

Table. 4 shows that the proposed technique tends to produce $\mathbf{r}(2)$ larger than $\mathbf{r}(1)$ in all remaining layers, except for the last two layers, showing that it has used 1×1 convolution kernel for dimension expansion more than 1×1 convolution kernels used in the dimension reduction. This property is typical to CNNs design, which increases the depth feature map according to the deeper layer; this property is also seen in the

proposed method ($\alpha = c_t(\mathbf{r}_{\mathbf{VBMF}})$). Conversely, the VBMF independently selects $\mathbf{r}(1)$ and $\mathbf{r}(2)$, ignoring the correlation between the two ranks.

D. COMPUTATIONAL COMPELXITY ANALYSIS

The computational complexity of VBMF is denoted as $\mathcal{O}(6|\Omega|K)$. Here, $|\Omega|$ and K represent the whole number of indices in the given matrix and the rank, respectively [23], [41]. In contrast, the computational complexity of BayesOpt is defined as $\mathcal{O}(N^3)$, where N is the number of observed samples [24], [35]–[37]. The major cause of computational complexity is the inversion of a covariance matrix in GP regression [24]. Although, the computational complexity of the proposed method is higher than that of VBMF, several studies have investigated reducing the computational complexity of BayesOpt.

- A parallel processing based BayesOpt reduces the $\mathcal{O}(N^3)$ to $\mathcal{O}(N^2)$ [30]–[32].
- A sparse approximation based BayesOpt alleviate from $\mathcal{O}(N^3)$ to $\mathcal{O}(M^2N)$, where M is the size of subsamples for sparse approximation; $M \ll N$ [24], [38], [39].
- An exact inference based BayesOpt decreases the $\mathcal{O}(N^3)$ to $\mathcal{O}(M^{1+1/D})$, where D is the dimension of objective function, using a Cartesian product structure [40].

Therefore, there exist several possible ways to reduce the computational complexity of the proposed method using a proper BayesOpt acceleration algorithm. We leave the use of BayesOpt acceleration methods to reduce the computational complexity for future work.

V. CONCLUSION

In this paper, a novel global optimal rank selection method of low-rank decomposition for CNN compression based on BayesOpt was proposed. The proposed algorithm designs the objective function suitable for rank selection and optimizes it through BayesOpt to provide a global optimal rank. It has the following characteristics: 1) It represents the trade-off between computational complexity and accuracy degradation of decomposed CNNs according to the selected ranking. 2) It reflects the dependencies of each rank that constitutes a multi-rank, providing optimal rank in multi-rank selection. 3) With our proposed objective function, the combinatorial rank selection problem is considerably alleviated by BayesOpt. BayesOpt ensures that the proposed algorithm provides the global optimal rank because it performs uncertainty based optimization by utilizing the Gaussian process based on Màtern 5/2 covariance function and the expected improvement acquisition function. To demonstrate that the proposed algorithm is globally applicable on all rank selection problems, experiments using five representative CNNs with different structural features (AlexNet, VGG-16, ResNet18, GoogLeNet, and SqueezeNet) from various datasets (ImageNet, CIFAR-100, and CIFAR-10) were conducted. The experimental results showed that the proposed algorithm provides multi-rank with higher compression and higher performance than the state-of-the-art rank selection technique, VBMF.

REFERENCES

[1] J. Wang, K. Sun, T. Cheng, B. Jiang, C. Deng, Y. Zhao, D. Liu, Y. Mu, M. Tan, X. Wang, W. Liu, and B. Xiao, "Deep high-resolution representation learning for visual recognition," 2019, *arXiv:1908.07919*. [Online]. Available: https://arxiv.org/abs/1908.07919

[2] H. Touvron, A. Vedaldi, M. Douze, and H. Jégou, "Fixing the train-test resolution discrepancy," 2019, *arXiv:1906.06423*. [Online]. Available: https://arxiv.org/abs/1906.06423

[3] Y. Liu, Y. Wang, S. Wang, T. Liang, Q. Zhao, Z. Tang, and H. Ling, "CBNet: A novel composite backbone network architecture for object detection," 2019, *arXiv:1909.03625*. [Online]. Available: https://arxiv.org/abs/1909.03625

[4] Y. Song, and S. Ermon, "Generative modeling by estimating gradients of the data distribution," 2019, *arXiv:1907.05600*. [Online]. Available: https://arxiv.org/abs/1907.05600

[5] X. Xu, X. Zhou, R. Venkatesan, G. Swaminathan, and O. Majumder, "D-SNE: Domain adaptation using stochastic neighborhood embedding," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 2497–2506.

