RETHINKING COMPRESSED CONVOLUTIONAL NEURAL NETWORKS FROM A STATISTICAL PERSPECTIVE

Anonymous authors
Paper under double-blind review

ABSTRACT

Many designs have recently been proposed to improve the model efficiency of convolutional neural networks (CNNs) at a fixed resource budget, while there is a lack of theoretical analysis to justify them. This paper first formulates CNNs with high-order inputs into statistical models, which have a special "Tucker-like" formulation. This makes it possible to further conduct the sample complexity analysis to CNNs as well as compressed CNNs via tensor decomposition. Tucker and CP decompositions are commonly adopted to compress CNNs in the literature. The low rank assumption is usually imposed on the output channels, which according to our study, may not be beneficial to obtain a computationally efficient model while a similar accuracy can be maintained. Our finding is further supported by ablation studies on CIFAR10, SVNH and UCF101 datasets.

1 INTRODUCTION

The introduction of AlexNet (Krizhevsky et al., 2012) spurred a line of research in 2D CNNs, which progressively achieve high levels of accuracy in the domain of image recognition (Simonyan & Zisserman, 2015; Szegedy et al., 2015; He et al., 2016; Huang et al., 2017). The current state-of-the-art CNNs leave little room to achieve significant improvement on accuracy in learning still-images, and attention has hence been diverted towards two directions. The first is to deploy deep CNNs on mobile devices by removing redundancy from the over-parametrized network, and some representative models include MobileNetV1 & V2 (Howard et al., 2017; Sandler et al., 2018).

The second direction is to utilize CNNs to learn from higher-order inputs, for instance, video clips (Tran et al., 2018; Hara et al., 2017) or electronic health records (Cheng et al., 2016; Suo et al., 2017). This area has not yet seen a widely-accepted state-of-the-art network. High-order kernel tensors are usually required to account for the multiway dependence of the input. This notoriously leads to heavy computational burden, as the number of parameters to be trained grows exponentially with the dimension of inputs. Subsequently, model compression becomes the critical juncture to guarantee the successful training and deployment of tensor CNNs.

This paper is motivated by CNNs with high-order inputs and focuses on the statistical analysis to better understand CNNs and compressed CNNs via tensor decomposition. It makes three main contributions:

1. We formulate CNNs with high-order inputs into statistical models, which have an explicit "Tucker-like" form.

2. The sample complexity analysis can then be conducted to CNNs as well as compressed CNNs via tensor decomposition, with weak conditions allowing for time-dependent inputs like video data.

3. From theoretical analysis, we draw an interesting finding that forcing low dimensionality on output channels may introduce unnecessary parameter redundancy to a compressed network.

1.1 RELATED WORKS

Tensor decomposition for convolution kernels. Denil et al. (2013) showed that there is huge redundancy in network weights such that the entire network can be approximately recovered with a small fraction of parameters. To exploit the implicit low dimensionality in convolution kernels and

to achieve a speedup in the training process, tensor decomposition has been adopted to compress 2D CNNs. Denton et al. (2014) used a two-step approach, by first clustering the subtensors of a convolution kernel that share similar structures, and then applying CP decomposition to approximate each cluster. Lebedev et al. (2015) directly applied CP decomposition to the whole convolution kernel tensor. However, the instability of low-rank CP decomposition renders it difficult to be applied to the entire network. Astrid & Lee (2017) attempted to resolve the issue via iterative fine-tuning. Kim et al. (2016), on the other hand, proposed Tucker decomposition for convolution layers, which is more stable than CP. In a most recent work by Kossaifi et al. (2020), CP decomposition is extended to factorize higher-order convolution kernels for 3D video learning problems.

For deep CNNs, various compressed block structures have been proposed to increase depths while limiting the number of newly added parameters, to ensure computational efficiency. Some prominent examples include the bottleneck block in ResNet (He et al., 2016), the depthwise separable block in MobileNetV1 (Howard et al., 2017) and the inverted residual block in MobileNetV2 (Sandler et al., 2018). These seemingly standalone block designs can, in fact, be categorized as special cases of Tucker or CP decomposition on the convolution kernel tensor (Kossaifi et al., 2020; Hayashi et al., 2019). We are hence able to apply our theoretical findings for tensor decomposed CNNs to analyze these popular block designs.

Sample complexity analysis. One longstanding interest in deep learning is to understand the number of effective parameters in neural networks, i.e. how many training samples are needed. Many focus on the convergence analysis for specific algorithms (Zhong et al., 2017; Goel et al., 2018; Du & Goel, 2018; Cao & Gu, 2019; Fu et al., 2020). Some extend to the overparametrized regime with certain weight constraints and derive the model agnostic generalization bounds (Neyshabur et al., 2015; Bartlett et al., 2017; Golowich et al., 2018; Arora et al., 2018; Suzuki et al., 2018; Bartlett et al., 2019; Li et al., 2018). Our paper differs from these works in that we are algorithm-free and focus only on the statistical sample complexity of a CNN, akin to Du et al. (2018).

Du et al. (2018) first characterized the precise sample complexity of a CNN. Our paper differs from theirs in some key aspects. They adopted a 1-dimensional convolution with a single kernel, followed by weighted summation. In comparison, we present a more realistic modeling of CNN by introducing a general N-dimensional convolution with multiple kernels, followed by an average pooling layer and a fully-connected layer. And we further show that it is possible to include more "convolution-pooling" layers into our model. To accommodate video learning problems, we relax the technical assumptions and allow for general time-dependent inputs. Under the case of a single kernel and no pooling, our derived bound for the tensor CNN exactly matches the information-theoretic lower bound in Du et al. (2018), suggesting that our sample complexity analysis is sharp. Moreover, we extend our sample complexity analysis to compressed CNNs via tensor decomposition, while Du et al. (2018) cannot be applied.

2 Formulating CNNs with higher-order inputs

2.1 Notation

Tensor notations. We follow the notations in Kolda & Bader (2009) to denote vectors by lowercase boldface letters, e.g. $\boldsymbol{a}$; matrices by capital boldface letters, e.g. $\boldsymbol{A}$; tensors of order 3 or higher by Euler script boldface letters, e.g. $\boldsymbol{\mathcal{A}}$. For an Nth-order tensor $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{l_1 \times \cdots \times l_N}$, denote its elements by $\boldsymbol{\mathcal{A}}(i_1, i_2, \ldots, i_N)$ and the n-mode unfolding by $\boldsymbol{\mathcal{A}}_{(n)}$, where the columns of $\boldsymbol{\mathcal{A}}_{(n)}$ are the n-mode vectors of $\boldsymbol{\mathcal{A}}$, for $1 \leq n \leq N$. And $\boldsymbol{\mathcal{A}}(:, \cdots, i_n, \cdots, :)$ denotes the subtensor of $\boldsymbol{\mathcal{A}}$, holding only the nth index fixed. The vectorization operation is denoted by $\text{vec}(\cdot)$. The inner product of two tensors $\boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{B}} \in \mathbb{R}^{l_1 \times \cdots \times l_N}$ is defined as $\langle \boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{B}} \rangle = \sum_{i_1} \cdots \sum_{i_N} \boldsymbol{\mathcal{A}}(i_1, \ldots, i_N) \boldsymbol{\mathcal{B}}(i_1, \ldots, i_N)$, and the Frobenius norm is $\|\boldsymbol{\mathcal{A}}\|_{\text{F}} = \sqrt{\langle \boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{A}} \rangle}$. The mode-$n$ multiplication $\times_n$ of a tensor $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{l_1 \times \cdots \times l_N}$ and a matrix $\boldsymbol{B} \in \mathbb{R}^{p_n \times l_n}$ is defined as $(\boldsymbol{\mathcal{A}} \times_n \boldsymbol{B})(i_1, \ldots, j_n, \ldots, i_N) = \sum_{i_n=1}^{l_n} \boldsymbol{\mathcal{A}}(i_1, \ldots, i_n, \ldots, i_N) \boldsymbol{B}(j_n, i_n)$, for $1 \leq n \leq N$, respectively. The mode-n multiplication $\bar{\times}_N$ of a tensor $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{l_1 \times \cdots \times l_N}$ and a vector $\boldsymbol{b} \in \mathbb{R}^{l_n}$ is defined as $(\boldsymbol{\mathcal{A}} \bar{\times}_n \boldsymbol{b})(i_1, \ldots, i_{n-1}, i_{n+1}, \ldots, i_N) = \sum_{i_n=1}^{l_n} \boldsymbol{\mathcal{A}}(i_1, \ldots, i_n, \ldots, i_N) \boldsymbol{b}(i_n)$, for $1 \leq n \leq N$. The symbol "$\otimes$" is the Kronecker product and "$\circ$" is the outer product. We extend the definition of Khatri-Rao product to tensors: given tensors $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{l_1 \times l_2 \times \cdots l_N \times K}$ and $\boldsymbol{\mathcal{B}} \in \mathbb{R}^{p_1 \times p_2 \times \cdots p_N \times K}$,

their Khatri-Rao product is a tensor of size $\mathbb{R}^{l_1 p_1 \times l_2 p_2 \times \cdots l_N p_N \times K}$, denoted by $\mathcal{C} = \mathcal{A} \odot \mathcal{B}$, where $\mathcal{C}(:, \cdots, k) = \mathcal{A}(:, \cdots, k) \otimes \mathcal{B}(:, \cdots, k), 1 \leq k \leq K$.

CP decomposition. The Canonical Polyadic (CP) decomposition (Kolda & Bader, 2009) factorizes the tensor $\mathcal{A} \in \mathbb{R}^{l_1 \times \cdots \times l_N}$ into a sum of rank-1 tensors, i.e. $\mathcal{A} = \sum_{r=1}^{R} \alpha_r \boldsymbol{h}_r^{(1)} \circ \boldsymbol{h}_r^{(2)} \circ \cdots \circ \boldsymbol{h}_r^{(N)}$, where $\boldsymbol{h}_r^{(j)}$ is a unit norm vector of size $\mathbb{R}^{l_j}$ for all $1 \leq j \leq N$. The CP rank is the number of rank-1 tensors R.

Tucker decomposition. The Tucker ranks of an Nth-order tensor $\mathcal{A} \in \mathbb{R}^{l_1 \times \cdots \times l_N}$ are defined as the matrix ranks of the unfoldings of $\mathcal{A}$ along all modes. If the Tucker ranks of $\mathcal{A}$ are $(R_1, \ldots, R_N)$, then there exist a core tensor $\mathcal{G} \in \mathbb{R}^{R_1 \times \cdots \times R_N}$ and matrices $\boldsymbol{H}^{(i)} \in \mathbb{R}^{l_i \times R_i}$, such that $\mathcal{A} = \mathcal{G} \times_1 \boldsymbol{H}^{(1)} \times_2 \boldsymbol{H}^{(2)} \cdots \times_N \boldsymbol{H}^{(N)}$, known as Tucker decomposition (Tucker, 1966),

2.2 Basic three-layer CNNs

Consider a three-layer CNN with one convolution, one average pooling and one fully-connected layers, and we assume linear activations for simplicity. Specifically, for a general tensor-structured input $\mathcal{X} \in \mathbb{R}^{d_1 \times d_2 \times \cdots \times d_N}$, we first perform its convolution with an Nth-order kernel tensor $\mathcal{A} \in \mathbb{R}^{l_1 \times l_2 \times \cdots \times l_N}$ to get an intermediate output $\mathcal{X}_c \in \mathbb{R}^{m_1 \times m_2 \times \cdots \times m_N}$, and then use average pooling with pooling sizes $(q_1, \cdots, q_N)$ to derive another intermediate output $\mathcal{X}_{cp} \in \mathbb{R}^{p_1 \times p_2 \times \cdots \times p_N}$. Finally, $\mathcal{X}_{cp}$ goes through a fully-connected layer, with weight tensor $\mathcal{B} \in \mathbb{R}^{p_1 \times p_2 \times \cdots \times p_N}$ to produce a scalar output; see Figure 1(1).

We consider the convolution layer with the stride size of s_c along each dimension. Assume that $m_j = (d_j - l_j)/s_c + 1$ are integers for $1 \leq j \leq N$, otherwise zero-padding will be needed. Let

$$\boldsymbol{U}_{i_j}^{(j)} = [\underbrace{\boldsymbol{0}}_{(i_j-1)s_c} \underbrace{\boldsymbol{I}}_{l_j} \underbrace{\boldsymbol{0}}_{d_j-(i_j-1)s_c-l_j}]' \in \mathbb{R}^{d_j \times l_j} \text{ for } 1 \leq i_j \leq m_j,\ 1 \leq j \leq N, \tag{1}$$

which act as positioning factors to stretch the kernel tensor $\mathcal{A}$ into a tensor of the same size as $\mathcal{X}$, while the rest of entries are filled with zeros; see Figure 1(2). As a result, $\mathcal{X}_c$ has entries

$$\mathcal{X}_c(i_1, i_2, \ldots, i_N) = \langle \mathcal{X}, \mathcal{A} \times_1 \boldsymbol{U}_{i_1}^{(1)} \times_2 \boldsymbol{U}_{i_2}^{(2)} \times_3 \cdots \times_N \boldsymbol{U}_{i_N}^{(N)} \rangle.$$

For the pooling layer with the stride size of s_p along each dimension, we assume the pooling sizes $\{q_j\}_{j=1}^N$ satisfy the relationship of $m_j = q_j + (p_j - 1)s_p$, so the sliding windows can be overlapped. But for ease of notation, we can simply take $q_j = m_j/p_j$. The average pooling operation is equivalent to forming $p_1 p_2 p_3 \cdots p_N$ consecutive blocks within $\mathcal{X}_c$, each of size $\mathbb{R}^{q_1 \times q_2 \times \cdots \times q_N}$, and then take the average per block. The resulting tensor $\mathcal{X}_{cp}$ has entries $\mathcal{X}_{cp}(i_1, i_2, \ldots, i_N) = (\prod_j q_j)^{-1} \sum_j \sum_{k_j=(i_j-1)q_j+1}^{i_j q_j} \mathcal{X}_c(k_1, k_2, \ldots, k_N)$, for $1 \leq i_j \leq p_j$ and $1 \leq j \leq N$. If we denote $\boldsymbol{U}_{\mathcal{F},i_j}^{(j)} = q_j^{-1} \sum_{k=(i_j-1)q_j+1}^{i_j q_j} \boldsymbol{U}_k^{(j)}$, then equivalently,

$$\mathcal{X}_{cp}(i_1, i_2, \ldots, i_N) = \langle \mathcal{X}, \mathcal{A} \times_1 \boldsymbol{U}_{\mathcal{F},i_1}^{(1)} \times_2 \boldsymbol{U}_{\mathcal{F},i_2}^{(2)} \times_3 \cdots \times_N \boldsymbol{U}_{\mathcal{F},i_N}^{(N)} \rangle.$$

The fully-connected layer performs a weighted summation over $\mathcal{X}_{cp}$, with weights given by entries of the tensor $\mathcal{B} \in \mathbb{R}^{p_1 \times p_2 \times \cdots \times p_N}$. Denote $\boldsymbol{U}_{\mathcal{F}}^{(j)} = (\boldsymbol{U}_{\mathcal{F},1}^{(j)}, \cdots, \boldsymbol{U}_{\mathcal{F},p_j}^{(j)})$ for $1 \leq j \leq N$, and the predicted output has the form of

$$\widehat{y} = \langle \mathcal{X}_{cp}, \mathcal{B} \rangle = \langle \mathcal{X}, (\mathcal{B} \otimes \mathcal{A}) \times_1 \boldsymbol{U}_{\mathcal{F}}^{(1)} \times_2 \boldsymbol{U}_{\mathcal{F}}^{(2)} \times_3 \cdots \times_N \boldsymbol{U}_{\mathcal{F}}^{(N)} \rangle.$$

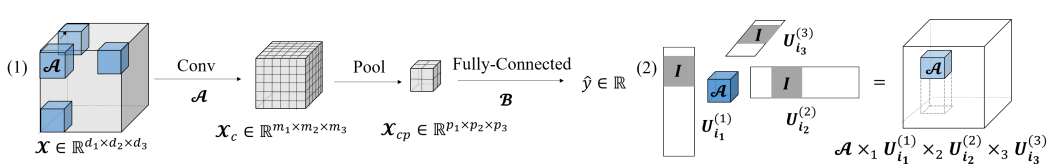

Figure 1: (1) 3-layer CNN for a 3D input $\mathcal{X}$ with one kernel tensor $\mathcal{A}$, average pooling, and the fully-connected weights $\mathcal{B}$. (2) $\boldsymbol{U}_{i_j}^{(j)}$s act as positioning factors. The white spaces indicate zero entries.

Similarly, for a CNN with K kernels, denote $\{\mathcal{A}_k, \mathcal{B}_k\}_{k=1}^K$ to be the set of kernels and fully-connected weights, where $\mathcal{B}_k \in \mathbb{R}^{p_1 \times p_2 \times \cdots \times p_N}$ and $\mathcal{A}_k \in \mathbb{R}^{l_1 \times l_2 \times \cdots \times l_N}$. The model can then be explicitly represented by

$$y^i = \widehat{y}^i + \xi^i = \langle \mathcal{X}^i, \mathcal{W}_X \rangle + \xi^i, \tag{2}$$

where ξ^i is the additive error, $\mathcal{X}^i \in \mathbb{R}^{d_1 \times d_2 \times \cdots \times d_N}$, and the composite weight tensor

$$\mathcal{W}_X = (\sum_{k=1}^K \mathcal{B}_k \otimes \mathcal{A}_k) \times_1 \boldsymbol{U}_{\mathcal{F}}^{(1)} \times_2 \boldsymbol{U}_{\mathcal{F}}^{(2)} \times_3 \cdots \times_N \boldsymbol{U}_{\mathcal{F}}^{(N)}. \tag{3}$$

The innate weight-sharing compactness of CNN can be equivalently represented as a "Tucker-like" form at (3). The factor matrices $\{\boldsymbol{U}_{\mathcal{F}}^{(j)} \in \mathbb{R}^{d_j \times l_j p_j}\}_{j=1}^N$ are fixed and solely determined by CNN operations on the inputs. They have full column ranks given that $p_j l_j \leq d_j$ always holds. The core tensor is a special Kronecker product that depicts the interaction between shared kernels and fully-connected weights.

