Tensors in Modern Statistical Learning

Will Wei Sun[†], Botao Hao[‡], and Lexin Li[*]

[†]Purdue University, [‡]DeepMind, and [*]University of California at Berkeley

Abstract

Tensor learning is gaining increasing attention in recent years. This survey provides an overview of tensor analysis in modern statistical learning. It consists of four main topics, including tensor supervised learning, tensor unsupervised learning, tensor reinforcement learning, and tensor deep learning. This review emphasizes statistical models and properties, as well as connections between tensors and other learning topics such as reinforcement learning and deep learning.

Keywords: Tensor decomposition; Tensor regression; Tensor clustering; Tensor graphical model; Tensor reinforcement learning; Tensor deep learning.

1 Introduction

Tensors, also known as multidimensional arrays, are generalizations of vectors and matrices to higher dimensions. In recent years, tensor data are fast emerging in a wide variety of scientific and business applications, including but not limited to recommendation systems (Rendle and Schmidt-Thieme, 2010; Bi et al., 2018), speech or facial recognitions (Vasilescu and Terzopoulos, 2002; Ma et al., 2019), networks analysis (Li et al., 2011; Ermis et al., 2015), knowledge graphs and relational learning (Trouillon et al., 2017; Liu et al., 2020), among many others. Tensor data analysis is thus gaining increasing attention in statistics and machine learning communities. In this survey, we provide an overview of tensor analysis in modern statistical learning.

We begin with a brief introduction of tensor notations, tensor algebra, and tensor decompositions. For more details on tensor basics, we refer to Kolda and Bader (2009). We then divide our survey into four topics, depending on the nature of the learning problems: (a) tensor supervised learning, including tensor predictor regression and tensor response regression, (b) tensor unsupervised learning, including tensor clustering and tensor graphical

model, (c) tensor reinforcement learning, including low-rank tensor bandit and low-rank Markov decision process, and (d) tensor deep learning, including deep neural networks compression and deep learning theory via tensor formulation. For each topic, we start with the study goals and some motivating applications. We then review several key methods and some related solutions. We conclude each topic by a discussion of some open problems and potential future directions.

We also note that, there have already been several excellent survey papers on tensor learning in statistics and machine learning, for instance, Rabanser et al. (2017); Sidiropoulos et al. (2017); Janzamin et al. (2019); Song et al. (2019); Bi et al. (2020). However, our review differs in terms of the focus and the organization of different tensor learning topics. Particularly, Rabanser et al. (2017); Sidiropoulos et al. (2017); Janzamin et al. (2019) concentrated on tensor decomposition, which aims to dissolve tensors into separable representations, while Song et al. (2019) reviewed tensor completion, which aims to impute the unobserved entries of a partially observed tensor. Tensor decomposition and tensor completion are both fundamental problems in tensor data analysis. However, given there are already fairly thorough reviews on these topics, we will not go over them in detail, but instead refer to the aforementioned survey articles. Bi et al. (2020) divided numerous tensor methods by three major application areas, i.e., recommendation systems, biomedical imaging, and network analysis. We instead divide our review by different types of learning problems. Moreover, Bi et al. (2020) only briefly mentioned some connections between tensor analysis and deep learning, while one of the focuses of our review is about more recent topics of tensor reinforcement learning and tensor deep learning and their relations with tensor analysis.

Given fast development of tensor learning, it is inevitable that we will miss some important papers in this survey. Nevertheless, our goal is to provide a good entry point to the area of tensor data analysis, with emphasis on statistical models and properties, as well as connections with other learning topics.

2 Background

We begin with a brief review of some basics of tensors. For more details, we refer to Kolda and Bader (2009) for an excellent review.

Notations: The *order* of a tensor, also referred to as the *mode*, is the dimension of the array. A first-order tensor is a vector, a second-order tensor is a matrix, and tensors of order three and higher are referred to as high-order tensors, see Figure 1. The *fiber* of a tensor is defined by fixing all indices but one. For example, given a third-order tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{p_1 \times p_2 \times p_3}$, its mode-1, 2 and 3 fibers are denoted as $\boldsymbol{\mathcal{X}}_{:jk}$, $\boldsymbol{\mathcal{X}}_{i:k}$ and $\boldsymbol{\mathcal{X}}_{ij:}$, respectively.