[6] M. Jaderberg, A. Vedaldi, and A. Zisserman, "Speeding up convolutional neural networks with low rank expansions," in *Proc. Brit. Mach. Vis. Conf.*, 2014, pp. 1–13.

[7] Y.-D. Kim, E. Park, S. Yoo, T. Choi, L. Yang, and D. Shin, "Compression of deep convolutional neural networks for fast and low power mobile applications," 2015, *arXiv:1511.06530*. [Online]. Available: https://arxiv.org/abs/1511.06530

[8] A. Novikov, D. Podoprikhin, A. Osokin, and D. P. Vetrov, "Tensorizing neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 442–450.

[9] T. Garipov, D. Podoprikhin, A. Novikov, and D. Vetrov, "Ultimate tensorization: Compressing convolutional and FC layers alike," 2016, *arXiv:1611.03214*. [Online]. Available: https://arxiv.org/abs/1611.03214

[10] P. Wang and J. Cheng, "Accelerating convolutional neural networks for mobile applications," in *Proc. ACM Multimedia Conf.-MM*, 2016, pp. 541–545.

[11] J. Cheng, P.-S. Wang, G. Li, Q.-H. Hu, and H.-Q. Lu, "Recent advances in efficient computation of deep convolutional neural networks," *Frontiers Inf. Technol. Electron. Eng.*, vol. 19, no. 1, pp. 64–77, Jan. 2018.

[12] D. R. Jones, M. Schonlau, and W. J. Welch, "Efficient global optimization of expensive black-box functions," *J. Global Optim.*, vol. 13, no. 4, pp. 455–492, 1998.

[13] J. Mockus, V. Tiesis, and A. Zilinskas, "The application of Bayesian methods for seeking the extremum," *Towards Global Optim.*, vol. 2, pp. 117–129, Dec. 1978.

[14] J. Snoek, H. Larochelle, R. P. Adams, "Practical Bayesian optimization of machine learning algorithms," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 2951–2959.

[15] P. I. Frazier, "A tutorial on Bayesian optimization," 2018, *arXiv:1807.02811*. [Online]. Available: https://arxiv.org/abs/1807.02811

[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

[17] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*. [Online]. Available: https://arxiv.org/abs/1409.1556

[18] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[19] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1–9.

[20] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size," 2016, *arXiv:1602.07360*. [Online]. Available: https://arxiv.org/abs/1602.07360

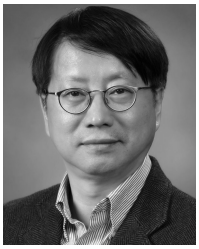
[21] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.

[22] A. Mnih and R. R. Salakhutdinov, "Probabilistic matrix factorization," in *Proc. Advances Neural Inf. Process. Syst.*, 2008, pp. 1257–1264.