2.3 Sample complexity analysis

We can now derive the non-asymptotic error bound for the CNN model. Let $\mathcal{Z}^i = \mathcal{X}^i \times_1 \boldsymbol{U}_{\mathcal{F}}^{(1)\prime} \times_2 \boldsymbol{U}_{\mathcal{F}}^{(2)\prime} \times_3 \cdots \times_N \boldsymbol{U}_{\mathcal{F}}^{(N)\prime} \in \mathbb{R}^{l_1 p_1 \times l_2 p_2 \times \cdots \times l_N p_N}$. Model (2) is equivalent to

$$y^i = \langle \mathcal{Z}^i, \mathcal{W} \rangle + \xi^i = \sum_{k=1}^K \langle \mathcal{Z}^i, \mathcal{B}_k \otimes \mathcal{A}_k \rangle + \xi^i, \tag{4}$$

where $\mathcal{W} = \sum_{k=1}^K \mathcal{B}_k \otimes \mathcal{A}_k$. The trained weights have the form of $\widehat{\mathcal{W}} = \sum_{k=1}^K \widehat{\mathcal{B}}_k \otimes \widehat{\mathcal{A}}_k$, where

$$\{\widehat{\mathcal{B}}_k, \widehat{\mathcal{A}}_k\}_{1 \leq k \leq K} = \underset{\mathcal{B}_k, \mathcal{A}_k, 1 \leq k \leq K}{\arg\min} \frac{1}{n} \sum_{i=1}^n \left(y^i - \sum_{k=1}^K \langle \mathcal{Z}^i, \mathcal{B}_k \otimes \mathcal{A}_k \rangle \right)^2. \tag{5}$$

Denote $\boldsymbol{x}^i = \mathrm{vec}(\mathcal{X}^i)$ and $\boldsymbol{z}^i = \mathrm{vec}(\mathcal{Z}^i)$, and it can be verified that $\boldsymbol{z}^i = \boldsymbol{U}_G' \boldsymbol{x}^i$, where $\boldsymbol{U}_G = \boldsymbol{U}_{\mathcal{F}}^{(1)} \otimes \{\boldsymbol{U}_{\mathcal{F}}^{(N)} \otimes [\boldsymbol{U}_{\mathcal{F}}^{(N-1)} \otimes \cdots \otimes (\boldsymbol{U}_{\mathcal{F}}^{(3)} \otimes \boldsymbol{U}_{\mathcal{F}}^{(2)})]\}$ represents the CNN operations on the inputs. Let $\bar{\boldsymbol{x}} = (\boldsymbol{x}^{1\prime}, \boldsymbol{x}^{2\prime}, \ldots, \boldsymbol{x}^{n\prime})'$, and we make the following technical assumptions.

Assumption 1. *(Time-dependent inputs) $\bar{\boldsymbol{x}}$ is normally distributed with mean zero and variance $\boldsymbol{\Sigma} = \mathbb{E}(\bar{\boldsymbol{x}}\bar{\boldsymbol{x}}')$, where $c_x \boldsymbol{I} \leq \boldsymbol{\Sigma} \leq C_x \boldsymbol{I}$ for some $0 < c_x < C_x$.*

Denote by $\lambda_{\max}(\boldsymbol{A})$ and $\lambda_{\min}(\boldsymbol{A})$ the maximum and minimum eigenvalues of a symmetric matrix $\boldsymbol{A}$, respectively. When $\{\boldsymbol{x}^i\}$ is a stationary time series with spectral density function $f_X(\theta)$, we can take $c_x = \inf_{-\pi \leq \theta \leq \pi} \lambda_{\min}(f_X(\theta))$ and $C_x = \sup_{-\pi \leq \theta \leq \pi} \lambda_{\max}(f_X(\theta))$; see Basu & Michailidis (2015). For independent inputs, $\boldsymbol{\Sigma} = \mathrm{diag}\{\boldsymbol{\Sigma}_{11}, \ldots, \boldsymbol{\Sigma}_{nn}\}$ is block diagonal, and it holds that $c_x = \min_{1 \leq j \leq n} \lambda_{\min}(\boldsymbol{\Sigma}_{jj})$ and $C_x = \max_{1 \leq j \leq n} \lambda_{\max}(\boldsymbol{\Sigma}_{jj})$, where $\boldsymbol{\Sigma}_{jj} = \mathbb{E}(\boldsymbol{x}^j \boldsymbol{x}^{j\prime})$.

Assumption 2. *(Sub-Gaussian errors) $\{\xi^i\}$ are independent σ^2-sub-Gaussian random variables with mean zero, and is independent of $\{\mathcal{X}^j, 1 \leq j \leq i\}$ for all $1 \leq i \leq n$.*

Assumption 3. *(Restricted isometry property) $c_u \boldsymbol{I} \leq \boldsymbol{U}_G' \boldsymbol{U}_G \leq C_u \boldsymbol{I}$ for some $0 < c_u < C_u$.*

Denote $\kappa_U = C_x C_u$ and $\kappa_L = c_x c_u$. Let $\mathcal{W}^* = \sum_{k=1}^K \mathcal{B}_k^* \otimes \mathcal{A}_k^*$ be the true weight. The model complexity of CNN at (4) is $d_{\mathcal{M}} = K(P + L + 1)$.

Theorem 1 (CNN)**.** *Suppose that Assumptions 1-3 hold and $n \gtrsim (\kappa_U/\kappa_L)^2 d_{\mathcal{M}}$. Then,*

$$\|\widehat{\mathcal{W}} - \mathcal{W}^*\|_{\mathrm{F}} \leq \frac{16\sigma\sqrt{\alpha_{\mathrm{RSM}}}}{\alpha_{\mathrm{RSC}}} \left(\sqrt{\frac{d_{\mathcal{M}}}{n}} + \sqrt{\frac{\delta}{n}} \right)$$

for some $\delta > 0$, with probability $1 - 4\exp\{-[c_{\mathrm{H}}(0.25\kappa_L/\kappa_U)^2 n - 9d_{\mathcal{M}}]\} - \exp\{-d_{\mathcal{M}} - 8\delta\}$, where c_{H} is a positive value, $\alpha_{\mathrm{RSC}} = \kappa_L/2$, and $\alpha_{\mathrm{RSM}} = 3\kappa_U/2$.

It can be seen that the estimation error, $\|\widehat{\mathcal{W}} - \mathcal{W}^*\|_{\mathrm{F}}$, is $O_p(\sqrt{d_{\mathcal{M}}/n})$, and hence the sample complexity is of order $O(d_{\mathcal{M}}/\varepsilon^2)$, to achieve an estimation error ε. Technical proofs of all theorems and corollary in the paper are deferred to Appendix A.3.

2.4 CNNs WITH MORE LAYERS

Consider a 5-layer CNN with "convolution → pooling → convolution → pooling → fully connected" layers, where the settings for the first "convolution → pooling" layers are the same as those in Section 2.2.

Denote the kernel for the second convolution by $\widetilde{\mathcal{A}} \in \mathbb{R}^{\tilde{l}_1 \times \tilde{l}_2 \times \cdots \times \tilde{l}_N}$, and the fully-connected weight tensor $\mathcal{B}$ is of size $\mathbb{R}^{\tilde{p}_1 \times \tilde{p}_2 \times \cdots \times \tilde{p}_N}$. We similarly define matrices $\widetilde{\boldsymbol{U}}_{i_j}^{(j)}$ and $\widetilde{\boldsymbol{U}}_{\mathcal{F},i_j}^{(j)}$, which are both of size $\mathbb{R}^{p_j \times \tilde{l}_j}$ with $1 \le i_j \le \tilde{p}_j$ and $1 \le j \le N$, and can be used to represent the second convolution and pooling operations, respectively. Note that the output from the first pooling layer, $\mathcal{X}_{cp}$, is the input to the second convolution layer, and hence the output from the second pooling layer has entries

$$\widetilde{\mathcal{X}}_{cp}(i_1, i_2, \ldots, i_N) = \langle \mathcal{X}_{cp}, \widetilde{\mathcal{A}} \times_1 \widetilde{\boldsymbol{U}}_{\mathcal{F},i_1}^{(1)} \times_2 \widetilde{\boldsymbol{U}}_{\mathcal{F},i_2}^{(2)} \times_3 \cdots \times_N \widetilde{\boldsymbol{U}}_{\mathcal{F},i_N}^{(N)} \rangle.$$

Stack the matrices $\{\boldsymbol{U}_{\mathcal{F},i_j}^{(j)} \in \mathbb{R}^{d_j \times l_j}\}_{1 \le i_j \le p_j}$ and $\{\widetilde{\boldsymbol{U}}_{\mathcal{F},i_j}^{(j)} \in \mathbb{R}^{p_j \times \tilde{l}_j}\}_{1 \le i_j \le \tilde{p}_j}$ into 3D tensors $\mathcal{U}^{(j)} \in \mathbb{R}^{d_j \times l_j \times p_j}$ and $\widetilde{\mathcal{U}}^{(j)} \in \mathbb{R}^{p_j \times \tilde{l}_j \times \tilde{p}_j}$, respectively. Let $\boldsymbol{U}_{\mathcal{DF}}^{(j)} = (\mathcal{U}^{(j)} \times_3 \widetilde{\mathcal{U}}^{(j)})_{(1)}$ for $1 \le j \le N$ and, by some algebra in Appendix A.2, we can show that the predicted output has the form of

$$\widehat{y} = \langle \widetilde{\mathcal{X}}_{cp}, \mathcal{B} \rangle = \langle \mathcal{X}, (\mathcal{B} \otimes \widetilde{\mathcal{A}} \otimes \mathcal{A}) \times_1 \boldsymbol{U}_{\mathcal{DF}}^{(1)} \times_2 \boldsymbol{U}_{\mathcal{DF}}^{(2)} \times_3 \cdots \times_N \boldsymbol{U}_{\mathcal{DF}}^{(N)} \rangle.$$

The case with multiple kernels will have a form similar to (3). Following the same logic, it can indeed be generalized to an even deeper CNN by adding more "convolution-pooling" layers. Since the composite weight tensor always has a "Tucker-like" form, techniques in proving Theorem 1 can be adopted to conduct the sample complexity analysis for deep CNNs.

In practice, zero-padding can be used before each convolution layer to preserve the input size. This is compatible with our framework, since we can propagate the padding of zeros to previous layers and all the way back to the input tensor $\mathcal{X}$.

3 COMPRESSED CNNS

For a high-order input, a deep CNN with a larger number of kernels may involve heavy computation, which renders its training difficult for portable devices with limited resources. In real applications, many compressed CNN block designs have been proposed to improve the efficiency, and most of them are based on either matrix factorization or tensor decomposition; see Lebedev et al. (2015); Kim et al. (2016); Astrid & Lee (2017); Kossaifi et al. (2020).

Tucker decomposition can be used to compress CNNs, which is same as introducing a multilayer CNN block; see Figure 3 in Kim et al. (2016). Specifically, we stack the kernels $\mathcal{A}_k \in \mathbb{R}^{l_1 \times l_2 \times \cdots \times l_N}$ with $1 \le k \le K$ into a higher order tensor $\mathcal{A}_{\text{stack}} \in \mathbb{R}^{l_1 \times l_2 \times \cdots \times l_N \times K}$, and assume that $\mathcal{A}_{\text{stack}}$ has Tucker ranks of $(R_1, R_2, \cdots, R_N, R_{N+1})$. As a result,

$$\mathcal{A}_{\text{stack}} = \mathcal{G} \times_1 \boldsymbol{H}^{(1)} \times_2 \boldsymbol{H}^{(2)} \cdots \times_{N+1} \boldsymbol{H}^{(N+1)}, \tag{6}$$

where $\mathcal{G} \in \mathbb{R}^{R_1 \times \cdots \times R_N \times R_{N+1}}$ is the core tensor, and $\boldsymbol{H}^{(j)} \in \mathbb{R}^{l_j \times R_j}$ are factor matrices for $1 \le j \le N+1$. From Section 2, to train such a compressed CNN block with linear activations is equivalent to search for the least-square estimators at (5) with $\mathcal{A}_{\text{stack}}$ being constrained to have the form of (6). Denote the estimators by $\{\widehat{\mathcal{B}}_k^{\text{TU}}, \widehat{\mathcal{A}}_k^{\text{TU}}\}_{1 \le k \le K}$. The trained weights then have a form of $\widehat{\mathcal{W}}_{\text{TU}} = \sum_{k=1}^K \widehat{\mathcal{B}}_k^{\text{TU}} \otimes \widehat{\mathcal{A}}_k^{\text{TU}}$, and the model complexity is $d_{\mathcal{M}}^{\text{TU}} = \prod_{j=1}^{N+1} R_j + \sum_{i=1}^N l_i R_i + R_{N+1} P$.

CP decomposition is more popular in compressing CNNs; see Kossaifi et al. (2020); Lebedev et al. (2015); Astrid & Lee (2017). Let $\{\widehat{\mathcal{B}}_k^{\text{CP}}, \widehat{\mathcal{A}}_k^{\text{CP}}\}_{1 \le k \le K}$ be the least-square estimators at (5) with $\mathcal{A}_{\text{stack}}$ having a CP decomposition with a rank of R. Then, the trained weights are $\widehat{\mathcal{W}}_{\text{CP}} = \sum_{k=1}^K \widehat{\mathcal{B}}_k^{\text{CP}} \otimes \widehat{\mathcal{A}}_k^{\text{CP}}$, and the model complexity is $d_{\mathcal{M}}^{\text{CP}} = R^{N+1} + R(\sum_{i=1}^N l_i + P)$.

Theorem 2 (Compressed CNN)**.** *Let $(\widehat{\mathcal{W}}, d_{\mathcal{M}})$ be $(\widehat{\mathcal{W}}_{\text{TU}}, d_{\mathcal{M}}^{\text{TU}})$ for Tucker decomposition, or $(\widehat{\mathcal{W}}_{\text{CP}}, d_{\mathcal{M}}^{\text{CP}})$ for CP decomposition. Suppose that Assumptions 1-3 hold and $n \gtrsim (\kappa_U/\kappa_L)^2 c_N d_{\mathcal{M}}$.*

Then,

$$\|\widehat{\mathcal{W}} - \mathcal{W}^*\|_{\mathrm{F}} \leq \frac{16\sigma\sqrt{\alpha_{\mathrm{RSM}}}}{\alpha_{\mathrm{RSC}}} \left(\sqrt{\frac{c_N d_{\mathcal{M}}}{n}} + \sqrt{\frac{\delta}{n}} \right),$$

for some $\delta > 0$, with probability $1 - 4\exp\{-[c_{\mathrm{H}}(0.25\kappa_L/\kappa_U)^2 n - 3c_N d_{\mathcal{M}}]\} - \exp\{-d_{\mathcal{M}} - 8\delta\}$, where $c_N = 2^{N+1}\log(N+2)$, c_{H} is a positive value and $\alpha_{\mathrm{RSC}}, \alpha_{\mathrm{RSM}}$ are defined in Theorem 1.

Theorem 2 shows that, as expected, the sample complexity is proportional to the model complexity, which however depends on the rank R_{N+1} or R, rather than the number of kernels K. This differs from the naive parameter counting, and the rationale behind it is provided in the corollary below.

Corollary 1. *(a) If $\mathcal{A}$ has a Tucker decomposition with the ranks of $(R_1, R_2, \cdots, R_{N+1})$, the CNN is then equivalent to one with R_{N+1} kernels, and each kernel $\mathcal{A}_r$, where $1 \leq r \leq R_{N+1}$, has a Tucker decomposition with ranks of $(R_1, R_2, \cdots, R_N)$. Moreover, (b) if the stacked kernel tensor $\mathcal{A}$ has a CP decomposition with rank R, the corresponding CNN can then be reparameterized into one with R kernels, and each kernel $\mathcal{A}_r$, where $1 \leq r \leq R$, has a CP decomposition form with rank R.*

Define the K/R ratio to be the value of K/R_{N+1} for Tucker decomposition or K/R for CP decomposition, and it always holds that $K \geq R$. From Theorem 2 and Corollary 1, the CNN model with linear activations will be overparameterized if the K/R ratio is greater than one. Though, with nonlinear activations, overparameterization may bring some benefit, it is often not guaranteed and depends on specific datasets. More importantly, when $K \gg R$, the parameter redundancy will, without a doubt, greatly hinder computational efficiency. We next use this finding to evaluate the efficiency of various block designs; see Figure 2.

(T1) Standard bottleneck block. The basic building block in ResNet (He et al., 2016) can be exactly replicated by a Tucker decomposition on $\mathcal{A}_{\mathrm{stack}} \in \mathbb{R}^{l_1 \times l_2 \times C \times K}$, with ranks of (l_1, l_2, R, R), where C is the number of input channels (Kossaifi et al., 2020). As shown by the ablation studies in Session 4, when $K \gg R$, this design may not be the most parameter efficient.

(T2) Funnel block. As a straightforward revision on a standard bottleneck block, the funnel block maintains an output channel size of $K = R$. This is hence an efficient block design. Also, when $R = C$, we obtain a normalized funnel block, with the same number of input and output channels.

(C1) Depthwise separable block. This block is the basic module in MobileNetV1 (Howard et al., 2017) with a pair of depthwise and pointwise separable convolution layers. It is equivalent to assuming a CP decomposition on $\mathcal{A}_{\mathrm{stack}} \in \mathbb{R}^{l_1 \times l_2 \times C \times K}$ with the rank of C (Kossaifi et al., 2020). The 1×1 pointwise convolution expands the numbers of channels from C to K. When $K \gg C$, this design is parameter inefficient from a computation viewpoint.