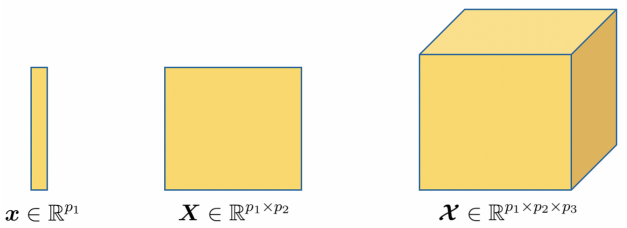

Figure 1: An example of first, second and third-order tensors.

Tensor operations: Tensor *unfolding*, also known as tensor *matricization*, is a tensor operation that arranges tensor fibers into a matrix. Given a tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{p_1 \times p_2 \times \ldots \times p_D}$, the mode-$d$ unfolding, denotes as $\boldsymbol{\mathcal{X}}_{(d)}$, arranges the mode-$d$ fibers to be the columns of the resulting matrix. For example, the mold-1 unfolding of a third-order tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{p_1 \times p_2 \times d_3}$, denoted by $\boldsymbol{\mathcal{X}}_{(1)}$, results in the matrix $[\boldsymbol{\mathcal{X}}_{:11}, \ldots, \boldsymbol{\mathcal{X}}_{:p_2 1}, \ldots, \boldsymbol{\mathcal{X}}_{:p_2 p_3}] \in \mathbb{R}^{p_1 \times (p_2 p_3)}$; see Figure 2 for a graphic illustration. Tensor *vectorization* is a tensor operation that arranges tensor fibers into a vector. The vectorization of tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{p_1 \times p_2 \times \ldots \times p_D}$, denoted by $\text{vec}(\boldsymbol{\mathcal{X}})$, is the vector of length $\prod_{d=1}^{D} p_d$ that is obtained by stacking the mode-1 fibers of $\boldsymbol{\mathcal{X}}$. For example, given an order-three tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{p_1 \times p_2 \times p_3}$, $\text{vec}(\boldsymbol{\mathcal{X}}) = \left(\boldsymbol{\mathcal{X}}_{:11}^\top, \ldots, \boldsymbol{\mathcal{X}}_{:p_2 1}^\top, \ldots, \boldsymbol{\mathcal{X}}_{:p_2 p_3}^\top\right)^\top$; again see Figure 2 for an illustration.

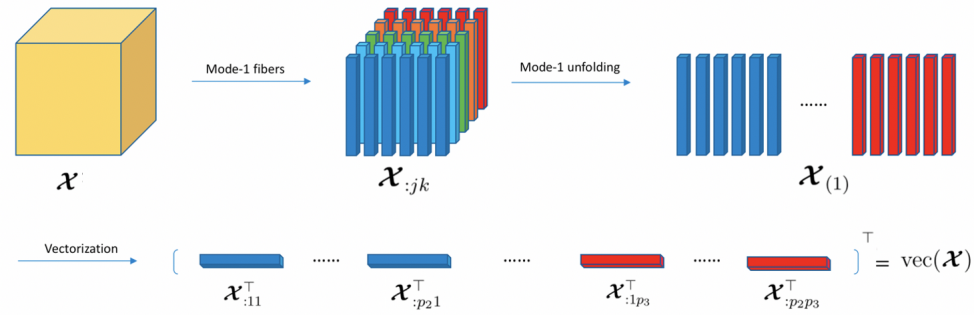

Figure 2: Tensor fibers, unfolding and vectorization.