[23] S. Nakajima, M. Sugiyama, S. D. Babacan, and R. Tomioka, "Global analytic solution of fully-observed variational Bayesian matrix factorization," *J. Mach. Learn. Res.* vol. 14, pp. 1–37, Jan. 2013.
[24] C. E. Rasmussen, "Gaussian processes in machine learning," in *Proc. Summer School Mach. Learn.* Berlin, Germany: Springer, 2003, pp. 63–71.
[25] J. Močkus, "On Bayesian methods for seeking the extremum," in *Proc. Optim. Techn. IFIP Tech. Conf.* Berlin, Germany: Springer, 1975.
[26] A. Paszke *et al.*, "Automatic differentiation in PyTorch," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS) Workshop*, 2017, pp. 1–4.
[27] J. Gonzalez and Z. Dai, *GPyOpt: A Bayesian Optimization Framework in Python*. Accessed: 2016. [Online]. Available: http://github.com/SheffieldML/GPyOpt
[28] J. Kossaifi, Y. Panagakis, A. Anandkumar, and M. Pantic, "Tensorly: Tensor learning in python," *J. Mach. Learn. Res.* vol. 20, no. 1, pp. 925–930, 2019.
[29] S. Marcel and Y. Rodriguez, "Torchvision the machine-vision package of torch," in *Proc. Int. Conf. Multimedia-MM*, 2010, pp. 1485–1488.
[30] J. Očenášek and J. Schwarz, "The parallel Bayesian optimization algorithm," in *The State of the Art in Computational Intelligence*. Heidelberg, Germany: Physica, 2000, pp. 61–67.
[31] A. Munawar, M. Wahib, M. Munetomo, and K. Akama, "Theoretical and empirical analysis of a GPU based parallel Bayesian optimization algorithm," in *Proc. Int. Conf. Parallel Distrib. Comput., Appl. Technol.*, Dec. 2009, pp. 457–462.
[32] K. Kandasamy, A. Krishnamurthy, J. Schneider, and B. Poczos, "Parallelised Bayesian optimisation via thompson sampling," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2018, pp. 133–142.
[33] C. E. Clark, "The greatest of a finite set of random variables," *Oper. Res.*, vol. 9, no. 2, pp. 145–162, Apr. 1961.
[34] A. Krizhevsky, "Learning multiple layers of features from tiny images," M.S. thesis, Dept. Comput. Sci., Univ. Toronto, Toronto, ON, Canada, 2009.
[35] C. Qin, D. Klabjan, D. Russo, "Improving the expected improvement algorithm," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5381–5391.
[36] I. O. Ryzhov, "On the Convergence Rates of Expected Improvement Methods," *Oper. Res.*, vol. 64, no. 6, pp. 1515–1528, Dec. 2016.
[37] M. Pelikan, D. E. Goldberg, and E. Cantú-Paz, "BOA: The Bayesian optimization algorithm," in *Proc. 1st Annu. Conf. Genetic Evol. Comput.*, vol. 1. San Mateo, CA, USA: Morgan Kaufmann, 1999.
[38] C. E. Rasmussen and Z. Ghahramani, "Infinite mixtures of Gaussian process experts," in *Proc. Adv. Neural Inf. Process. Syst.*, 2002, pp. 881–888.
[39] J. Quiñonero-Candela and C. E. Rasmussen, "A unifying view of sparse approximate Gaussian process regression," *J. Mach. Learn. Res.*, vol. 6, pp. 1939–1959, Dec. 2005.
[40] M. Belyaev, E. Burnaev, and Y. Kapushev, "Exact inference for Gaussian process regression in case of big data with the Cartesian product structure," 2014, *arXiv:1403.6573*. [Online]. Available: https://arxiv.org/abs/1403.6573
[41] Y.-D. Kim and S. Choi, "Scalable variational Bayesian matrix factorization with side information," in *Proc. Artif. Intell. Statist.*, 2014, pp. 493–502.
[42] M. Astrid, S.-I. Lee, and B.-S. Seo, "Rank selection of CP-decomposed convolutional layers with variational Bayesian matrix factorization," in *Proc. 20th Int. Conf. Adv. Commun. Technol. (ICACT)*, Feb. 2018, pp. 347–350.
[43] A. Saha, K. S. Ram, J. Mukhopadhyay, P. P. Das, and A. Patra, "Fitness based layer rank selection algorithm for accelerating CNNs by candecomp/parafac (CP) decompositions," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2019, pp. 3402–3406.

TAEHYEON KIM received the B.S. degree in electronics from Kangwon National University, Chuncheon, South Korea, in 2017. He is currently pursuing the Ph.D. degree in electrical and electronic engineering from Yonsei University, Seoul, South Korea. His research interests include neural network compression, tensor decomposition, and Bayesian optimization.

JIEUN LEE received the M.S. degree in electrical and electronic engineering from Yonsei University, Seoul, South Korea, in 2015, where she is currently pursuing the Ph.D. degree in electrical and electronic engineering. Her current research interests include low-rank approximation and nonlinear dimensionality reduction.



YOONSIK CHOE (Senior Member, IEEE) received the B.S. degree in electrical engineering from Yonsei University, Seoul, South Korea, in 1979, the M.S.E.E. degree in systems engineering from Case Western Reserve University, Cleveland, OH, USA, in 1984, the M.S. degree in electrical engineering from Pennsylvania State University, State College, PA, USA, in 1987, and the Ph.D. degree in electrical engineering from Purdue University, West Lafayette, IN, USA, in 1990. From 1990 to 1993, he was a Principal Research Staff with the Industrial Electronics Research Center, Hyundai Electronics Company, Ltd. Since 1993, he has been with the Department of Electrical and Electronic Engineering, Yonsei University, where he is currently a Professor. His research interests include video coding, video communication, statistical signal processing, and digital image processing.