(C2) Inverted residual block. Sandler et al. (2018) later proposed this design in MobileNetV2. It includes expansive layers between the input and output layers, with the channel size of $x \cdot C (x \geq 1)$, where x represents the expansion factor. As discussed by Kossaifi et al. (2020), it can heuristically correspond to a CP decomposition on $\mathcal{A}_{stack}$, with CP rank equals to $x \cdot C$. Since, the rank of output channel dimension can be at most $x \cdot C$, as long as $K \leq x \cdot C$ holds, it is theoretically efficient and provides leeway for exploring thicker layers within blocks.

There have been comprehensive empirical studies in Sandler et al. (2018) showing that MobileNetV2 achieves similar or even superior accuracy with much less parameters than MobileNetV1. So, for our ablation studies, we focus on exploring the bottleneck structure of ResNet.

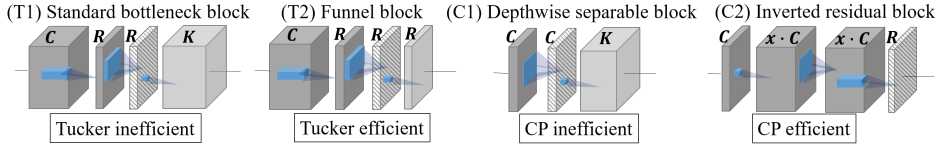

Figure 2: We use thickness of each layer to indicate its relative number of channels. The last layer indicates the output of the block. The dashed layer represents the "bottleneck" of the block.

Table 1: Different settings for verifying Theorem 1 (left), and Theorem 2 (right).

	Input sizes	Kernel sizes	Pooling sizes	# Kernels	Input sizes	Kernel sizes	Pooling sizes	Tucker ranks	# Kernels
Setting 1 (S1)	(7, 5, 7)	(2, 2, 2)	(3, 2, 3)	1	(10, 10, 8, 3)	(5, 5, 3)	(3, 3, 3)	(2, 2, 2, 1)	2
Setting 2 (S2)	(7, 5, 7)	(2, 2, 2)	(3, 2, 3)	3	(10, 10, 8, 3)	(5, 5, 3)	(3, 3, 3)	(2, 2, 2, 1)	3
Setting 3 (S3)	(8, 8, 3)	(3, 3, 3)	(3, 3, 1)	1	(12, 12, 6, 3)	(7, 7, 3)	(3, 3, 2)	(2, 3, 2, 1)	2
Setting 4 (S4)	(8, 8, 3)	(3, 3, 3)	(3, 3, 1)	3	(12, 12, 6, 3)	(7, 7, 3)	(3, 3, 2)	(2, 3, 2, 1)	3

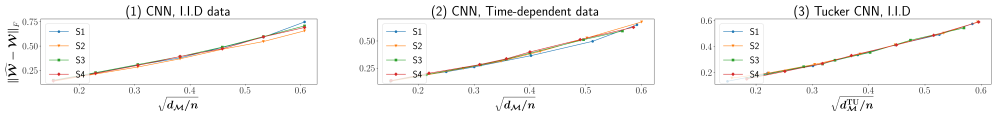

Figure 3: (1)-(2) are the experiment results for Theorem 1 with independent and dependent inputs, respectively. (3) is the experiment results for Tucker CNN in Theorem 2.

4 EXPERIMENTS

This section first verifies the theoretical results with synthetic datasets, and then conducts ablation studies on the bottleneck blocks in ResNet (He et al., 2016) with different K/R ratios.

4.1 NUMERICAL ANALYSIS FOR THEORETICAL RESULTS

We choose four settings to verify the sample complexity in Theorem 1; see Table 1. The parameter tensors $\{\mathcal{A}_k, \mathcal{B}_k\}_{k=1}^{K}$ are generated to have standard normal entries. We consider two types of inputs: the independent inputs are generated with entries being standard normal random variables, and the time-dependent inputs $\{\boldsymbol{x}^i\}$ are generated from a stationary VAR(1) process. The random additive noises are generated from the standard normal distribution. The number of training samples n varies such that $\sqrt{d_{\mathcal{M}}/n}$ is equally spaced in the interval $[0.15, 0.60]$, with all other parameters fixed. For each n, we generate 200 training sets to calculate the averaged estimation error in Frobenius norm. It can be seen that the estimation error increases linearly with the square root of $d_{\mathcal{M}}/n$, which is consistent with our finding in Theorem 1.

For Theorem 2, we also adopt four different settings; see Table 1. Here, we consider 4D input tensors. The stacked kernel $\mathcal{A}_{\text{stack}}$ is generated by (6), where the core tensor has standard normal entries and the factor matrices are generated to have orthonormal columns. The number of training samples n is similarly chosen such that $\sqrt{d_{\mathcal{M}}^{\text{TU}}/n}$ is equally spaced. A linear trend between the estimation error and the square root of $d_{\mathcal{M}}^{\text{TU}}/n$ can be observed, which verifies Theorem 2.

For details of implementation, we employed the gradient descent method for the optimization with a learning rate of 0.01 and a momentum of 0.9. The procedure is deemed to have reached convergence if the target function drops by less than 10^{-8}.

4.2 ABLATION STUDIES ON COMPUTATIONALLY-EFFICIENT BLOCK DESIGNS

Following the notation in Section 3, we denote by K, the number of output channels of a bottleneck block, and by R, the Tucker rank of $\mathcal{A}_{\text{stack}}$ along the output channels dimension. Since $K \geq R$, we let $t = K/R$ be the expansive ratio of the block. As t increases, the number of parameters within the block increases. Here, ablation studies are conducted to show that the increase in t does not necessarily lead to the increase in test accuracy. We analyze two image recognition datasets, CIFAR-

Table 2: Test accuracy(%) on CIFAR-10, SVHN and UCF101. For UCF101, we only count the number of parameters and FLOPs for the added blocks.

t	CIFAR-10	SVHN	#FLOPs	#Params	UCF101	#Added FLOPs	#Added Params
1	93.53	96.30	0.17GMac	5.56M	79.31	0.125GMac	6.97M
4	93.10	96.09	0.26GMac	8.55M	79.36	0.145GMac	9.16M
8	93.59	96.32	0.43GMac	14.15M	79.25	0.175GMac	11.11M
16	94.03	96.31	0.87GMac	29.28M	79.52	0.235GMac	15.00M

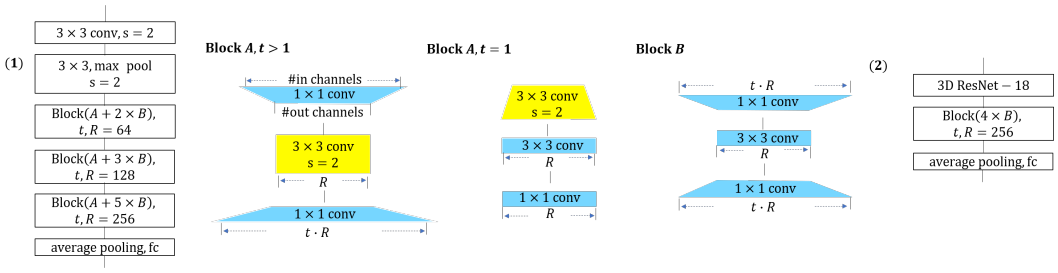

Figure 4: (1) 41-layer Residual Network for CIFAR-10&SVNH. (2) 30-layer 3D Residual Network for UCF101. Here, $t = K/R$ represents the expansion ratio, and takes values of 1,4,8,16.

10 (Krizhevsky et al., 2009) and Street View House Numbers (SVHN) (BNetzer et al., 2011), and an action recognition dataset, UCF101 (Soomro et al., 2012). Standard data pre-processing and augmentation techniques are adopted to all three datasets (He et al., 2016; Hara et al., 2017).

Network architecture. Two network architectures are considered in our study; as shown in Figure 4. For image recognition datasets, we adopt a 41-layer Residual Network: it consists of a 3×3 convolution layer, a max pooling layer, followed by 3 groups of $A + x \times B$ residual blocks with different R, and ends with an average pooling and fully-connected layer. Block B represents the bottleneck structure that we are interested in. The standard bottleneck block corresponds to $t = 4$. When $t = 1$, it corresponds to the normalized funnel block in Section 3 (T2). For sake of comparison, we also take $t = 8$ or 16. When $t > 1$, Block A is the standard downsampling block that contains convolution with the stride size of 2. It is inefficient according to our study. So when $t = 1$, we propose minor revisions to simultaneously increase the channels while performing the stride-2 3×3 convolution. For UCF101 dataset, we use 3D ResNet-18 as the basic framework and insert 4 stacks of Block B with $R = 256$ before the average pooling and fully-connected layer.

Results. The results in Table 2 provide empirical support for our theoretical finding. Though the number of parameters in the network increases with larger t, the overall test accuracy remains roughly comparable. For CIFAR-10, it may appear that the test accuracy for $t = 16$ is slightly better than $t = 1$ but it uses 5 times as many parameters.

Implementation details. All experiments are conducted in PyTorch and on Tesla V100-DGXS. Following the practice in He et al. (2016), we adopt batch normalization(BN) (Ioffe & Szegedy, 2015) right after each convolution and before the ReLU activation. For network (1), we initialize the weights as in He et al. (2015). We use stochastic gradient descent with weight decay 10^{-4}, momentum 0.9 and mini-batch size 128. The learning rate starts from 0.1 and is divided by 10 for every 100 epochs. We stop training after 300 epochs, since the training accuracy hardly changes. The training accuracy is approximately 93% for CIFAR-10 and 96% for SVHN. We set seeds 1-5 and report the worst case scenario of the test accuracy in Table 2. For network (2), we follow Hara et al. (2018) and use the Kinetics pretrained 3D ResNet-18. The weights are fixed for the first few layers, and backward propagation is only applied to the added Block B layers and the fully-connected layer. We use stochastic gradient descent with weight decay 10^{-5}, momentum 0.9 and mini-batch size 64 with 16 frames per clip. The learning rate starts from 0.001 and divided by 10 for every 30 epochs. Training is stopped after 80 epochs and top-1 clip accuracy is around 79% for UCF101.

5 Conclusion

Our paper proposes a unified theoretical framework that can adopt sample complexity analysis as a tool to study the effective number of parameter in a tensor decomposed CNN block. The main practical takeaways are: (i) For a large kernel tensor $\mathcal{A}$ (Peng et al., 2017), it is always effective to impose low-rank structure on its input dimensions, such as height, width or time-length, etc; and (ii) It is essential to maintain $K/R = 1$ for a CNN block in order to increase accuracy in a more parameter efficient way. In this regard, one can either choose a small R with deeper networks or a larger R with shallower networks. In fact, when $K/R = 1$, the block corresponds exactly to the "straightened" bottleneck in Zagoruyko & Komodakis (2016), who showed in their empirical study that by increasing R in shallow networks, the test accuracy can indeed improve quite significantly.

REFERENCES

Sanjeev Arora, Rong Ge, Behnam Neyshabur, and Yi Zhang. Stronger generalization bounds for deep nets via a compression approach. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80, pp. 254–263, 2018.

Marcella Astrid and Seung-Ik Lee. Cp-decomposition with tensor power method for convolutional neural networks compression. In *2017 IEEE International Conference on Big Data and Smart Computing (BigComp)*, pp. 115–118. IEEE, 2017.

Peter L Bartlett, Dylan J Foster, and Matus J Telgarsky. Spectrally-normalized margin bounds for neural networks. In *Advances in Neural Information Processing Systems*, pp. 6240–6249, 2017.

Peter L Bartlett, Nick Harvey, Christopher Liaw, and Abbas Mehrabian. Nearly-tight vc-dimension and pseudodimension bounds for piecewise linear neural networks. *Journal of Machine Learning Research*, 20(63):1–17, 2019.

Sumanta Basu and George Michailidis. Regularized estimation in sparse high-dimensional time series models. *The Annals of Statistics*, 43(4):1535–1567, 2015.

Yuval BNetzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. In *In NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*, 2011.

Emmanuel J Candes and Yaniv Plan. Tight oracle inequalities for low-rank matrix recovery from a minimal number of noisy random measurements. *IEEE Transactions on Information Theory*, 57 (4):2342–2359, 2011.

Yuan Cao and Quanquan Gu. Tight sample complexity of learning one-hidden-layer convolutional neural networks. In *Advances in Neural Information Processing Systems*, pp. 10611–10621, 2019.

Yu Cheng, Fei Wang, Ping Zhang, and Jianying Hu. Risk prediction with electronic health records: A deep learning approach. In *Proceedings of the 2016 SIAM International Conference on Data Mining*, pp. 432–440. SIAM, 2016.

Misha Denil, Babak Shakibi, Laurent Dinh, Marc'Aurelio Ranzato, and Nando De Freitas. Predicting parameters in deep learning. In *Advances in Neural Information Processing Systems*, pp. 2148–2156, 2013.

Emily L Denton, Wojciech Zaremba, Joan Bruna, Yann LeCun, and Rob Fergus. Exploiting linear structure within convolutional networks for efficient evaluation. In *Advances in Neural Information Processing Systems*, pp. 1269–1277, 2014.

Simon S Du and Surbhi Goel. Improved learning of one-hidden-layer convolutional neural networks with overlaps. *arXiv preprint arXiv:1805.07798*, 2018.

Simon S Du, Yining Wang, Xiyu Zhai, Sivaraman Balakrishnan, Russ R Salakhutdinov, and Aarti Singh. How many samples are needed to estimate a convolutional neural network? In *Advances in Neural Information Processing Systems*, pp. 373–383, 2018.

Haoyu Fu, Yuejie Chi, and Yingbin Liang. Guaranteed recovery of one-hidden-layer neural networks via cross entropy. *IEEE Transactions on Signal Processing*, 2020.

Surbhi Goel, Adam Klivans, and Raghu Meka. Learning one convolutional layer with overlapping patches. *arXiv preprint arXiv:1802.02547*, 2018.

Noah Golowich, Alexander Rakhlin, and Ohad Shamir. Size-independent sample complexity of neural networks. In *Proceedings of the 31st Conference On Learning Theory*, volume 75, pp. 297–299, 2018.

Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. Learning spatio-temporal features with 3d residual networks for action recognition. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pp. 3154–3160, 2017.

Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet? In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 6546–6555, 2018.

Kohei Hayashi, Taiki Yamaguchi, Yohei Sugawara, and Shin-ichi Maeda. Exploring unexplored tensor network decompositions for convolutional neural networks. In *Advances in Neural Information Processing Systems*, pp. 5552–5562, 2019.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on Computer Vision*, pp. 1026–1034, 2015.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016.

Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 4700–4708, 2017.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning*, pp. 448–456, 2015.

Yong-Deok Kim, Eunhyeok Park, Sungjoo Yoo, Taelim Choi, Lu Yang, and Dongjun Shin. Compression of deep convolutional neural networks for fast and low power mobile applications. In *International Conference on Learning Representations*, 2016.

Tamara G Kolda and Brett W Bader. Tensor decompositions and applications. *SIAM Review*, 51(3): 455–500, 2009.

Jean Kossaifi, Antoine Toisoul, Adrian Bulat, Yannis Panagakis, Timothy M Hospedales, and Maja Pantic. Factorized higher-order cnns with an application to spatio-temporal emotion estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6060–6069, 2020.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. *Tech Report*, 2009.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pp. 1097–1105, 2012.

Vadim Lebedev, Yaroslav Ganin, Maksim Rakhuba, Ivan Oseledets, and Victor Lempitsky. Speeding-up convolutional neural networks using fine-tuned cp-decomposition. In *International Conference on Learning Representations*, 2015.

Xingguo Li, Junwei Lu, Zhaoran Wang, Jarvis Haupt, and Tuo Zhao. On tighter generalization bound for deep neural networks: Cnns, resnets, and beyond. *arXiv preprint arXiv:1806.05159*, 2018.

Behnam Neyshabur, Ryota Tomioka, and Nathan Srebro. Norm-based capacity control in neural networks. In *Conference on Learning Theory*, pp. 1376–1401, 2015.

Chao Peng, Xiangyu Zhang, Gang Yu, Guiming Luo, and Jian Sun. Large kernel matters–improve semantic segmentation by global convolutional network. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 4353–4361, 2017.

Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 4510–4520, 2018.

Max Simchowitz, Horia Mania, Stephen Tu, Michael I Jordan, and Benjamin Recht. Learning without mixing: Towards a sharp analysis of linear system identification. *Proceedings of Machine Learning Research vol*, 75:1–35, 2018.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.

Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

Qiuling Suo, Fenglong Ma, Ye Yuan, Mengdi Huai, Weida Zhong, Aidong Zhang, and Jing Gao. Personalized disease prediction using a cnn-based similarity learning method. In *2017 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pp. 811–816. IEEE, 2017.

Taiji Suzuki, Hiroshi Abe, Tomoya Murata, Shingo Horiuchi, Kotaro Ito, Tokuma Wachi, So Hirai, Masatoshi Yukishima, and Tomoaki Nishimura. Spectral-pruning: Compressing deep neural network via spectral analysis. *arXiv preprint arXiv:1808.08558*, 2018.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 1–9, 2015.

Du Tran, Heng Wang, Lorenzo Torresani, Jamie Ray, Yann LeCun, and Manohar Paluri. A closer look at spatiotemporal convolutions for action recognition. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 6450–6459, 2018.

Ledyard R Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31(3): 279–311, 1966.

R. Vershynin. *High-dimensional probability: an introduction with applications in data science*. Cambridge University Press, Cambridge, 2018.

Di Wang, Feiqing Huang, Jingyu Zhao, Guodong Li, and Guangjian Tian. Compact autoregressive network. *arXiv preprint arXiv:1909.03830*, 2019.

Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

Kai Zhong, Zhao Song, Prateek Jain, Peter L Bartlett, and Inderjit S Dhillon. Recovery guarantees for one-hidden-layer neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, pp. 4140–4149, 2017.

A APPENDIX

This appendix contains four sections. In the first section, we provide details for our mathematical formulation for a high-order 3-layer CNN. In the next section, we discuss the formulation of a 5-layer CNN as an illustrating example of how to add more layers to our framework. A complete version of Theorems 1&2 together with the technical proofs for Theorems 1&2 and Corollary 1 are provided in the third section. More implementation details and results for additional experiments are presented in the last section.