For two tensors $\boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{Y}} \in \mathbb{R}^{p_1 \times p_2 \times \ldots \times p_D}$, their *inner product* is defined as $\langle \boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{Y}} \rangle = \sum_{i_1, \ldots, i_D} \boldsymbol{\mathcal{X}}_{i_1, \ldots, i_D} \boldsymbol{\mathcal{Y}}_{i_1, \ldots, i_D}$. For a tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{p_1 \times p_2 \times \ldots \times p_M}$ and a matrix $\boldsymbol{A} \in \mathbb{R}^{J \times p_m}$, the *d-mode tensor matrix product*, denoted by $\times_d$, is defined as $\boldsymbol{\mathcal{X}} \times_d \boldsymbol{A} \in \mathbb{R}^{p_1 \times \ldots \times p_{d-1} \times J \times p_{d+1} \times \ldots \times p_D}$. In this operation, each mode-d fiber of $\boldsymbol{\mathcal{X}}$ is multiplied by the matrix $\boldsymbol{A}$, and elementwisely, we have $(\boldsymbol{\mathcal{X}} \times_d \boldsymbol{A})_{i_1, \ldots, i_{d-1}, j, i_{d+1}, \ldots, i_D} = \sum_{i_d=1}^{p_d} \boldsymbol{\mathcal{X}}_{i_1, \ldots, i_D} \boldsymbol{A}_{j i_d}$.

Tensor decompositions: We next introduce two tensor decompositions that play fundamental roles in tensor data analysis.

The first is the *CP-decomposition*. For a Dth-order tensor $\boldsymbol{\mathcal{B}}^*$, the rank-$R$ CP decomposi-

tion of $\boldsymbol{\mathcal{B}}^*$ is defined as,

$$\boldsymbol{\mathcal{B}}^* = \sum_{r=1}^{R} w_r^* \boldsymbol{\beta}_{r,1}^* \circ \cdots \circ \boldsymbol{\beta}_{r,D}^*, \tag{1}$$

where $w_r^* \in \mathbb{R}$, $\boldsymbol{\beta}_{r,d}^* \in \mathbb{S}^{p_d}$, $r = 1, \ldots, R, d = 1, \ldots, D$, $\mathbb{S}^d = \{\mathbf{v} \in \mathbb{R}^d \,|\, \|\mathbf{v}\| = 1\}$, and $\circ$ denotes the outer product. The CP-decomposition is sometimes abbreviated as $\boldsymbol{\mathcal{B}}^* = [\![\boldsymbol{W}^*; \boldsymbol{B}_1^*, \ldots, \boldsymbol{B}_D^*]\!]$, where $\boldsymbol{W}^* = \mathrm{diag}(w_1^*, \ldots, w_R^*) \in \mathbb{R}^{R \times R}$ is a diagonal matrix, and $\boldsymbol{B}_d^* = [\boldsymbol{\beta}_{1,d}^*, \ldots, \boldsymbol{\beta}_{R,d}^*] \in \mathbb{R}^{p_d \times R}$ are the factor matrices. If $\boldsymbol{\mathcal{B}}^*$ admits a CP structure (1), then the number of free parameters in $\boldsymbol{\mathcal{B}}^*$ is reduced from $\prod_{d=1}^D p_d$ to $R \times \sum_{d=1}^D p_d$.

The second is the *Tucker decomposition*. For a Dth-order tensor $\boldsymbol{\mathcal{B}}^*$, the rank-$(R_1, \ldots, R_D)$ Tucker decomposition of $\boldsymbol{\mathcal{B}}^*$ is defined as,

$$\boldsymbol{\mathcal{B}}^* = \sum_{r_1=1}^{R_1} \cdots \sum_{r_D=1}^{R_D} w_{r_1,\ldots,r_D}^* \boldsymbol{\beta}_{r_1,1}^* \circ \ldots \circ \boldsymbol{\beta}_{r_D,D}^*, \tag{2}$$