A.1 CNN FORMULATION

For a tensor input $\mathcal{X} \in \mathbb{R}^{d_1 \times \cdots \times d_N}$, it first convolutes with an N-dimensional kernel tensor $\mathcal{A} \in \mathbb{R}^{l_1 \times \cdots \times l_N}$ with stride sizes equal to s_c, and then performs an average pooling with pooling sizes equal to $(q_1, \cdots, q_N)$. It ends with a fully-connected layer with the weight tensor $\mathcal{B}$ and produces a scalar output. We assume that $m_j = (d_j - l_j)/s_c + 1$ are integers for $1 \leq j \leq N$, otherwise zero-padding will be needed. For ease of notation, we take the pooling sizes $\{q_j\}_{j=1}^N$ to satisfy the relationship $m_j = p_j q_j$.

To duplicate the operation of the convolution layer using a simple mathematical expression, we first need to define a set of matrices $\{\boldsymbol{U}_{i_j}^{(j)} \in \mathbb{R}^{d_j \times l_j}\}$, where

$$\boldsymbol{U}_{i_j}^{(j)} = [\ \underbrace{\boldsymbol{0}}_{(i_j-1)s_c}\ \underbrace{\boldsymbol{I}}_{l_j}\ \underbrace{\boldsymbol{0}}_{d_j-(i_j-1)s_c-l_j}\ ]' \in \mathbb{R}^{d_j \times l_j} \text{ for } 1 \leq i_j \leq m_j,\ 1 \leq j \leq N. \tag{7}$$

$\boldsymbol{U}_{i_j}^{(j)}$ acts as a positioning factor to transform the kernel tensor $\mathcal{A}$ into a tensor of same size as the input $\mathcal{X}$, with the rest of the entries equal to zero.

We are ready to construct our main formulation for a 3-layer tensor CNN. To begin with, we first illustrate the process using a vector input $\boldsymbol{x} \in \mathbb{R}^d$, with a kernel vector $\boldsymbol{a} \in \mathbb{R}^l$; see Fig 5. Using the ith positioning matrix $\boldsymbol{U}_i^{(1)}$, we can propagate the small kernel vector $\boldsymbol{a}$, into a vector $\boldsymbol{U}_i^{(1)}\boldsymbol{a}$ of size $\mathbb{R}^d$, by filling the rest of the entries with zeros. The intermediate output vector has entries given by $\boldsymbol{x}_c(i) = \langle \boldsymbol{x}, \boldsymbol{U}_i^{(1)} \rangle$, for $1 \leq i \leq m$. The average pooling operation is equivalent to forming p consecutive vectors within $\boldsymbol{x}_c$, each of size $\mathbb{R}^q$, and then take the average. This results in a vector $\boldsymbol{x}_{cp}$ of size $\mathbb{R}^p$, with ith entry equal to $\boldsymbol{x}_{cp}(i) = q^{-1} \sum_{k=(i-1)q+1}^{iq} \boldsymbol{x}_c(i) = \langle \boldsymbol{x}, q^{-1} \sum_{k=(i-1)q+1}^{iq} \boldsymbol{U}_k^{(1)} \boldsymbol{a} \rangle$. The fully-connected layer performs a weighted summation over the p vectors, with weights given by entries of the vector $\boldsymbol{b} \in \mathbb{R}^p$. This gives us the predicted output $\widehat{y} = \langle \boldsymbol{b}, \boldsymbol{x}_{cp} \rangle = \langle \boldsymbol{x}, \boldsymbol{w}_X \rangle$, where

$$\boldsymbol{w}_X = \sum_{i=1}^{p} b_i \frac{1}{q} \sum_{k=(i-1)q+1}^{iq} \boldsymbol{U}_k^{(1)} \boldsymbol{a} = \boldsymbol{U}_{\mathcal{F}}^{(1)} (\boldsymbol{b} \otimes \boldsymbol{a}) = (\boldsymbol{b} \otimes \boldsymbol{a}) \times_1 \boldsymbol{U}_{\mathcal{F}}^{(1)},$$

and $\boldsymbol{U}_{\mathcal{F}}^{(1)} = q^{-1}(\sum_{k=1}^{q} \boldsymbol{U}_k^{(1)}, \cdots, \sum_{k=m-q+1}^{m} \boldsymbol{U}_k^{(1)})$, and "$\times_1$" represents the mode-1 product.

For matrix input $\boldsymbol{X}$ with matrix kernel $\boldsymbol{A}$, however, we need 2 sets of positioning matrices, $\{\boldsymbol{U}_{i_1}^{(1)}\}_{i_1=1}^{m_1}$ and $\{\boldsymbol{U}_{i_2}^{(2)}\}_{i_2=1}^{m_2}$, one for the height dimension and the other for width. Then, the intermediate output from convolution has entries given by $\boldsymbol{X}_c(i_1, i_2) = \langle \boldsymbol{X}, \boldsymbol{U}_{i_1}^{(1)} \boldsymbol{A} \boldsymbol{U}_{i_2}^{(2)\prime} \rangle$, for $1 \leq i_1 \leq p_1$, $1 \leq i_2 \leq p_2$. For the average pooling, we form $p_1 p_2$ consecutive matrices from $\boldsymbol{X}_c$, each of size $\mathbb{R}^{q_1 \times q_2}$ and take the average. This results in $\boldsymbol{X}_{cp} \in \mathbb{R}^{p_1 \times p_2}$, with $\boldsymbol{X}_{cp}(i_1, i_2) = \langle \boldsymbol{X}, (q_1^{-1} \sum_{k_1=(i_1-1)q_1+1}^{i_1 q_1} \boldsymbol{U}_{k_1}^{(1)}) \boldsymbol{A} (q_2^{-1} \sum_{k_2=(i_2-1)q_2+1}^{i_2 q_2} \boldsymbol{U}_{k_2}^{(2)})' \rangle$. The output $\boldsymbol{X}_{cp}$ goes through fully-connected layer with weight matrix $\boldsymbol{B}$ and gives the predicted output $\widehat{y} = \langle \boldsymbol{X}, \boldsymbol{W}_X \rangle$, where

$$\boldsymbol{W}_X = \sum_{i_2=1}^{p_2} \sum_{i_1=1}^{p_1} b_{i_1} b_{i_2} \frac{1}{q_1 q_2} (\sum_{k_1=(i_1-1)q_1+1}^{i_1 q_1} \boldsymbol{U}_{k_1}^{(1)}) \boldsymbol{A} (\sum_{k_2=(i_2-1)q_2+1}^{i_2 q_2} \boldsymbol{U}_{k_2}^{(2)\prime}) = (\boldsymbol{B} \otimes \boldsymbol{A}) \times_1 \boldsymbol{U}_{\mathcal{F}}^{(1)} \times_2 \boldsymbol{U}_{\mathcal{F}}^{(2)},$$

with $\boldsymbol{U}_{\mathcal{F}}^{(j)} = q_j^{-1}(\sum_{k=1}^{q_j} \boldsymbol{U}_k^{(j)}, \cdots, \sum_{k=m_j-q_j+1}^{m_j} \boldsymbol{U}_k^{(j)})$ for $j = 1$ or 2. And "$\times_1$", "$\times_2$" represent the mode-1 and mode-2 product.

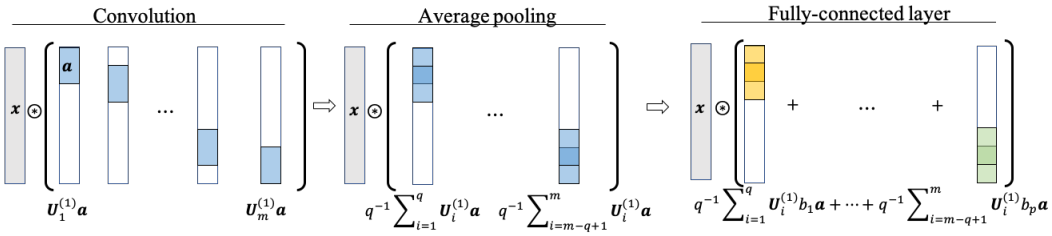

Figure 5: Formulating the process with an input vector $\boldsymbol{x}$. Note that we combine the convolution, pooling and fully connected layers into the composite weight vector $\boldsymbol{w}_X$, where $\boldsymbol{w}_X = q^{-1}\sum_{i=1}^{q}\boldsymbol{U}_i^{(1)}\boldsymbol{a} + \cdots + q^{-1}\sum_{i=m-q+1}^{q}\boldsymbol{U}_i^{(1)}\boldsymbol{a}$. The $\circledast$ represents the convolution operation.

From here, together with the case of high-order tensor input discussed in Session 2.2, we can then derive the form of predicted outcome as $\widehat{y} = \langle \mathcal{X}, \mathcal{W}_X^{\text{single}} \rangle$, where

$$\mathcal{W}_X^{\text{single}} = (\mathcal{B} \otimes \mathcal{A}) \times_1 \boldsymbol{U}_{\mathcal{F}}^{(1)} \times_2 \boldsymbol{U}_{\mathcal{F}}^{(2)} \times \cdots \times \boldsymbol{U}_{\mathcal{F}}^{(N)},$$

with $\boldsymbol{U}_{\mathcal{F}}^{(j)} = q_j^{-1}(\sum_{k=1}^{q_j}\boldsymbol{U}_k^{(j)}, \cdots, \sum_{k=m_j-q_j+1}^{m_j}\boldsymbol{U}_k^{(j)})$, and "$\times_j$" represents the mode-$j$ product for $1 \leq j \leq N$.

With K kernels, we denote the set of kernels and the corresponding fully-connected weight tensors as $\{\mathcal{A}_k, \mathcal{B}_k\}_{k=1}^K$. Since the convolution and pooling operations are identical across kernels, we can use a summation over kernels to derive the weight tensor for multiple kernels, which is

$$\mathcal{W}_X = (\sum_{k=1}^{K}\mathcal{B}_k \otimes \mathcal{A}_k) \times_1 \boldsymbol{U}_{\mathcal{F}}^{(1)} \times_2 \boldsymbol{U}_{\mathcal{F}}^{(2)} \times \cdots \times \boldsymbol{U}_{\mathcal{F}}^{(N)},$$

and we arrive at the formulation for 3-layer tensor CNN.

A.2 Five-layer CNN formulation

Consider a 5-layer CNN with "convolution→pooling→convolution→pooling→fully connected" layers and a 3D tensor input $\mathcal{X} \in \mathbb{R}^{d_1 \times d_2 \times d_3}$. Here, we denote the intermediate output from the first convolution by $\mathcal{X}_c \in \mathbb{R}^{m_1 \times m_2 \times m_3}$, from the second convolution by $\widetilde{\mathcal{X}}_c \in \mathbb{R}^{\tilde{m}_1 \times \tilde{m}_2 \times \tilde{m}_3}$, and the output from the first pooling by $\mathcal{X}_{cp} \in \mathbb{R}^{p_1 \times p_2 \times p_3}$ and from the second pooling by $\widetilde{\mathcal{X}}_{cp} \in \mathbb{R}^{\tilde{p}_1 \times \tilde{p}_2 \times \tilde{p}_3}$. We can first see that the predicted output from the 5-layer CNN is the same as directly feeding $\mathcal{X}_{cp}$ to the second convolution layer, followed by an average pooling layer and a fully-connected layer.

Denote the first convolution kernel tensor by $\mathcal{A} \in \mathbb{R}^{l_1 \times l_2 \times l_3}$, the second convolution kernel tensor by $\mathcal{A} \in \mathbb{R}^{\tilde{l}_1 \times \tilde{l}_2 \times \tilde{l}_3}$, and the fully-connected weight tensor by $\mathcal{B} \in \mathbb{R}^{\tilde{p}_1 \times \tilde{p}_2 \times \tilde{p}_3}$. Define the set of matrices $\{\boldsymbol{U}_{i_j}^{(j)} \in \mathbb{R}^{d_j \times l_j}\}$, for $1 \leq i_j \leq m_j, 1 \leq j \leq N$ as in (1). And similarly, define $\{\widetilde{\boldsymbol{U}}_{i_j}^{(j)} \in \mathbb{R}^{p_j \times \tilde{l}_j}\}$, with

$$\widetilde{\boldsymbol{U}}_{i_j}^{(j)} = [\underbrace{\boldsymbol{0}}_{(i_j-1)\tilde{s}_c}\ \underbrace{\boldsymbol{I}}_{\tilde{l}_j}\ \underbrace{\boldsymbol{0}}_{p_j-(i_j-1)\tilde{s}_c-\tilde{l}_j}]', \tag{8}$$

where $\tilde{s}_c$ is the stride size for the second convolution. Let $\widetilde{\boldsymbol{U}}_{\mathcal{F}}^{(j)} = (\widetilde{\boldsymbol{U}}_{\mathcal{F},1}^{(j)}, \cdots, \widetilde{\boldsymbol{U}}_{\mathcal{F},p_j}^{(j)}) = (\tilde{q}_j^{-1}\sum_{k=1}^{\tilde{q}_j}\widetilde{\boldsymbol{U}}_k^{(j)}, \cdots, \tilde{q}_j^{-1}\sum_{k=\tilde{m}_j-\tilde{q}_j+1}^{\tilde{m}_j}\widetilde{\boldsymbol{U}}_k^{(j)})$. We further stack the matrices $\{\boldsymbol{U}_{\mathcal{F},i_j}^{(j)} \in \mathbb{R}^{d_j \times l_j}\}_{1 \leq i_j \leq p_j}$ and $\{\widetilde{\boldsymbol{U}}_{\mathcal{F},i_j}^{(j)} \in \mathbb{R}^{p_j \times \tilde{l}_j}\}_{1 \leq i_j \leq \tilde{p}_j}$ into 3D tensors $\mathcal{U}^{(j)} \in \mathbb{R}^{d_j \times l_j \times p_j}$ and $\widetilde{\mathcal{U}}^{(j)} \in \mathbb{R}^{p_j \times \tilde{l}_j \times \tilde{p}_j}$.

We have that the predicted output

$$
\begin{aligned}
\widehat{y} &= \langle \mathcal{B} \otimes \widetilde{\mathcal{A}}, \mathcal{X}_{cp} \times_1 \widetilde{\boldsymbol{U}}_{\mathcal{F}}^{(1)\prime} \times_2 \widetilde{\boldsymbol{U}}_{\mathcal{F}}^{(2)\prime} \times_3 \widetilde{\boldsymbol{U}}_{\mathcal{F}}^{(3)\prime} \rangle \\
&= \left\langle \mathcal{B} \otimes \widetilde{\mathcal{A}}, \sum_{i_1=1}^{p_1} \sum_{i_2=1}^{p_2} \sum_{i_3=1}^{p_3} \langle \mathcal{A}, \mathcal{X} \times_1 \boldsymbol{U}_{\mathcal{F},i_1}^{(1)\prime} \times_2 \boldsymbol{U}_{\mathcal{F},i_2}^{(2)\prime} \times_3 \boldsymbol{U}_{\mathcal{F},i_3}^{(3)\prime} \rangle \widetilde{\boldsymbol{u}}_{i_1} \circ \widetilde{\boldsymbol{u}}_{i_2} \circ \widetilde{\boldsymbol{u}}_{i_3} \right\rangle \\
&= \left\langle \mathcal{B} \otimes \widetilde{\mathcal{A}} \otimes \mathcal{A}, \sum_{i_1=1}^{p_1} \sum_{i_2=1}^{p_2} \sum_{i_3=1}^{p_3} \mathcal{X} \times_1 (\widetilde{\boldsymbol{u}}_{i_1} \otimes \boldsymbol{U}_{\mathcal{F},i_1}^{(1)\prime}) \times_2 (\widetilde{\boldsymbol{u}}_{i_2} \otimes \boldsymbol{U}_{\mathcal{F},i_2}^{(2)\prime}) \times_3 (\widetilde{\boldsymbol{u}}_{i_3} \otimes \boldsymbol{U}_{\mathcal{F},i_3}^{(3)\prime}) \right\rangle \\
&= \left\langle \mathcal{B} \otimes \widetilde{\mathcal{A}} \otimes \mathcal{A}, \mathcal{X} \times_1 (\mathcal{U}^{(1)} \times_3 \widetilde{\mathcal{U}}^{(1)})_{(1)}' \times_2 (\mathcal{U}^{(2)} \times_3 \widetilde{\mathcal{U}}^{(2)})_{(1)}' \times_3 (\mathcal{U}^{(3)} \times_3 \widetilde{\mathcal{U}}^{(3)})_{(1)}' \right\rangle \\
&= \left\langle \mathcal{X}, (\mathcal{B} \otimes \widetilde{\mathcal{A}} \otimes \mathcal{A}) \times_1 (\mathcal{U}^{(1)} \times_3 \widetilde{\mathcal{U}}^{(1)})_{(1)} \times_2 (\mathcal{U}^{(2)} \times_3 \widetilde{\mathcal{U}}^{(2)})_{(1)} \times_3 (\mathcal{U}^{(3)} \times_3 \widetilde{\mathcal{U}}^{(3)})_{(1)} \right\rangle \\
&= \left\langle \mathcal{X}, (\mathcal{B} \otimes \widetilde{\mathcal{A}} \otimes \mathcal{A}) \times_1 \boldsymbol{U}_{\mathcal{DF}}^{(1)} \times_2 \boldsymbol{U}_{\mathcal{DF}}^{(2)} \times_3 \boldsymbol{U}_{\mathcal{DF}}^{(3)} \right\rangle
\end{aligned}
$$

where $\widetilde{\boldsymbol{u}}_{i_j} = \mathrm{vec}(\widetilde{\mathcal{U}}^{(1)}(i_j, :, :)) \in \mathbb{R}^{\tilde{l}_j \tilde{p}_j}$ and $\boldsymbol{U}_{\mathcal{DF}}^{(j)} = (\mathcal{U}^{(j)} \times_3 \widetilde{\mathcal{U}}^{(j)})_{(1)}$, for $1 \leq i_j \leq p_j$ and $1 \leq j \leq 3$.