where $w_{r_1,\ldots,r_D}^* \in \mathbb{R}$, $\boldsymbol{\beta}_{r_d,d}^* \in \mathbb{S}^{p_d}$, $r_d = 1, \ldots, R_d, d = 1, \ldots, D$. The Tucker decomposition is sometimes abbreviated as $\boldsymbol{\mathcal{B}}^* = [\![\boldsymbol{W}^*; \boldsymbol{B}_1^*, \ldots, \boldsymbol{B}_D^*]\!]$, where $\boldsymbol{W}^* = (w_{r_1,\ldots,r_D}^*) \in \mathbb{R}^{R_1 \times \ldots \times R_D}$ is the Dth-order core tensor, and $\boldsymbol{B}_d^* = [\boldsymbol{\beta}_{1,d}^*, \ldots, \boldsymbol{\beta}_{R_d,d}^*] \in \mathbb{R}^{p_d \times R_d}$ are the factor matrices. If $\boldsymbol{\mathcal{B}}^*$ admits a Tucker structure (2), then the number of free parameters in $\boldsymbol{\mathcal{B}}^*$ is reduced from $\prod_{d=1}^D p_d$ to $\sum_{d=1}^D R_d \times p_d + \prod_{d=1}^D R_d$.

3 Tensor Supervised Learning

The first topic we review is tensor supervised learning, where the primary goal is to study the association between a tensor object and some other univariate or multivariate variables. The problem can be cast as a regression, and tensor can appear at either the predictor side or the response side. This leads to the two subtopics we review: the tensor predictor regression and the tensor response regression. The tensor supervised learning idea can also be generalized to involve multiple tensors on one side of the regression, or having tensors showing up on both sides of the regression model.

3.1 Tensor Predictor Regression

Motivating examples: Neuroimaging data often take the form of tensors. For instance, electroencephalography (EEG) measures voltage value from numerous electrodes placed on scalp over time, and the resulting data is a two-dimensional matrix. Anatomical magnetic resonance imaging (MRI) measures brain structural features such as cortical thickness, and the data is a three-dimensional tensor. Figure 3 shows an example of 3D MRI at different

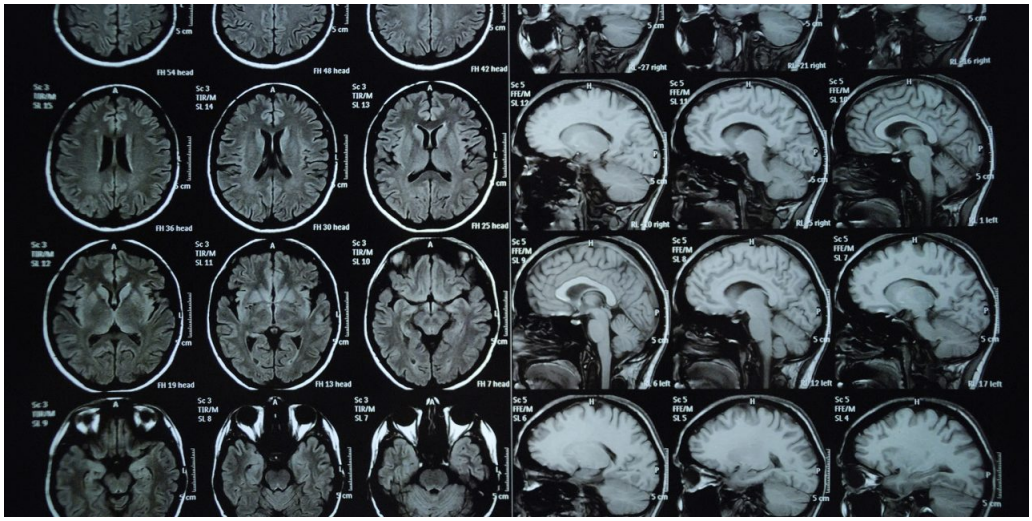

Figure 3: An example of magnetic resonance imaging. The image is obtained from internet.

slices and directions. It is often of great scientific interest to model the association between the tensor-valued images and the clinical outcomes such as diagnostic status, or cognition and memory scores. This can be formulated as a tensor predictor regression problem, where the response is a binary or continuous scalar, the predictor is an imaging tensor, and the goal is to understand the change of the outcome as a function of the tensor.