A.3 Theoretical results and technical proofs

Let $\mathcal{Z}^i = \mathcal{X}^i \times_1 \boldsymbol{U}_{\mathcal{F}}^{(1)\prime} \times_2 \boldsymbol{U}_{\mathcal{F}}^{(2)\prime} \times_3 \cdots \times_N \boldsymbol{U}_{\mathcal{F}}^{(N)\prime} \in \mathbb{R}^{l_1 p_1 \times l_2 p_2 \times \cdots \times l_N p_N}$. The trained weights have the form of $\widehat{\mathcal{W}} = \sum_{k=1}^{K} \widehat{\mathcal{B}}_k \otimes \widehat{\mathcal{A}}_k$, where

$$
\{\widehat{\mathcal{B}}_k, \widehat{\mathcal{A}}_k\}_{1 \leq k \leq K} = \underset{\mathcal{B}_k, \mathcal{A}_k, 1 \leq k \leq K}{\arg\min} \frac{1}{n} \sum_{i=1}^{n} \left(y^i - \sum_{k=1}^{K} \langle \mathcal{Z}^i, \mathcal{B}_k \otimes \mathcal{A}_k \rangle \right)^2.
$$

Given a test sample $(\mathcal{X}, y)$, where $y = \langle \mathcal{X}, \mathcal{W}_X \rangle + \xi$, and $(\mathcal{X}, \xi)$ satisfies Assumptions 1 and 2. The mean-square training and test error are defined as

$$
\mathrm{err}_{\mathrm{train}}(\widehat{\mathcal{W}}) = \sqrt{\frac{1}{n} \sum_{i=1}^{n} \left(y^i - \langle \mathcal{X}^i, \widehat{\mathcal{W}}_X \rangle \right)^2} \text{ and } \mathrm{err}(\widehat{\mathcal{W}}) = \sqrt{\mathbb{E}_{(\mathcal{X},y)} \left(y - \langle \mathcal{X}, \widehat{\mathcal{W}}_X \rangle \right)^2}.
$$

Denote $\kappa_U = C_x C_u$ and $\kappa_L = c_x c_u$. Let $\mathcal{W}^* = \sum_{k=1}^{K} \mathcal{B}_k^* \otimes \mathcal{A}_k^*$ be the true weights, and $\|\mathcal{W}\|_n^2 := n^{-1} \sum_{i=1}^{n} (\langle \mathcal{Z}^i, \mathcal{W} \rangle)^2$ be the empirical norm with respect to $\mathcal{W}$. The model complexity of CNN at equation (1) or (2) is $d_{\mathcal{M}} = K(P + L + 1)$.

Theorem 1 (Complete Version)**.** *Suppose that Assumptions 1-3 hold and $n \gtrsim (\kappa_U/\kappa_L)^2 d_{\mathcal{M}}$. Then, for some $\delta > 0$,*

$$
\begin{aligned}
&(a) \quad \|\widehat{\mathcal{W}} - \mathcal{W}^*\|_{\mathrm{F}} \leq \frac{16\sigma\sqrt{\alpha_{\mathrm{RSM}}}}{\alpha_{\mathrm{RSC}}} \left(\sqrt{\frac{d_{\mathcal{M}}}{n}} + \sqrt{\frac{\delta}{n}} \right) \\
&(b) \quad \|\widehat{\mathcal{W}} - \mathcal{W}^*\|_{n} \leq \frac{16\sigma\alpha_{\mathrm{RSM}}}{\alpha_{\mathrm{RSC}}} \left(\sqrt{\frac{d_{\mathcal{M}}}{n}} + \sqrt{\frac{\delta}{n}} \right) \\
&(c) \quad \mathit{err}^2(\widehat{\mathcal{W}}) \leq \left(\frac{16\sigma\alpha_{\mathrm{RSM}}}{\alpha_{\mathrm{RSC}}} \right)^2 \left(\frac{d_{\mathcal{M}}}{n} + \frac{\delta}{n} \right) + \sigma^2,
\end{aligned}
$$

with probability $1 - 4\exp\{-[c_{\mathrm{H}}(0.25\kappa_L/\kappa_U)^2 n - 9d_{\mathcal{M}}]\} - \exp\{-d_{\mathcal{M}} - 8\delta\}$, where c_{H} is a positive value, $\alpha_{\mathrm{RSC}} = \kappa_L/2$, and $\alpha_{\mathrm{RSM}} = 3\kappa_U/2$.

It can be seen that the estimation error, $\|\widehat{\mathcal{W}} - \mathcal{W}^*\|_{\mathrm{F}}$, is $O_p(\sqrt{d_{\mathcal{M}}/n})$, and hence the sample complexity is of order $O(d_{\mathcal{M}}/\varepsilon^2)$, to achieve a training error ε. The empirical norm is equivalent to the training error minus the sample standard deviation of noise. And we can see that the training error $\mathrm{err}_{\mathrm{train}}(\widehat{\mathcal{W}}) = O_p(\sqrt{d_{\mathcal{M}}/n}) + \widehat{\sigma}$ and test error $\mathrm{err}(\widehat{\mathcal{W}}) = O_p(\sqrt{d_{\mathcal{M}}/n}) + \sigma$, where $\widehat{\sigma}$ and σ represent the sample and population standard deviation of noise, respectively.

Now we proceed to provide a complete version of the sample complexity of compressed CNNs. Let $d_{\mathcal{M}}^{\mathrm{TU}} = \prod_{j=1}^{N+1} R_j + \sum_{i=1}^{N} l_i R_i + R_{N+1} P$ and $d_{\mathcal{M}}^{\mathrm{CP}} = R^{N+1} + R(\sum_{i=1}^{N} l_i + P)$.

Theorem 2 (Complete Version)**.** *Let $(\widehat{\mathcal{W}}, d_{\mathcal{M}})$ be $(\widehat{\mathcal{W}}_{\mathrm{TU}}, d_{\mathcal{M}}^{\mathrm{TU}})$ for Tucker decomposition, or $(\widehat{\mathcal{W}}_{\mathrm{CP}}, d_{\mathcal{M}}^{\mathrm{CP}})$ for CP decomposition. Suppose that Assumptions 1-3 hold and $n \gtrsim (\kappa_U/\kappa_L)^2 c_N d_{\mathcal{M}}$. Then, for some $\delta > 0$,*

$$(a)\|\widehat{\mathcal{W}} - \mathcal{W}^*\|_{\mathrm{F}} \leq \frac{16\sigma\sqrt{\alpha_{\mathrm{RSM}}}}{\alpha_{\mathrm{RSC}}}\left(\sqrt{\frac{c_N d_{\mathcal{M}}}{n}} + \sqrt{\frac{\delta}{n}}\right),$$

$$(b)\|\widehat{\mathcal{W}} - \mathcal{W}^*\|_{n} \leq \frac{16\sigma\alpha_{\mathrm{RSM}}}{\alpha_{\mathrm{RSC}}}\left(\sqrt{\frac{c_N d_{\mathcal{M}}}{n}} + \sqrt{\frac{\delta}{n}}\right),$$

$$(c)err^2(\widehat{\mathcal{W}}) \leq \left(\frac{16\sigma\alpha_{\mathrm{RSM}}}{\alpha_{\mathrm{RSC}}}\right)^2\left(\frac{c_N d_{\mathcal{M}}}{n} + \frac{\delta}{n}\right) + \sigma^2,$$

with probability $1 - 4\exp\{-[c_{\mathrm{H}}(0.25\kappa_L/\kappa_U)^2 n - 3c_N d_{\mathcal{M}}]\} - \exp\{-d_{\mathcal{M}} - 8\delta\}$, where $c_N = 2^{N+1}\log(N+2)$, c_{H} is a positive value and $\alpha_{\mathrm{RSC}}, \alpha_{\mathrm{RSM}}$ are defined in Theorem 1.

Because CP decomposition can be considered as a special case of Tucker decomposition where $R_j = R$, for $1 \leq j \leq N+1$. We only provide the proof for Tucker CNN.

A.3.1 Proof of Theorem 1

Denote the sets $\widehat{\mathcal{S}}_K = \{\sum_{k=1}^{K} \mathcal{B}_k \otimes \mathcal{A}_k : \mathcal{A}_k \in \mathbb{R}^{l_1 \times l_2 \times \cdots \times l_N} \text{ and } \mathcal{B}_k \in \mathbb{R}^{p_1 \times p_2 \times \cdots \times p_N}\}$ and $\mathcal{S}_K = \{\mathcal{W} \in \widehat{\mathcal{S}}_K : \|\mathcal{W}\|_{\mathrm{F}} = 1\}$. Let $\widehat{\mathbf{\Delta}} = \widehat{\mathcal{W}} - \mathcal{W}^*$, and then

$$\frac{1}{n}\sum_{i=1}^{n}(y^i - \langle \mathcal{Z}^i, \widehat{\mathcal{W}}\rangle)^2 \leq \frac{1}{n}\sum_{i=1}^{n}(y^i - \langle \mathcal{Z}^i, \mathcal{W}^*\rangle)^2,$$

which implies that

$$\|\widehat{\mathbf{\Delta}}\|_n^2 \leq \frac{2}{n}\sum_{i=1}^{n}\xi^i\langle \mathcal{Z}^i, \widehat{\mathbf{\Delta}}\rangle \leq 2\|\widehat{\mathbf{\Delta}}\|_{\mathrm{F}} \sup_{\mathbf{\Delta}\in\mathcal{S}_{2K}} \frac{1}{n}\sum_{i=1}^{n}\xi^i\langle \mathcal{Z}^i, \mathbf{\Delta}\rangle, \tag{9}$$

where $\|\mathbf{\Delta}\|_n^2 := n^{-1}\sum_{i=1}^{n}(\langle \mathcal{Z}^i, \mathbf{\Delta}\rangle)^2$ is the empirical norm with respect to $\mathbf{\Delta}$, and $\widehat{\mathbf{\Delta}} \in \widehat{\mathcal{S}}_{2K}$.

Consider a ε-net $\bar{\mathcal{S}}_{2K}$, with the cardinality of $\mathcal{N}(2K, \varepsilon)$, for the set $\mathcal{S}_{2K}$. For any $\mathbf{\Delta} \in \mathcal{S}_{2K}$, there exists a $\bar{\mathbf{\Delta}}_j \in \bar{\mathcal{S}}_{2K}$ such that $\|\mathbf{\Delta} - \bar{\mathbf{\Delta}}_j\|_{\mathrm{F}} \leq \varepsilon$. Note that $\mathbf{\Delta} - \bar{\mathbf{\Delta}}_j \in \widehat{\mathcal{S}}_{4K}$ and, from Lemma 1(a), we can further find $\mathbf{\Delta}_1, \mathbf{\Delta}_2 \in \widehat{\mathcal{S}}_{2K}$ such that $\langle \mathbf{\Delta}_1, \mathbf{\Delta}_2\rangle = 0$ and $\mathbf{\Delta} - \bar{\mathbf{\Delta}}_j = \mathbf{\Delta}_1 + \mathbf{\Delta}_2$. It then holds that $\|\mathbf{\Delta}_1\|_{\mathrm{F}} + \|\mathbf{\Delta}_2\|_{\mathrm{F}} \leq \sqrt{2}\|\mathbf{\Delta} - \bar{\mathbf{\Delta}}_j\|_{\mathrm{F}} \leq \sqrt{2}\varepsilon$ since $\|\mathbf{\Delta} - \bar{\mathbf{\Delta}}_j\|_{\mathrm{F}}^2 = \|\mathbf{\Delta}_1\|_{\mathrm{F}}^2 + \|\mathbf{\Delta}_2\|_{\mathrm{F}}^2$. As a result,

$$\begin{aligned}\frac{1}{n}\sum_{i=1}^{n}\xi^i\langle \mathcal{Z}^i, \mathbf{\Delta}\rangle &= \frac{1}{n}\sum_{i=1}^{n}\xi^i\langle \mathcal{Z}^i, \bar{\mathbf{\Delta}}_j\rangle + \frac{1}{n}\sum_{i=1}^{n}\xi^i\langle \mathcal{Z}^i, \mathbf{\Delta}_1\rangle + \frac{1}{n}\sum_{i=1}^{n}\xi^i\langle \mathcal{Z}^i, \mathbf{\Delta}_2\rangle \\ &\leq \max_{1\leq j\leq \mathcal{N}(2K,\varepsilon)} \frac{1}{n}\sum_{i=1}^{n}\xi^i\langle \mathcal{Z}^i, \bar{\mathbf{\Delta}}_j\rangle + \sqrt{2}\varepsilon \sup_{\mathbf{\Delta}\in\mathcal{S}_{2K}} \frac{1}{n}\sum_{i=1}^{n}\xi^i\langle \mathcal{Z}^i, \mathbf{\Delta}\rangle,\end{aligned}$$

which leads to

$$\sup_{\mathbf{\Delta}\in\mathcal{S}_{2K}} \frac{1}{n}\sum_{i=1}^{n}\xi^i\langle \mathcal{Z}^i, \mathbf{\Delta}\rangle \leq (1-\sqrt{2}\varepsilon)^{-1} \max_{1\leq j\leq \mathcal{N}(2K,\varepsilon)} \frac{1}{n}\sum_{i=1}^{n}\xi^i\langle \mathcal{Z}^i, \bar{\mathbf{\Delta}}_j\rangle. \tag{10}$$

Note that, from Lemma 1(b), $\log \mathcal{N}(2K, \varepsilon) \leq 2d_{\mathcal{M}} \log(9/\varepsilon)$, where $d_{\mathcal{M}} = K(P+L+1)$. Let $\varepsilon = (2\sqrt{2})^{-1}$, and then $8 - 2\log(9/\varepsilon) > 1$. As a result, by (10) and Lemma 3,

$$\begin{aligned}
&\mathbb{P}\left[\sup_{\boldsymbol{\Delta}\in\mathcal{S}_{2K}} \frac{1}{n}\sum_{i=1}^{n} \xi^i \langle \mathcal{Z}^i, \boldsymbol{\Delta}\rangle \geq 8\sigma\sqrt{\alpha_{\mathrm{RSM}}}\left(\sqrt{\frac{d_{\mathcal{M}}}{n}} + \sqrt{\frac{\delta}{n}}\right), \quad \sup_{\boldsymbol{\Delta}\in\mathcal{S}_{2K}} \|\boldsymbol{\Delta}\|_n^2 \leq \alpha_{\mathrm{RSM}}\right] \\
&\leq \mathbb{P}\left[\max_{1\leq j\leq \mathcal{N}(2K,\varepsilon)} \frac{1}{n}\sum_{i=1}^{n} \xi^i \langle \mathcal{Z}^i, \bar{\boldsymbol{\Delta}}_j\rangle \geq 4\sigma\sqrt{\alpha_{\mathrm{RSM}}}\left(\sqrt{\frac{d_{\mathcal{M}}}{n}} + \sqrt{\frac{\delta}{n}}\right), \quad \sup_{\boldsymbol{\Delta}\in\mathcal{S}_{2K}} \|\boldsymbol{\Delta}\|_n^2 \leq \alpha_{\mathrm{RSM}}\right] \\
&\leq \sum_{j=1}^{\mathcal{N}(2K,\varepsilon)} \mathbb{P}\left(\frac{1}{n}\sum_{i=1}^{n} \xi^i \langle \mathcal{Z}^i, \bar{\boldsymbol{\Delta}}_j\rangle \geq 4\sigma\sqrt{\alpha_{\mathrm{RSM}}}\left(\sqrt{\frac{d_{\mathcal{M}}}{n}} + \sqrt{\frac{\delta}{n}}\right), \quad \|\bar{\boldsymbol{\Delta}}_j\|_n^2 \leq \alpha_{\mathrm{RSM}}\right) \\
&\leq \exp\{-d_{\mathcal{M}} - 8\delta\}.
\end{aligned} \tag{11}$$

Note that

$$\begin{aligned}
&\mathbb{P}\left[\sup_{\boldsymbol{\Delta}\in\mathcal{S}_{2K}} \frac{1}{n}\sum_{i=1}^{n} \xi^i \langle \mathcal{Z}^i, \boldsymbol{\Delta}\rangle \geq 8\sigma\sqrt{\alpha_{\mathrm{RSM}}}\left(\sqrt{\frac{d_{\mathcal{M}}}{n}} + \sqrt{\frac{\delta}{n}}\right)\right] \\
&\leq \mathbb{P}\left[\sup_{\boldsymbol{\Delta}\in\mathcal{S}_{2K}} \frac{1}{n}\sum_{i=1}^{n} \xi^i \langle \mathcal{Z}^i, \boldsymbol{\Delta}\rangle \geq 8\sigma\sqrt{\alpha_{\mathrm{RSM}}}\left(\sqrt{\frac{d_{\mathcal{M}}}{n}} + \sqrt{\frac{\delta}{n}}\right), \quad \sup_{\boldsymbol{\Delta}\in\mathcal{S}_{2K}} \|\boldsymbol{\Delta}\|_n^2 \leq \alpha_{\mathrm{RSM}}\right] \\
&\qquad + \mathbb{P}\left(\sup_{\boldsymbol{\Delta}\in\mathcal{S}_{2K}} \|\boldsymbol{\Delta}\|_n^2 \geq \alpha_{\mathrm{RSM}}\right).
\end{aligned}$$

From (11) and Lemma 2, we then have that, with probability $1 - 4\exp\{-[c_{\mathrm{H}}(0.25\kappa_L/\kappa_U)^2 n - 9d_{\mathcal{M}}]\} - \exp\{-d_{\mathcal{M}} - 8\delta\}$,

$$\sup_{\boldsymbol{\Delta}\in\mathcal{S}_{2K}} \frac{1}{n}\sum_{i=1}^{n} \xi^i \langle \mathcal{Z}^i, \boldsymbol{\Delta}\rangle \leq 8\sigma\sqrt{\alpha_{\mathrm{RSM}}}\left(\sqrt{\frac{d_{\mathcal{M}}}{n}} + \sqrt{\frac{\delta}{n}}\right) \quad \text{and} \quad \|\boldsymbol{\Delta}\|_n^2 \geq \alpha_{\mathrm{RSC}} \|\boldsymbol{\Delta}\|_{\mathrm{F}}^2,$$

which, together with (9), leads to

$$\|\widehat{\boldsymbol{\Delta}}\|_{\mathrm{F}} \leq \frac{16\sigma\sqrt{\alpha_{\mathrm{RSM}}}}{\alpha_{\mathrm{RSC}}}\left(\sqrt{\frac{d_{\mathcal{M}}}{n}} + \sqrt{\frac{\delta}{n}}\right) \quad \text{and} \quad \|\widehat{\boldsymbol{\Delta}}\|_n \leq 16\sigma\frac{\alpha_{\mathrm{RSM}}}{\alpha_{\mathrm{RSC}}}\left(\sqrt{\frac{d_{\mathcal{M}}}{n}} + \sqrt{\frac{\delta}{n}}\right).$$

Finally we prove (c). Given a test sample $(\mathcal{X}, y)$, and let $\mathcal{Z} = \mathcal{X} \times_1 \boldsymbol{U}_{\mathcal{F}}^{(1)\prime} \times_2 \boldsymbol{U}_{\mathcal{F}}^{(2)\prime} \times_3 \cdots \times_N \boldsymbol{U}_{\mathcal{F}}^{(N)\prime}$. It holds that

$$\mathbb{E}(y - \langle \mathcal{X}, \mathcal{W}_X\rangle)^2 = \mathbb{E}(y - \langle \mathcal{Z}, \mathcal{W}\rangle)^2 = \mathbb{E}(\langle \mathcal{Z}, \boldsymbol{\Delta}\rangle)^2 + \sigma^2,$$

and

$$\mathbb{E}(\langle \mathcal{Z}, \boldsymbol{\Delta}\rangle)^2 = \boldsymbol{\Delta}'\mathbb{E}(\bar{\boldsymbol{z}}\bar{\boldsymbol{z}}')\boldsymbol{\Delta} = \boldsymbol{\Delta}'\boldsymbol{U}_G\mathbb{E}(\bar{\boldsymbol{x}}\bar{\boldsymbol{x}}')\boldsymbol{U}_G'\boldsymbol{\Delta} \leq \kappa_U\|\boldsymbol{\Delta}\|_2^2.$$

As a result, from (a) of this theorem, we have

$$\mathrm{err}^2(\widehat{\mathcal{W}}) \leq \left(\frac{16\sigma\alpha_{\mathrm{RSM}}}{\alpha_{\mathrm{RSC}}}\right)^2 \left(\frac{d_{\mathcal{M}}}{n} + \frac{\delta}{n}\right) + \sigma^2.$$

This accomplishes the proof.