Low-rank linear and generalized linear model: Consider a Dth-order tensor predictor $\boldsymbol{\mathcal{X}}_i \in \mathbb{R}^{p_1 \times \ldots \times p_D}$ and a scalar response $y_i \in \mathbb{R}$, for i.i.d. data replications $i = 1, \ldots, n$. Zhou et al. (2013) considered the tensor predictor regression model of the form,

$$y_i = \langle \boldsymbol{\mathcal{B}}^*, \boldsymbol{\mathcal{X}}_i \rangle + \epsilon_i, \tag{3}$$

where $\boldsymbol{\mathcal{B}}^* \in \mathbb{R}^{p_1 \times \ldots \times p_D}$ denotes the coefficient tensor that captures the association between $\boldsymbol{\mathcal{X}}_i$ and y_i and is of the primary interest, and $\epsilon_i \in \mathbb{R}$ denotes the measurement error. Without loss of generality, the intercept term is set to zero to simplify the presentation. Model (3) is a direct generalization of the classical multivariate linear regression model. The issue, however, is that $\boldsymbol{\mathcal{B}}^*$ involves $\prod_{d=1}^{D} p_d$ parameters, which is ultrahigh dimensional and far exceeds the typical sample size. To efficiently reduce the dimensionality, Zhou et al. (2013) imposed the CP low-rank structure (1) on $\boldsymbol{\mathcal{B}}^*$. Accordingly, the number of unknown parameters involved in $\boldsymbol{\mathcal{B}}^*$ is reduced to $R \sum_{d=1}^{D} p_d$. They then proposed to estimate $\boldsymbol{\mathcal{B}}^*$ via penalized maximal likelihood estimation, by solving

$$\min_{w_r, \boldsymbol{\beta}_{r,1}, \ldots, \boldsymbol{\beta}_{r,D}} \sum_{i=1}^{n} \left(y_i - \left\langle \sum_{r=1}^{R} w_r \boldsymbol{\beta}_{r,1} \circ \ldots \circ \boldsymbol{\beta}_{r,D}, \boldsymbol{\mathcal{X}}_i \right\rangle \right)^2 + \sum_{d=1}^{D} \sum_{r=1}^{R} P_\lambda(|\boldsymbol{\beta}_{r,d}|), \tag{4}$$

under the additional constraints that $w_r > 0$ and $\|\boldsymbol{\beta}_{r,d}\|_2 = 1$ for all $r = 1, \ldots, R$ and $d = 1, \ldots, D$, and $P_\lambda(\cdot)$ is a sparsity-inducing penalty function indexed by the tuning parameter λ. This penalty helps to obtain a sparse estimate of $\boldsymbol{\beta}_{r,d}$, which translates to sparsity in the blocks of $\boldsymbol{\mathcal{B}}^*$, and in turn facilitates the interpretation of $\boldsymbol{\mathcal{B}}^*$. Denote the factor matrices $\boldsymbol{B}_d = [\boldsymbol{\beta}_{1,d}, \ldots, \boldsymbol{\beta}_{R,d}] \in \mathbb{R}^{p_d \times R}$, for $d = 1, \ldots, D$. Zhou et al. (2013) proposed a block updating algorithm to solve (4) for each $\boldsymbol{B}_d$ while fixing all other $\boldsymbol{B}_{d'}, d' \neq d$. They further considered a generalized linear model formulation of (3) by introducing a link function so to work with a binary or count type y_i.

Relatedly, Li et al. (2016) extended (3) to multivariate response variables. Guhaniyogi et al. (2017) formulated the tensor predictor regression (3) in a Bayesian setting, and introduced a novel class of multiway shrinkage priors for tensor coefficients. Li et al. (2018b) considered the Tucker decomposition (2) for $\boldsymbol{\mathcal{B}}^*$ and demonstrated its flexibility over the CP decomposition. Zhang et al. (2019) extended (3) to the generalized estimating equation setting for longitudinally observed imaging tensors.

Large-scale tensor regression via sketching: A common challenge associated with the tensor predictor regression with a low-rank factorization is the high computational cost. This is especially true when the dimension of the tensor predictor is large. Sketching offers a natural solution to address this challenge, and is particularly useful when the dimensionality is ultrahigh, the sample size is super large, or the data is extremely sparse.