A.3.2 Proof of Theorem 2

Denote the sets $\widehat{\mathcal{S}}^{\mathrm{TU}}(R_1, \cdots, R_{N+1}) = \{\sum_{k=1}^{K} \mathcal{B}_k \otimes \mathcal{A}_k$: the stacked kernel $\mathcal{A}_{\mathrm{stack}} \in \mathbb{R}^{l_1\times l_2\times\cdots\times l_N\times K}$ has the ranks of $(R_1, ..., R_N, R_{N+1})$ and $\mathcal{B}_k \in \mathbb{R}^{p_1\times p_2\times\cdots\times p_N}\}$, $\widehat{\mathcal{S}}_{2K}^{\mathrm{TU}} = \{\mathcal{W}_1 + \mathcal{W}_2 : \mathcal{W}_1, \mathcal{W}_2 \in \widehat{\mathcal{S}}^{\mathrm{TU}}(R_1, \cdots, R_{N+1})\}$, and $\mathcal{S}_{2K}^{\mathrm{TU}} = \{\mathcal{W} \in \widehat{\mathcal{S}}_{2K}^{\mathrm{TU}} : \|\mathcal{W}\|_{\mathrm{F}} = 1\}$. Note that $\widehat{\mathcal{W}}_{\mathrm{TU}}, \mathcal{W}^* \in \widehat{\mathcal{S}}^{\mathrm{TU}}(R_1, \cdots, R_{N+1})$, and $\widehat{\boldsymbol{\Delta}} = \widehat{\mathcal{W}}_{\mathrm{TU}} - \mathcal{W}^* \in \widehat{\mathcal{S}}_{2K}^{\mathrm{TU}}$.

We first consider a ε-net for $\mathcal{S}_{2K}^{\mathrm{TU}}$. For each $1 \le k \le K$, Let $\boldsymbol{b}_k = (b_{k1}, \ldots, b_{kP})' = \mathrm{vec}(\mathcal{B}_k)$, and we can rearrange $\mathcal{B}_k \otimes \mathcal{A}_k$ into the form of $\mathcal{A}_k \circ \boldsymbol{b}_k$, which is a tensor of size $\mathbb{R}^{l_1 \times l_2 \times \cdots \times l_N \times P}$, where $P = p_1 p_2 \cdots p_N$. Denote $\boldsymbol{B} = (b_{kj}) \in \mathbb{R}^{P \times K}$, and it holds that

$$\sum_{k=1}^{K} \mathcal{A}_k \circ \boldsymbol{b}_k = \mathcal{A} \times_{N+1} \boldsymbol{B} = \mathcal{G} \times_1 \boldsymbol{H}^{(1)} \times_2 \boldsymbol{H}^{(2)} \cdots \times_{N+1} \widetilde{\boldsymbol{B}},$$

which is a tensor with the size of $\mathbb{R}^{l_1 \times l_2 \times \cdots \times l_N \times P}$ and the ranks of $(R_1, ..., R_N, R_{N+1})$ where $\widetilde{\boldsymbol{B}} = \boldsymbol{B}\boldsymbol{H}^{(N+1)} \in \mathbb{R}^{P \times R_{N+1}}$. Essentially, in this step, we rewrite the model into one with R_{k+1} kernels instead. Specifically, we now have $\mathcal{W} = \sum_{r=1}^{R_{N+1}} \widetilde{\mathcal{A}}_r \otimes \widetilde{\mathcal{B}}_r$, where $\widetilde{\mathcal{A}}_r = \mathcal{G}_r \times_1 \boldsymbol{H}^{(1)} \times_2 \boldsymbol{H}^{(2)} \cdots \times_N \boldsymbol{H}^{(N)}$ with $\mathcal{G}_r = \mathcal{G}(:, :, \cdots, r)$ and the r-th column of $\widetilde{\boldsymbol{B}}$ is the vectorization of $\widetilde{\mathcal{B}}_r$ for $1 \le r \le R_{N+1}$.

As a result, $\widehat{\mathcal{S}}_{2K}^{\mathrm{TU}}$ consists of tensors with the ranks of $(2R_1, ..., 2R_N, 2R_{N+1})$ at most.

Denote $\mathcal{S}_{\mathrm{Tucker}}(r_1, \cdots, r_N, r_{N+1}) = \{\mathcal{T} \in \mathbb{R}^{l_1 \times \cdots \times l_N \times l_{N+1}} : \|\mathcal{T}\|_{\mathrm{F}} = 1, \mathcal{T}$ has the Tucker ranks of $(r_1, \cdots, r_N, r_{N+1})\}$, where $l_{N+1} = P = p_1 p_2 \cdots p_N$. Then the ε-covering number for $\mathcal{S}_{2K}^{\mathrm{TU}}$ satisfies

$$|\mathcal{S}_{2K}^{\mathrm{TU}}| \le |\mathcal{S}_{\mathrm{Tucker}}(2R_1, \cdots, 2R_N, 2R_{N+1})|.$$

For each $\mathcal{T} \in \mathcal{S}_{\mathrm{Tucker}}(r_1, \cdots, r_N, r_{N+1})$, we have

$$\mathcal{T} = \mathcal{G} \times_1 \boldsymbol{U}^{(1)} \times_2 \cdots \times_{N+1} \boldsymbol{U}^{(N+1)},$$

where $\mathcal{G} \in \mathbb{R}^{r_1 \times \cdots \times r_N \times r_{N+1}}$ with $\|\mathcal{G}\|_{\mathrm{F}} = 1$, and $\boldsymbol{U}^{(i)} \in \mathbb{R}^{l_i \times r_i}$ with $1 \le i \le N+1$ are orthonormal matrices. We now construct an ε-net for $\mathcal{S}_{\mathrm{Tucker}}(r_1, \cdots, r_N, r_{N+1})$ by covering the sets of $\mathcal{G}$ and all $\boldsymbol{U}^{(i)}$s, and the proof hinges on the covering number of low-multilinear-rank tensors in Wang et al. (2019). Treating $\mathcal{G}$ as $\prod_{j=1}^{N+1} r_j$-dimensional vector with $\|\mathcal{G}\|_{\mathrm{F}} = 1$, we can find an $\varepsilon/(N+2)$-net for it, denoted by $\bar{G}$, with the cardinality of $|\bar{G}| \le (3(N+2)/\varepsilon)^{\prod_{j=1}^{N+1} r_j}$.

Next, let $O_{n,r} = \{\boldsymbol{U} \in \mathbb{R}^{n \times r} : \boldsymbol{U}^\top \boldsymbol{U} = I_r\}$. To cover $O_{n,r}$, it is beneficial to use the $\|\cdot\|_{1,2}$ norm, defined as

$$\|\boldsymbol{X}\|_{1,2} = \max_i \|\boldsymbol{X}_i\|_2,$$

where $\boldsymbol{X}_i$ denotes the ith column of $\boldsymbol{X}$. Let $Q_{n,r} = \{\boldsymbol{X} \in \mathbb{R}^{n \times r} : \|\boldsymbol{X}\|_{1,2} \le 1\}$. One can easily check that $O_{n,r} \subset Q_{n,r}$, and then an $\varepsilon/(N+2)$-net $\bar{O}_{n,r}$ for $O_{n,r}$ has the cardinality of $|\bar{O}_{n,r}| \le (3(N+2)/\varepsilon)^{nr}$. Denote $\bar{\mathcal{S}}_{\mathrm{Tucker}}(r_1, \cdots, r_N, r_{N+1}) = \{\mathcal{G} \times_1 \boldsymbol{U}^{(1)} \times_2 \cdots \times_{N+1} \boldsymbol{U}^{(N+1)} : \mathcal{G} \in \bar{G}, \boldsymbol{U}^{(i)} \in \bar{O}_{l_i, r_i}, 1 \le i \le N+1\}$. By a similar argument presented in Lemma A.1 of Wang et al. (2019), we can show that $\bar{\mathcal{S}}_{\mathrm{Tucker}}(r_1, \cdots, r_N, r_{N+1})$ is a ε-net for the set $\mathcal{S}_{\mathrm{Tucker}}(r_1, \cdots, r_N, r_{N+1})$ with the cardinality of

$$\left(\frac{3N+6}{\varepsilon}\right)^{\prod_{j=1}^{N+1} r_j + \sum_{j=1}^{N+1} l_j r_j}.$$

where $l_{N+1} = P$. Thus, the ε-covering number of $\mathcal{S}_{2K}^{\mathrm{TU}}$ is

$$\mathcal{N}^{\mathrm{TU}}(\varepsilon) = |\mathcal{S}_{\mathrm{Tucker}}(2R_1, \cdots, 2R_N, 2R_{N+1})| \le \left(\frac{3N+6}{\varepsilon}\right)^{2^{N+1} \prod_{j=1}^{N+1} R_j + 2\sum_{i=1}^{N} l_i R_i + 2R_{N+1}P}.$$

Let $\varepsilon = 1/2$. It then holds that $\log(3(N+2)/\varepsilon) < 3\log(N+2)$ and $\log \mathcal{N}^{\mathrm{TU}}(\varepsilon) \le 2^{N+1} d_{\mathcal{M}}^{\mathrm{TU}} \log[(3N+6)/\varepsilon]$, where $d_{\mathcal{M}}^{\mathrm{TU}} = \prod_{j=1}^{N+1} R_j + \sum_{i=1}^{N} l_i R_i + R_{N+1}P$. By a method similar to the proof of Lemma 2, we can show that

$$\mathbb{P}\left\{\sup_{\boldsymbol{\Delta} \in \mathcal{S}_{2K}^{\mathrm{TU}}} \|\boldsymbol{\Delta}\|_n^2 \ge \alpha_{\mathrm{RSM}}\right\} \le 2\exp\left\{-c_{\mathrm{H}} n \left(\frac{\kappa_L}{4\kappa_U}\right)^2 + 3c_N d_{\mathcal{M}}^{\mathrm{TU}}\right\}, \tag{12}$$

and

$$\mathbb{P}\left\{\inf_{\boldsymbol{\Delta} \in \mathcal{S}_{2K}^{\mathrm{TU}}} \|\boldsymbol{\Delta}\|_n^2 \le \alpha_{\mathrm{RSC}}\right\} \le 2\exp\left\{-c_{\mathrm{H}} n \left(\frac{\kappa_L}{4\kappa_U}\right)^2 + 3c_N d_{\mathcal{M}}^{\mathrm{TU}}\right\}. \tag{13}$$

where $c_N = 2^{N+1}\log(N+2)$. Moreover, similar to equation 11, by applying Lemma 3, we can show that

$$\begin{aligned}
&\mathbb{P}\left[\sup_{\boldsymbol{\Delta}\in\mathcal{S}_{2K}^{\mathrm{TU}}}\frac{1}{n}\sum_{i=1}^{n}\xi^i\langle\mathfrak{Z}^i,\boldsymbol{\Delta}\rangle \geq 8\sigma\sqrt{\alpha_{\mathrm{RSM}}}\left(\sqrt{\frac{c_N d_{\mathcal{M}}^{\mathrm{TU}}}{n}}+\sqrt{\frac{\delta}{n}}\right),\ \sup_{\boldsymbol{\Delta}\in\mathcal{S}_{2K}^{\mathrm{TU}}}\|\boldsymbol{\Delta}\|_n^2\leq\alpha_{\mathrm{RSM}}\right]\\
&\leq \mathbb{P}\left[\max_{1\leq j\leq\mathcal{N}^{\mathrm{TU}}(\varepsilon)}\frac{1}{n}\sum_{i=1}^{n}\xi^i\langle\mathfrak{Z}^i,\bar{\boldsymbol{\Delta}}_j\rangle \geq 4\sigma\sqrt{\alpha_{\mathrm{RSM}}}\left(\sqrt{\frac{c_N d_{\mathcal{M}}^{\mathrm{TU}}}{n}}+\sqrt{\frac{\delta}{n}}\right),\ \sup_{\boldsymbol{\Delta}\in\mathcal{S}_{2K}^{\mathrm{TU}}}\|\boldsymbol{\Delta}\|_n^2\leq\alpha_{\mathrm{RSM}}\right]\\
&\leq \sum_{j=1}^{\mathcal{N}^{\mathrm{TU}}(\varepsilon)}\mathbb{P}\left(\frac{1}{n}\sum_{i=1}^{n}\xi^i\langle\mathfrak{Z}^i,\bar{\boldsymbol{\Delta}}_j\rangle \geq 4\sigma\sqrt{\alpha_{\mathrm{RSM}}}\left(\sqrt{\frac{c_N d_{\mathcal{M}}^{\mathrm{TU}}}{n}}+\sqrt{\frac{\delta}{n}}\right),\ \|\bar{\boldsymbol{\Delta}}_j\|_n^2\leq\alpha_{\mathrm{RSM}}\right)\\
&\leq \exp\{-d_{\mathcal{M}}^{\mathrm{TU}}-8\delta\},
\end{aligned}\tag{14}$$

where $\varepsilon = (2\sqrt{2})^{-1}$ and $2^{N+4}\log(N+2) - 2^{N+1}\log(3(N+1)/\varepsilon) > 1$. By a method similar to the proof of Theorem 1, we can show that, with probability $1 - 4\exp\{-[c_{\mathrm{H}}(0.25\kappa_L/\kappa_U)^2 n - 3c_N\log(N+2)d_{\mathcal{M}}^{\mathrm{TU}}]\} - \exp\{-d_{\mathcal{M}}^{\mathrm{TU}} - 8\delta\}$,

$$\sup_{\boldsymbol{\Delta}\in\mathcal{S}_{2K}^{\mathrm{TU}}}\frac{1}{n}\sum_{i=1}^{n}\xi^i\langle\mathfrak{Z}^i,\boldsymbol{\Delta}\rangle \leq 8\sigma\sqrt{\alpha_{\mathrm{RSM}}}\left(\sqrt{\frac{c_N d_{\mathcal{M}}^{\mathrm{TU}}}{n}}+\sqrt{\frac{\delta}{n}}\right)\quad\text{and}\quad\|\boldsymbol{\Delta}\|_n^2\geq\alpha_{\mathrm{RSC}}\|\boldsymbol{\Delta}\|_{\mathrm{F}}^2,$$

which, together with equation 9, leads to

$$\begin{aligned}
\|\widehat{\boldsymbol{\Delta}}\|_{\mathrm{F}} &\leq \frac{16\sigma\sqrt{\alpha_{\mathrm{RSM}}}}{\alpha_{\mathrm{RSC}}}\left(\sqrt{\frac{c_N d_{\mathcal{M}}^{\mathrm{TU}}}{n}}+\sqrt{\frac{\delta}{n}}\right),\\
\|\widehat{\boldsymbol{\Delta}}\|_n &\leq 16\sigma\frac{\alpha_{\mathrm{RSM}}}{\alpha_{\mathrm{RSC}}}\left(\sqrt{\frac{c_N d_{\mathcal{M}}^{\mathrm{TU}}}{n}}+\sqrt{\frac{\delta}{n}}\right),\\
\mathrm{err}^2(\widehat{\boldsymbol{\mathcal{W}}}_{\mathrm{TU}}) &\leq \left(\frac{16\sigma\alpha_{\mathrm{RSM}}}{\alpha_{\mathrm{RSC}}}\right)^2\left(\frac{c_N d_{\mathcal{M}}^{\mathrm{TU}}}{n}+\frac{\delta}{n}\right)+\sigma^2.
\end{aligned}$$

We accomplished the proof.