Yu and Liu (2016) introduced the subsampled tensor projected gradient approach for a variety of tensor regression problems, including the situation when the response is a tensor too. Their algorithm was built upon the projected gradient method with fast tensor power iterations, and leveraged randomized sketching for further acceleration. In particular, they used count sketch (Clarkson and Woodruff, 2017) as a subsampling step to generate a reduced data, then feed the data into tensor projected gradient to estimate the final parameters.

Zhang et al. (2020) utilized importance sketching for low-rank tensor regressions. They carefully designed sketches based on both the response and the low-dimensional structure of the parameter of interest. They proposed an efficient algorithm, which first used the high-order orthogonal iteration (De Lathauwer et al., 2000) to determine the importance sketching directions, then performed importance sketching and evaluated the dimension-reduced regression using the sketched tensors, and constructed the final tensor estimator using the sketched components. They showed that their algorithm achieves the optimal mean-squared error under the low-rank Tucker structure and randomized Gaussian design.

Nonparametric tensor regression: Although the linear tensor regression provides a simple and concise solution, the linearity assumption in (3) can be restrictive in numerous applications (Kanagawa et al., 2016; Suzuki et al., 2016). For instance, Hao et al. (2019) showed that, in a digital advertising study, the association between the click-through-rate

and the impression tensor of various ads on different devices is clearly nonlinear.

Hao et al. (2019) proposed a nonparametric extension of model (3), by assuming

$$y_i = \sum_{j_1=1}^{p_1} \cdots \sum_{j_D=1}^{p_D} f^*_{j_1 \ldots j_D} \left([\boldsymbol{\mathcal{X}}_i]_{j_1 \ldots j_D} \right) + \epsilon_i, \tag{5}$$

where $[\boldsymbol{\mathcal{X}}_i]_{j_1 \ldots j_D}$ denotes the $(j_1, \ldots, j_D)$th entry of the tensor $\boldsymbol{\mathcal{X}}_i$, and $f^*_{j_1 \ldots j_D}(\cdot)$ is some smooth function that can be approximated by B-splines (Hastie and Tibshirani, 1990),

$$f^*_{jkl}([\boldsymbol{\mathcal{X}}_i]_{jkl}) \approx \sum_{h=1}^{H} \beta^*_{j_1 \ldots j_D h} \psi_h([\boldsymbol{\mathcal{X}}_i]_{j_1 \ldots j_D}), \ 1 \le j_1 \le p_1, \ldots, 1 \le j_D \le p_3,$$

with the B-spline basis $\psi_{j_1 \ldots j_D h}$ and coefficients $\beta^*_{j_1 \ldots j_D h}$. Let $[\boldsymbol{\mathcal{F}}_h(\boldsymbol{\mathcal{X}}_i)]_{j_1 \ldots j_D} = \psi_{j_1 \ldots j_D h}([\boldsymbol{\mathcal{X}}_i]_{j_1 \ldots j_D})$ and $[\boldsymbol{\mathcal{B}}_h]_{j_1 \ldots j_D} = \beta^*_{j_1 \ldots j_D h}$. The compact tensor representation of their model is

$$y_i = \sum_{h=1}^{H} \langle \boldsymbol{\mathcal{B}}_h, \boldsymbol{\mathcal{F}}_h(\boldsymbol{\mathcal{X}}_i) \rangle + \epsilon_i. \tag{6}$$