A.3.3 Proofs of Corollary 1

When the kernel tensor $\boldsymbol{\mathcal{A}}$ has a Tucker decomposition form of $\boldsymbol{\mathcal{A}} = \mathcal{G}\times_1\boldsymbol{H}^{(1)}\times_2\boldsymbol{H}^{(2)}\times_3\cdots\times_{N+1}\boldsymbol{H}^{(N+1)}$, and the multilinear rank is $(R_1, R_2, \cdots, R_{N+1})$. Let $\mathcal{H} = \mathcal{G}\times_1\boldsymbol{H}^{(1)}\times_2\boldsymbol{H}^{(2)}\times_3\cdots\times_N\boldsymbol{H}^{(N)}$, which is a tensor of size $\mathbb{R}^{l_1\times\cdots\times l_N\times R_{N+1}}$. The mode-$(N+1)$ unfolding of $\mathcal{H}$ is a matrix with R_{N+1} row vectors, each of size $\mathbb{R}^L$, and we denote them by $\boldsymbol{g}_1, \cdots, \boldsymbol{g}_{R_{N+1}}$. Fold $\boldsymbol{g}_r$s back into tensors, i.e. $\boldsymbol{g}_r = \mathrm{vec}(\widetilde{\mathcal{H}}_r)$, where $\widetilde{\mathcal{H}}_r \in \mathbb{R}^{l_1\times\cdots\times l_N}$ and $1\leq r\leq R_{N+1}$. Moreover, let $\boldsymbol{H}^{(N+1)} = (\boldsymbol{h}_1^{(N+1)}, \cdots, \boldsymbol{h}_K^{(N+1)})'$, where $\boldsymbol{h}_k^{(N+1)}$ is a vector of size $\mathbb{R}^{R_{N+1}}$ and we denote its rth entry as $h_{k,r}^{(N+1)}$, for $1\leq r\leq R_{N+1}$. It can be verified that, for each $1\leq k\leq K$,

$$\boldsymbol{\mathcal{A}}_k = \mathcal{G}\times_1\boldsymbol{H}^{(1)}\times_2\boldsymbol{H}^{(2)}\cdots\times_N\boldsymbol{H}^{(N)}\bar{\times}_{N+1}\boldsymbol{h}_k^{(N+1)} = \mathcal{H}\bar{\times}_{N+1}\boldsymbol{h}_k^{(N+1)},$$

and hence,

$$\sum_{k=1}^{K}\mathcal{B}_k\otimes\boldsymbol{\mathcal{A}}_k = \sum_{r=1}^{R_{N+1}}\sum_{k=1}^{K}\mathcal{B}_k\otimes\left(\widetilde{\mathcal{H}}_r\cdot h_{k,r}^{(N+1)}\right) = \sum_{r=1}^{R_{N+1}}\left(\sum_{k=1}^{K}h_{k,r}^{(N+1)}\mathcal{B}_k\right)\otimes\widetilde{\mathcal{H}}_r.$$

By letting $\widetilde{\mathcal{B}}_r = \sum_{k=1}^{K}h_{k,r}^{(N+1)}\mathcal{B}_k$ and $\widetilde{\boldsymbol{\mathcal{A}}}_r = \widetilde{\mathcal{H}}_r$ for $1\leq r\leq R$, we can reformulate the model into

$$y^i = \langle\sum_{r=1}^{R}\widetilde{\mathcal{B}}_r\otimes\widetilde{\boldsymbol{\mathcal{A}}}_r, \mathfrak{Z}^i\rangle + \xi^i,$$

and the proof of (a) is then accomplished.

Suppose that the kernel tensor $\mathcal{A}$ has a CP decomposition form of $\mathcal{A} = \sum_{r=1}^{R} \alpha_r \boldsymbol{h}_r^{(1)} \circ \boldsymbol{h}_r^{(2)} \circ \cdots \circ \boldsymbol{h}_r^{(N+1)}$, where $\boldsymbol{h}_r^{(j)} \in \mathbb{R}^{l_j}$ and $\|\boldsymbol{h}_r^{(j)}\| = 1$ for all $1 \leq j \leq N$. Moreover, $\boldsymbol{h}_r^{(N+1)} = (h_{r,1}^{(N+1)}, ..., h_{r,K}^{(N+1)})' \in \mathbb{R}^K$ and $\|\boldsymbol{h}_r^{(N+1)}\| = 1$. Note that

$$\mathcal{A}_k = \sum_{r=1}^{R} \alpha_r h_{r,k}^{(N+1)} \boldsymbol{h}_r^{(1)} \circ \boldsymbol{h}_r^{(2)} \circ \cdots \circ \boldsymbol{h}_r^{(N)},$$

for all $1 \leq k \leq K$, and hence

$$\begin{aligned}\sum_{k=1}^{K} \mathcal{B}_k \otimes \mathcal{A}_k &= \sum_{r=1}^{R} \sum_{k=1}^{K} \mathcal{B}_k \otimes \left(\alpha_r h_{r,k}^{(N+1)} \boldsymbol{h}_r^{(1)} \circ \boldsymbol{h}_r^{(2)} \circ \cdots \circ \boldsymbol{h}_r^{(N)} \right) \\ &= \sum_{r=1}^{R} \left(h_{r,k}^{(N+1)} \mathcal{B}_k \right) \otimes \left(\alpha_r \boldsymbol{h}_r^{(1)} \circ \boldsymbol{h}_r^{(2)} \circ \cdots \circ \boldsymbol{h}_r^{(N)} \right).\end{aligned}$$

By letting $\widetilde{\mathcal{B}}_r = h_{r,k}^{(N+1)} \mathcal{B}_k$ and $\widetilde{\mathcal{A}}_r = \alpha_r \boldsymbol{h}_r^{(1)} \circ \boldsymbol{h}_r^{(2)} \circ \cdots \circ \boldsymbol{h}_r^{(N)}$ for all $1 \leq r \leq R$, we can reparameterize the model into

$$y^i = \langle \sum_{r=1}^{R} \widetilde{\mathcal{B}}_r \otimes \widetilde{\mathcal{A}}_r, \mathcal{Z}^i \rangle + \xi^i,$$

and the proof of (b) is then accomplished.

A.3.4 Lemmas Used in the Proofs of Theorems

Lemma 1. *(Partition and covering number of restricted spaces) Consider* $\widehat{\mathcal{S}}_K = \{\sum_{k=1}^{K} \mathcal{B}_k \otimes \mathcal{A}_k : \mathcal{A}_k \in \mathbb{R}^{l_1 \times l_2 \times \cdots \times l_N}$ *and* $\mathcal{B}_k \in \mathbb{R}^{p_1 \times p_2 \times \cdots \times p_N}\}$ *and* $\mathcal{S}_K = \{\mathcal{W} \in \widehat{\mathcal{S}}_K : \|\mathcal{W}\|_{\mathrm{F}} = 1\}$.

(a) For any $\boldsymbol{\Delta} \in \widehat{\mathcal{S}}_{2K}$*, there exist* $\mathcal{W}_1, \mathcal{W}_2 \in \widehat{\mathcal{S}}_K$ *such that* $\boldsymbol{\Delta} = \mathcal{W}_1 + \mathcal{W}_2$ *and* $\langle \mathcal{W}_1, \mathcal{W}_2 \rangle = 0$.

(b) The ε*-covering number of the set* $\mathcal{S}_K$ *is*

$$\mathcal{N}(K, \varepsilon) \leq (9/\varepsilon)^{K(L+P+1)},$$

where $L = l_1 l_2 \cdots l_N$ *and* $P = p_1 p_2 \cdots p_N$.

Proof. For each K, we first define a map $T_K : \widehat{\mathcal{S}}_K \to \mathbb{R}^{P \times L}$. For any $\mathcal{W} = \sum_{k=1}^{K} \mathcal{B}_k \otimes \mathcal{A}_k \in \widehat{\mathcal{S}}_K$, we define that $T_K(\mathcal{W}) = \sum_{k=1}^{K} \mathrm{vec}\,(\mathcal{B}_k)\, \mathrm{vec}(\mathcal{A}_k)' \in \mathbb{R}^{P \times L}$, which has the rank of at most K.

For any $\boldsymbol{\Delta} \in \widehat{\mathcal{S}}_{2K}$, the rank of matrix $T_K(\boldsymbol{\Delta})$ is at most $2K$. As shown in Figure 6, we can split the singular value decomposition (SVD) of $T_K(\boldsymbol{\Delta})$ into two parts, and it can be verified that $T_K(\boldsymbol{\Delta}) = \boldsymbol{U}_1 \boldsymbol{\Lambda}_1 \boldsymbol{V}_1{}' + \boldsymbol{U}_2 \boldsymbol{\Lambda}_2 \boldsymbol{V}_2{}'$ and $\langle \boldsymbol{U}_1 \boldsymbol{\Lambda}_1 \boldsymbol{V}_1{}', \boldsymbol{U}_2 \boldsymbol{\Lambda}_2 \boldsymbol{V}_2{}' \rangle = 0$. Denote $\boldsymbol{U}_i = (\boldsymbol{u}_1^{(i)}, \boldsymbol{u}_2^{(i)}, \ldots, \boldsymbol{u}_K^{(i)})$ and $\boldsymbol{V}_i \boldsymbol{\Lambda}_i' = (\boldsymbol{v}_1^{(i)}, \boldsymbol{v}_2^{(i)}, \ldots, \boldsymbol{v}_K^{(i)})$ for $i = 1$ and 2. We then fold $\boldsymbol{u}_j^{(i)}$s and $\boldsymbol{v}_j^{(i)}$s into tensors, i.e. $\boldsymbol{u}_k^{(i)} = \mathrm{vec}(\mathcal{B}_k^{(i)})$ and $\boldsymbol{v}_k^{(i)} = \mathrm{vec}(\mathcal{A}_k^{(i)})$ with $1 \leq k \leq K$. Let $\mathcal{W}_1 = \sum_{k=1}^{K} \mathcal{B}_k^{(1)} \otimes \mathcal{A}_k^{(1)}$ and $\mathcal{W}_2 = \sum_{k=1}^{K} \mathcal{B}_k^{(2)} \otimes \mathcal{A}_k^{(2)}$. It then can be verified that $\mathcal{W}_1, \mathcal{W}_2 \in \widehat{\mathcal{S}}_K$, $\boldsymbol{\Delta} = \mathcal{W}_1 + \mathcal{W}_2$ and $\langle \mathcal{W}_1, \mathcal{W}_2 \rangle = 0$. Thus, we accomplish the proof of (a).

Denote by $\mathcal{S}_{\mathrm{matrix}} \subset \mathbb{R}^{P \times L}$ the set of matrices with unit Frobenius norm and rank at most K. Note that $T(\mathcal{S}_K) \subset \mathcal{S}_{\mathrm{matrix}}$, while the ε-covering number for $\mathcal{S}_{\mathrm{matrix}}$ is

$$|\mathcal{S}_{\mathrm{matrix}}| \leq (9/\varepsilon)^{K(L+P+1)};$$

see Candes & Plan (2011). This accomplishes the proof of (b).

□

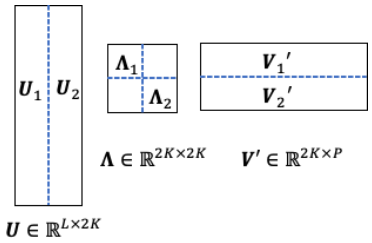

Figure 6: Splitting matrix $T(\boldsymbol{\Delta})$ based on its singular value decomposition.

Lemma 2. *(Restricted strong convexity and smoothness) Suppose that $n \gtrsim (\kappa_U/\kappa_L)^2 d_{\mathcal{M}}$. It holds that, with probability at least $1 - 4\exp\{-[c_{\mathrm{H}}(0.25\kappa_L/\kappa_U)^2 n - 9d_{\mathcal{M}}]\}$,*

$$\alpha_{\mathrm{RSC}} \|\boldsymbol{\Delta}\|_{\mathrm{F}}^2 \leq \frac{1}{n}\sum_{i=1}^{n}(\langle \mathfrak{Z}^i, \boldsymbol{\Delta}\rangle)^2 \leq \alpha_{\mathrm{RSM}} \|\boldsymbol{\Delta}\|_{\mathrm{F}}^2,$$

for all $\boldsymbol{\Delta} \in \widehat{\mathcal{S}}_{2K}$, where $\widehat{\mathcal{S}}_{2K}$ is defined in Lemma 1, c_{H} is a positive constant from the Hanson-Wright inequality, $\alpha_{\mathrm{RSC}} = \kappa_L/2$ and $\alpha_{\mathrm{RSM}} = 3\kappa_U/2$.

Proof. It is sufficient to show that $\sup_{\boldsymbol{\Delta}\in\mathcal{S}_{2K}} \|\boldsymbol{\Delta}\|_n^2 \leq \alpha_{\mathrm{RSM}}$ and $\inf_{\boldsymbol{\Delta}\in\mathcal{S}_{2K}} \|\boldsymbol{\Delta}\|_n^2 \geq \alpha_{\mathrm{RSC}}$ hold with a high probability, where $\|\boldsymbol{\Delta}\|_n^2 := n^{-1}\sum_{i=1}^n(\langle \mathfrak{Z}^i, \boldsymbol{\Delta}\rangle)^2$ is the empirical norm, and $\mathcal{S}_{2K}$ is defined in Lemma 1. Without confusion, in this proof, we will also use the notation of $\boldsymbol{\Delta}$ for its vectorized version, $\mathrm{vec}(\boldsymbol{\Delta})$. Note that

$$\begin{aligned}\|\boldsymbol{\Delta}\|_n^2 &= \frac{1}{n}\sum_{i=1}^{n}(\langle \mathfrak{Z}^i, \boldsymbol{\Delta}\rangle)^2 = \frac{1}{n}\sum_{i=1}^{n}\boldsymbol{z}^{i\prime}\boldsymbol{\Delta}\boldsymbol{\Delta}'\boldsymbol{z}^i = \frac{1}{n}\sum_{i=1}^{n}\boldsymbol{x}^{i\prime}\boldsymbol{U}_G\boldsymbol{\Delta}\boldsymbol{\Delta}'\boldsymbol{U}_G'\boldsymbol{x}^i \\ &= \frac{1}{n}\bar{\boldsymbol{x}}'\Big[\boldsymbol{I}_n \otimes (\boldsymbol{U}_G\boldsymbol{\Delta}\boldsymbol{\Delta}'\boldsymbol{U}_G')\Big]\bar{\boldsymbol{x}} = \bar{\boldsymbol{w}}'\boldsymbol{Q}\bar{\boldsymbol{w}},\end{aligned} \tag{15}$$

where $\boldsymbol{Q} = n^{-1}\boldsymbol{\Sigma}^{\frac{1}{2}}[\boldsymbol{I}_n \otimes (\boldsymbol{U}_G\boldsymbol{\Delta}\boldsymbol{\Delta}'\boldsymbol{U}_G')]\boldsymbol{\Sigma}^{\frac{1}{2}}$, $\bar{\boldsymbol{w}} \in \mathbb{R}^{nD}$ is a random vector with $i.i.d.$ standard normal entries, $\boldsymbol{z}^i = \boldsymbol{U}_G'\boldsymbol{x}^i$, $\boldsymbol{x}^i = \mathrm{vec}(\boldsymbol{\mathcal{X}}^i)$, $\bar{\boldsymbol{x}} = (\boldsymbol{x}^{1\prime}, \boldsymbol{x}^{2\prime}, \ldots, \boldsymbol{x}^{n\prime})'$, and $\boldsymbol{\Sigma} = \mathbb{E}(\bar{\boldsymbol{x}}\bar{\boldsymbol{x}}') \in \mathbb{R}^{nD\times nD}$.