In this model, $\boldsymbol{\mathcal{F}}_h(\boldsymbol{\mathcal{X}}_i) \in \mathbb{R}^{p_1 \times \ldots \times p_D}$ is the predictor tensor under the B-spline transformation, and $\boldsymbol{\mathcal{B}}_h \in \mathbb{R}^{p_1 \times \ldots \times p_D}$ captures the association information. The linear tensor regression model (3) becomes a special case of (6), with $\psi_{j_1 \ldots j_D h}(x) = x$ and $H = 1$. By considering nonlinear basis functions, e.g., trigonometric functions, model (6) is more flexible and has a better prediction power. Moreover, Hao et al. (2019) imposed the CP structure (1) on $\boldsymbol{\mathcal{B}}_h$, and a group-wise penalty to screen out the nuisance components. They proposed to solve the following penalized optimization problem,

$$\min_{\boldsymbol{\beta}_{1hr}, \ldots, \boldsymbol{\beta}_{Dhr}} \frac{1}{n} \sum_{i=1}^{n} \left(y_i - \sum_{h=1}^{H} \left\langle \sum_{r=1}^{R} \boldsymbol{\beta}_{1hr} \circ \ldots \circ \boldsymbol{\beta}_{Dhr}, \boldsymbol{\mathcal{F}}_h(\boldsymbol{\mathcal{X}}_i) \right\rangle \right)^2 + \lambda \sum_{d=1}^{D} \sum_{j=1}^{p_d} \sqrt{\sum_{h=1}^{H} \sum_{r=1}^{R} \beta^2_{dhrj}}. \tag{7}$$

The optimization in (7) is done in a block-wise fashion for $\boldsymbol{\beta}_{dhr}$, $d = 1, \ldots, D$, and each block is solved by the back-fitting algorithm for the standard sparse additive model (Ravikumar et al., 2009). The regularization parameter λ is tuned by cross-validation.

Relatedly, Zhou et al. (2020b) considered a broadcasted nonparametric tensor regression model where all entries of the tensor covariate are assumed to share the same function, which is a special case of (5).

Future directions: There are a number of open questions for tensor predictor regression. One is to integrate multiple tensor predictors, each of which represents a tensor measurement from a data modality, and there are multiple modalities of data collected for the same group of experimental subjects. Challenges include how to model the interactions between different

tensors, and how to perform statistical inference. In addition, it is of interest to investigate how to speed-up the computation in nonparametric tensor regression. One possible solution is to use the sketching idea, or the divide-and-conquer approach (Zhang et al., 2015b), when the data can not fit into a single machine.

3.2 Tensor Response Regression

Motivating examples: While the tensor predictor regression focuses on understanding the change of a phenotypic outcome as the tensor varies, in numerous applications, it is important to study the change of the tensor as the covariates vary. One example is anatomical MRI, where the data takes the form of a 3D tensor, and voxels correspond to brain spatial locations. Another example is functional magnetic resonance imaging (fMRI), where the goal is to understand brain functional connectivity encoded by a symmetric matrix, with rows and columns corresponding to brain regions, and entries corresponding to interactions between those regions. In both examples, it is of keen scientific interest to compare the scans of brains, or the brain connectivity patterns, between the subjects with some neurological disorder to the healthy controls, after adjusting for additional covariates such as age and sex. Both can be formulated as a regression problem, with image tensor or connectivity matrix serving as the response, and the disease indicator and other covariates forming the predictors.

Sparse low-rank tensor response model: Consider a Dth-order tensor response $\boldsymbol{\mathcal{Y}}_i \in \mathbb{R}^{p_1 \times \ldots \times p_D}$, and a vector of predictors $\boldsymbol{x}_i \in \mathbb{R}^{p_0}$, for i.i.d. data replications $i = 1, \ldots, n$. Rabusseau and Kadri (2016); Sun and Li (2017) considered the tensor response regression model of the form,

$$\boldsymbol{\mathcal{Y}}_i = \boldsymbol{\mathcal{B}}^* \times_{m+1} \boldsymbol{x}_i + \boldsymbol{\mathcal{E}}_i, \tag{8}$$

where $\boldsymbol{\mathcal{B}}^* \in \mathbb{R}^{p_1 \times \ldots \times p_D \times p_0}$ is an $(D+1)$th-order tensor coefficient that captures the association between $\boldsymbol{x}_i$ and $\boldsymbol{\mathcal{Y}}_i$, and $\boldsymbol{\mathcal{E}}_i \in \mathbb{R}^{p_1 \times \ldots \times p_D}$ is an error tensor that is independent of $\boldsymbol{x}_i$. Without loss of generality, the intercept term is set to zero to simplify the presentation.