Denote $\boldsymbol{w}_G = \boldsymbol{U}_G\boldsymbol{\Delta} \in \mathbb{R}^D$ and $\mathcal{S}^{n-1} = \{\boldsymbol{v} \in \mathbb{R}^n : \|\boldsymbol{v}\|_2 = 1\}$. Let $\lambda_{\max}(\boldsymbol{\Sigma}_{ii})$ be the maximum eigenvalue of $\boldsymbol{\Sigma}_{ii} = \mathbb{E}(\mathrm{vec}\,(\boldsymbol{\mathcal{X}}^i)\,\mathrm{vec}\,(\boldsymbol{\mathcal{X}}^i)')$ for $1 \leq i \leq n$, and it holds that $\lambda_{\max}(\boldsymbol{\Sigma}_{ii}) \leq C_x$ for all $1 \leq i \leq n$. For matrix $\boldsymbol{Q}$, we have

$$\begin{aligned}\mathrm{tr}(\boldsymbol{Q}) &= \frac{1}{n}\,\mathrm{tr}\left(\boldsymbol{\Sigma}^{\frac{1}{2}}[\boldsymbol{I}_n \otimes \boldsymbol{w}_G][\boldsymbol{I}_n \otimes \boldsymbol{w}_G']\boldsymbol{\Sigma}^{\frac{1}{2}}\right) = \frac{1}{n}\,\mathrm{tr}\left([\boldsymbol{I}_n \otimes \boldsymbol{w}_G']\boldsymbol{\Sigma}[\boldsymbol{I}_n \otimes \boldsymbol{w}_G]\right) = \frac{1}{n}\sum_{i=1}^{n}(\boldsymbol{w}_G'\boldsymbol{\Sigma}_{ii}\boldsymbol{w}_G) \\ &\leq \frac{1}{n}\sum_{i=1}^{n}\left(\sup_{\boldsymbol{w}_G\in\mathcal{S}^{PL-1}}\frac{\boldsymbol{w}_G'\boldsymbol{\Sigma}_{ii}\boldsymbol{w}_G}{\boldsymbol{w}_G'\boldsymbol{w}_G}\right)\left(\sup_{\boldsymbol{\Delta}\in\mathcal{S}^{PL-1}}\boldsymbol{\Delta}'\boldsymbol{U}_G'\boldsymbol{U}_G\boldsymbol{\Delta}\right) \leq \frac{1}{n}\sum_{i=1}^{n}\lambda_{\max}(\boldsymbol{\Sigma}_{ij})C_u \leq \kappa_U,\end{aligned} \tag{16}$$

and similarly, we can show that $\mathrm{tr}(\boldsymbol{Q}) \geq \kappa_L$, where $\kappa_L = c_x c_u$, $\kappa_U = C_x C_u$ and $\kappa_L \leq \kappa_U$. Thus,

$$\kappa_L \leq \mathbb{E}\,\|\boldsymbol{\Delta}\|_n^2 = \mathrm{tr}(\boldsymbol{Q}) \leq \kappa_U. \tag{17}$$

Moreover, denote $\boldsymbol{Q}_1 = n^{-\frac{1}{2}}\boldsymbol{\Sigma}^{\frac{1}{2}}[\boldsymbol{I}_n \otimes \boldsymbol{w}_G]$ and note that $\boldsymbol{Q} = \boldsymbol{Q}_1\boldsymbol{Q}_1'$. To bound the operator norm of $\boldsymbol{Q}$, we have

$$\begin{aligned}\|\boldsymbol{Q}\|_{\mathrm{op}} \leq \|\boldsymbol{Q}_1\|_{\mathrm{op}}^2 &= \sup_{\boldsymbol{u}\in\mathbb{S}^{n-1}}\boldsymbol{u}'\boldsymbol{Q}_1'\boldsymbol{Q}_1\boldsymbol{u} = \frac{1}{n}\sup_{\boldsymbol{u}\in\mathbb{S}^{n-1}}\boldsymbol{u}'[\boldsymbol{I}_n \otimes \boldsymbol{w}_G']\boldsymbol{\Sigma}[\boldsymbol{I}_n \otimes \boldsymbol{w}_G]\boldsymbol{u} \\ &\leq \frac{1}{n}\left(\sup_{\boldsymbol{u}\in\mathbb{S}^{n-1}}\frac{\boldsymbol{u}'[\boldsymbol{I}_n \otimes \boldsymbol{w}_G']\boldsymbol{\Sigma}[\boldsymbol{I}_n \otimes \boldsymbol{w}_G]\boldsymbol{u}}{\boldsymbol{u}'[\boldsymbol{I}_n \otimes \boldsymbol{w}_G'][\boldsymbol{I}_n \otimes \boldsymbol{w}_G]\boldsymbol{u}}\right)\left(\sup_{\boldsymbol{u}\in\mathbb{S}^{n-1}}\boldsymbol{u}'[\boldsymbol{I}_n \otimes \boldsymbol{w}_G'\boldsymbol{w}_G]\boldsymbol{u}\right) \\ &\leq \frac{C_x}{n}\|\boldsymbol{w}_G\|_2^2 \leq \frac{C_x}{n}\|\boldsymbol{U}_G\|_{\mathrm{op}}^2\|\boldsymbol{\Delta}\|_2^2 \leq \frac{\kappa_U}{n}.\end{aligned} \tag{18}$$

Finally we can use (16) and (18) to bound the Frobenius norm of $\boldsymbol{Q}$. By some algebra, for any square matrices $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{n\times n}$, $\|\boldsymbol{AB}\|_{\mathrm{F}}^2 \le \|\boldsymbol{A}\|_{\mathrm{op}}^2 \|\boldsymbol{B}\|_{\mathrm{F}}^2$ holds. Hence,

$$\|\boldsymbol{Q}\|_{\mathrm{F}}^2 = \|\boldsymbol{Q}_1\boldsymbol{Q}_1'\|_{\mathrm{F}}^2 \le \|\boldsymbol{Q}_1\|_{\mathrm{op}}^2 \|\boldsymbol{Q}_1\|_{\mathrm{F}}^2 = \|\boldsymbol{Q}_1\|_{\mathrm{op}}^2 \operatorname{tr}(\boldsymbol{Q}) \le \frac{\kappa_U^2}{n}.$$

This, together with (15), (18) and the Hanson-Wright inequality (Vershynin, 2018, Chapter 6), leads to

$$\begin{aligned}\mathbb{P}\left\{\left|\|\boldsymbol{\Delta}\|_n^2 - \mathbb{E}\|\boldsymbol{\Delta}\|_n^2\right| \ge t\right\} &\le 2\exp\left\{-c_{\mathrm{H}} \min\left(\frac{t}{\|\boldsymbol{Q}\|_{\mathrm{op}}}, \frac{t^2}{\|\boldsymbol{Q}\|_{\mathrm{F}}^2}\right)\right\} \\ &\le 2\exp\left\{-c_{\mathrm{H}} n \min\left(t/\kappa_U, (t/\kappa_U)^2\right)\right\},\end{aligned} \tag{19}$$

where c_{H} is a positive constant.

On the other hand,

$$\|\boldsymbol{\Delta}\|_n^2 - \mathbb{E}\|\boldsymbol{\Delta}\|_n^2 = \frac{1}{n}\sum_{i=1}^n \boldsymbol{\Delta}'\boldsymbol{z}^i\boldsymbol{z}^{i\prime}\boldsymbol{\Delta} - \mathbb{E}(\boldsymbol{\Delta}'\boldsymbol{z}^i\boldsymbol{z}^{i\prime}\boldsymbol{\Delta}) = \boldsymbol{\Delta}'\widehat{\boldsymbol{\Gamma}}\boldsymbol{\Delta},$$

where $\widehat{\boldsymbol{\Gamma}} = n^{-1}\sum_{i=1}^n[\boldsymbol{z}^i\boldsymbol{z}^{i\prime} - E(\boldsymbol{z}^i\boldsymbol{z}^{i\prime})]$ is a symmetric matrix. Consider a ε-net $\bar{\mathcal{S}}_{2K}$, with the cardinality of $\mathcal{N}(2K,\varepsilon)$, for the set $\mathcal{S}_{2K}$. For any $\boldsymbol{\Delta} \in \mathcal{S}_{2K}$, there exists a $\bar{\boldsymbol{\Delta}}_j \in \bar{\mathcal{S}}_{2K}$ such that $\|\boldsymbol{\Delta} - \bar{\boldsymbol{\Delta}}_j\|_{\mathrm{F}} \le \varepsilon$. Note that $\boldsymbol{\Delta} - \bar{\boldsymbol{\Delta}}_j \in \widehat{\mathcal{S}}_{4K}$ and, from Lemma 1, we can further find $\boldsymbol{\Delta}_1, \boldsymbol{\Delta}_2 \in \widehat{\mathcal{S}}_{2K}$ such that $\langle \boldsymbol{\Delta}_1, \boldsymbol{\Delta}_2\rangle = 0$ and $\boldsymbol{\Delta} - \bar{\boldsymbol{\Delta}}_j = \boldsymbol{\Delta}_1 + \boldsymbol{\Delta}_2$, and it then holds that $\|\boldsymbol{\Delta}_1\|_{\mathrm{F}} + \|\boldsymbol{\Delta}_2\|_{\mathrm{F}} \le \sqrt{2}\|\boldsymbol{\Delta} - \bar{\boldsymbol{\Delta}}_j\|_{\mathrm{F}} \le \sqrt{2}\varepsilon$. Moreover, for a general real symmetric matrix $\boldsymbol{A} \in \mathbb{R}^{d\times d}$, $\boldsymbol{u} \in \mathbb{R}^d$ and $\boldsymbol{v} \in \mathbb{R}^d$, $\sup_{\|\boldsymbol{u}\|_2=\|\boldsymbol{v}\|_2=1} \boldsymbol{u}'\boldsymbol{A}\boldsymbol{v} = \sup_{\|\boldsymbol{u}\|_2=1} \boldsymbol{u}'\boldsymbol{A}\boldsymbol{u}$. As a result,

$$\begin{aligned}\boldsymbol{\Delta}'\widehat{\boldsymbol{\Gamma}}\boldsymbol{\Delta} &= \bar{\boldsymbol{\Delta}}_j'\widehat{\boldsymbol{\Gamma}}\bar{\boldsymbol{\Delta}}_j + (\boldsymbol{\Delta}_1 + \boldsymbol{\Delta}_2)'\widehat{\boldsymbol{\Gamma}}(\boldsymbol{\Delta}_1 + \boldsymbol{\Delta}_2 + 2\bar{\boldsymbol{\Delta}}_j) \\ &\le \max_{1\le j\le \mathcal{N}(2K,\varepsilon)} \bar{\boldsymbol{\Delta}}_j'\widehat{\boldsymbol{\Gamma}}\bar{\boldsymbol{\Delta}}_j + 5\varepsilon \sup_{\boldsymbol{\Delta}\in\mathcal{S}_{2K}} \boldsymbol{\Delta}'\widehat{\boldsymbol{\Gamma}}\boldsymbol{\Delta},\end{aligned}$$

where $(\|\boldsymbol{\Delta}_1\|_{\mathrm{F}} + \|\boldsymbol{\Delta}_2\|_{\mathrm{F}})^2 + 2(\|\boldsymbol{\Delta}_1\|_{\mathrm{F}} + \|\boldsymbol{\Delta}_2\|_{\mathrm{F}})\|\bar{\boldsymbol{\Delta}}_j\|_{\mathrm{F}} \le 2(\varepsilon + \sqrt{2})\varepsilon \le 5\varepsilon$ as $\varepsilon \le 1$, and this leads to

$$\sup_{\boldsymbol{\Delta}\in\mathcal{S}_{2K}} \boldsymbol{\Delta}'\widehat{\boldsymbol{\Gamma}}\boldsymbol{\Delta} \le (1-5\varepsilon)^{-1} \max_{1\le j\le \mathcal{N}(2K,\varepsilon)} \bar{\boldsymbol{\Delta}}_j'\widehat{\boldsymbol{\Gamma}}\bar{\boldsymbol{\Delta}}_j. \tag{20}$$

Note that, from Lemma 1(b), $\log\mathcal{N}(2K,\varepsilon) \le 2d_{\mathcal{M}}\log(9/\varepsilon)$, where $d_{\mathcal{M}} = K(L+P+1)$. Let $\varepsilon = 1/10$, and then $2\log(9/\varepsilon) < 9$. Combining (19) and (20) and letting $t = \kappa_L/2$, we have

$$\mathbb{P}\left\{\sup_{\boldsymbol{\Delta}\in\mathcal{S}_{2K}} \|\boldsymbol{\Delta}\|_n^2 - \mathbb{E}\|\boldsymbol{\Delta}\|_n^2 \ge \frac{\kappa_L}{2}\right\} \le 2\exp\left\{-c_{\mathrm{H}} n\left(\frac{\kappa_L}{4\kappa_U}\right)^2 + 9d_{\mathcal{M}}\right\},$$

which, together with (17) and the fact that $\kappa_L \le \kappa_U$, implies that

$$\mathbb{P}\left\{\sup_{\boldsymbol{\Delta}\in\mathcal{S}_{2K}} \|\boldsymbol{\Delta}\|_n^2 \ge \alpha_{\mathrm{RSM}}\right\} \le 2\exp\left\{-c_{\mathrm{H}} n\left(\frac{\kappa_L}{4\kappa_U}\right)^2 + 9d_{\mathcal{M}}\right\}, \tag{21}$$

where $\alpha_{\mathrm{RSM}} = 1.5\kappa_U \ge \kappa_U + \kappa_L/2$.

By a method similar to (20), we can also show that

$$\sup_{\boldsymbol{\Delta}\in\mathcal{S}_{2K}} \mathbb{E}\|\boldsymbol{\Delta}\|_n^2 - \|\boldsymbol{\Delta}\|_n^2 \le (1-5\varepsilon)^{-1} \max_{1\le j\le \mathcal{N}(2K,\varepsilon)} \mathbb{E}\left\|\bar{\boldsymbol{\Delta}}_j\right\|_n^2 - \left\|\bar{\boldsymbol{\Delta}}_j\right\|_n^2,$$

which, together with (19), leads to

$$\mathbb{P}\left\{\sup_{\boldsymbol{\Delta}\in\mathcal{S}_{2K}} \mathbb{E}\|\boldsymbol{\Delta}\|_n^2 - \|\boldsymbol{\Delta}\|_n^2 \ge \frac{\kappa_L}{2}\right\} \le 2\exp\left\{-c_{\mathrm{H}} n\left(\frac{\kappa_L}{4\kappa_U}\right)^2 + 9d_{\mathcal{M}}\right\},$$

where $t = \kappa_L/2$ and $\varepsilon = 1/10$. As a result,

$$\mathbb{P}\left\{\inf_{\boldsymbol{\Delta}\in\mathcal{S}_{2K}} \|\boldsymbol{\Delta}\|_n^2 \le \alpha_{\mathrm{RSC}}\right\} \le 2\exp\left\{-c_{\mathrm{H}} n\left(\frac{\kappa_L}{4\kappa_U}\right)^2 + 9d_{\mathcal{M}}\right\},$$

where $\alpha_{\mathrm{RSC}} = \kappa_L - \kappa_L/2$. This, together with (21), accomplishes the proof. □

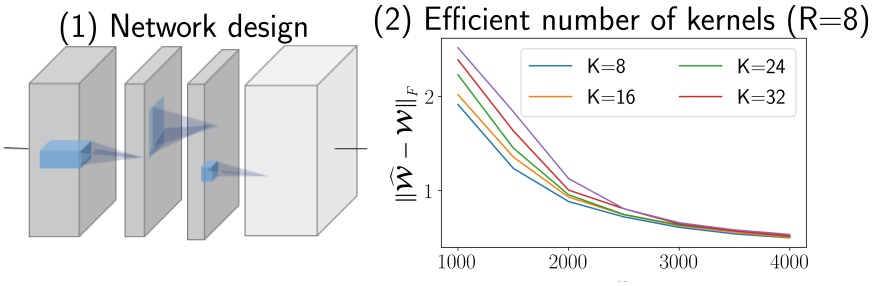

Figure 7: Additional experiments for efficient number of kernels.

Lemma 3. *(Concentration bound for martingale) Let $\{\xi^i, 1 \leq i \leq n\}$ be independent σ^2-sub-Gaussian random variables with mean zero, and $\{z^i, 1 \leq i \leq n\}$ is another sequence of random variables. Suppose that ξ^i is independent of $\{z^i, z^{i-1}, \ldots, z^1\}$ for all $1 \leq i \leq n$. It then holds that, for any real numbers $\alpha, \beta > 0$,*

$$\mathbb{P}\left[\left\{\frac{1}{n}\sum_{i=1}^{n}\xi^i z^i \geq \alpha\right\} \cap \left\{\frac{1}{n}\sum_{i=1}^{n}(z^i)^2 \leq \beta\right\}\right] \leq \exp\left(-\frac{n\alpha^2}{2\sigma^2\beta}\right).$$

Proof. We can prove the lemma by a method similar to Lemma 4.2 in Simchowitz et al. (2018). □

A.4 Additional Experiments

We first provide some additional implementation details for Section 4.1 in the paper. We consider two types of inputs for verification of the sample complexity in Theorem 1. For the time-dependent inputs, we generate a sequence of vectors $\{\boldsymbol{x}^i\}$ with size $\mathbb{R}^d$ from a stationary VAR(1) process, where d = 245 or 192. The VAR(1) process has a transition matrix with spectral norm less than 1 to ensure stationarity, and a white noise sequence which follows multivariate standard normal distribution. The first 50 simulated data points in the sequence are discarded to alleviate the influence of initialization, and the length of the resulting sequence is equal to the training sample size plus the testing sample size, i.e. $n + 200$. We then reshape $\{\boldsymbol{x}^i\}$ into a sequence of tensors of shape (7, 5, 7) or (8, 8, 3), which are the time-dependent inputs for experiments under S1/S2 or S3/S4.

Next, we conduct extra experiments for efficient number of kernels for a CP block design. This study uses 32×32 inputs with 16 channels, and we set stride to 1 and pooling sizes to $(5, 5)$. We generate the orthonormal factor matrices $\{\boldsymbol{H}^{(j)}, 1 \leq j \leq 4\}$, where $\boldsymbol{H}^{(1)}$ is of size $\mathbb{R}^{8\times R}$, $\boldsymbol{H}^{(2)}$ is of size $\mathbb{R}^{8\times R}$, $\boldsymbol{H}^{(3)}$ is of size $\mathbb{R}^{16\times R}$ and $\boldsymbol{H}^{(4)}$ is of size $\mathbb{R}^{K\times R}$ where $R = 8$ and $K \in \{8, 16, 24, 32\}$. If we denote the orthonormal column vectors of $\boldsymbol{H}^{(j)}$ by $\boldsymbol{h}_r^{(j)}$ where $1 \leq r \leq R$ and $1 \leq j \leq 4$, the stacked kernel tensor $\boldsymbol{\mathcal{A}}$ can then be generated as $\boldsymbol{\mathcal{A}} = \sum_{r=1}^{R} \boldsymbol{h}_r^{(1)} \circ \boldsymbol{h}_r^{(2)} \circ \boldsymbol{h}_r^{(3)} \circ \boldsymbol{h}_r^{(4)}$. And it can be seen that $\boldsymbol{\mathcal{A}}$ has a CP rank of $R = 8$. We split the stacked kernel tensor $\boldsymbol{\mathcal{A}}$ along the kernel dimension to obtain $\{\boldsymbol{\mathcal{A}}_k, 1 \leq k \leq K\}$ and generate the corresponding fully-connected weight tensors $\{\boldsymbol{\mathcal{B}}_k, 1 \leq k \leq K\}$ with standard normal entries. The parameter tensor $\boldsymbol{\mathcal{W}}$ is hence obtained, and we further normalize it to have unit Frobenius norm to ensure the comparability of estimation errors between different Ks. The block structure in Figure 7(1) is employed to train the network, and it is equivalent to the bottleneck structure with a CP decomposition on $\boldsymbol{\mathcal{A}}$; see Kossaifi et al. (2020). We can see that as K increases, there is more redundancy in the network parameters. Fifty training sets are generated for each training size, and we stop training when the target function drops by less than 10^{-5}. From Figure 7(2), the estimation error increases as K is larger, and the difference is more pronounced when training size is small.