Both Rabusseau and Kadri (2016) and Sun and Li (2017) imposed the rank-R CP structure (1) for the coefficient tensor $\boldsymbol{\mathcal{B}}^*$, while Sun and Li (2017) further incorporated the sparsity structure. Specifically, Sun and Li (2017) proposed to solve

$$\min_{\substack{w_r, \boldsymbol{\beta}_{r,d} \\ r \in [R], d \in [D+1]}} \frac{1}{n} \sum_{i=1}^{n} \left\| \boldsymbol{\mathcal{Y}}_i - \sum_{r=1}^{R} w_r (\boldsymbol{\beta}_{r,D+1}^\top \boldsymbol{x}_i) \boldsymbol{\beta}_{r,1} \circ \ldots \circ \boldsymbol{\beta}_{r,D} \right\|_F^2, \text{ subject to } \|\boldsymbol{\beta}_{r,d}\|_0 \leq s_d, \tag{9}$$

and $\|\boldsymbol{\beta}_{r,d}\|_2 = 1$, where s_j is the sparsity parameter. In (9), the sparsity of the decomposed components is encouraged via a hard-thresholding penalty. The optimization in (9) is utterly different from that of (4) for tensor predictor regression, which leads to a more complicated

algorithm and a more subtle interplay between the computational efficiency and the statistical rate of convergence. To solve (9), Sun and Li (2017) proposed an iterative updating algorithm consisting of two major steps. In the first step, the estimation of $w_r, \boldsymbol{\beta}_{r,1}, \ldots, \boldsymbol{\beta}_{r,d}$ for $k \in [K]$, given $\boldsymbol{\beta}_{r,D+1}$, $r \in [R]$ and $w_{r'}, \boldsymbol{\beta}_{r',1}, \ldots, \boldsymbol{\beta}_{r',d}$, $r' \neq r$, is reformulated as a sparse rank-1 tensor decomposition problem (Sun et al., 2017), while in the second step, the estimation of $\boldsymbol{\beta}_{r,D+1}$ for $r \in [R]$, given $w_r, \boldsymbol{\beta}_{r,1}, \ldots, \boldsymbol{\beta}_{r,D}$, $r \in [R]$ and $\boldsymbol{\beta}_{r',D+1}$, $r' \neq r$ becomes a standard least-squares optimization problem and has a closed-form solution.

Additional tensor response regression models: Li and Zhang (2017) proposed an envelope-based tensor response model, which utilized a generalized sparsity principle to exploit the redundant information in the tensor response, and seeked linear combinations of the response that are irrelevant to the regression. Raskutti et al. (2019) developed a class of sparse regression models, under the assumption of Gaussian error, when either or both the response and predictor are tensors. Their approach required a crucial condition that the regularizer was convex and weakly decomposable, and the low-rankness of the estimator was achieved via a tensor nuclear norm penalty. Later, Chen et al. (2019) proposed a projected gradient descent algorithm to efficiently solve the non-convex optimization in tensor response regression, and provided the theoretical guarantees for learning high-dimensional tensor regression models under different low-rank structural assumptions. Motivated by longitudinal neuroimaging studies where image tensors are often missing, Zhou et al. (2020a) developed a regression model with partially observed dynamic tensor as the response and external covariates as the predictor vector. Their solution combined the tensor completion loss idea of a single partially observed tensor (Jain and Oh, 2014) with the tensor response regression model of Sun and Li (2017), and developed an element-wise updating algorithm.

Future directions: There are a number of open questions for tensor response regression. One is how to obtain a consistent estimator of the rank R when the CP structure is employed. More importantly, it remains open to derive the corresponding convergence rate, and combine the estimated rank with the subsequent estimator of $\boldsymbol{\mathcal{B}}^*$ when studying the asymptotic properties. The existing solutions generally treat R as known in the asymptotic studies. Moreover, the current studies have primarily focused on parameter estimation, whereas parameter inference remains a challenging and open question for tensor response regression, especially when the sample size is limited.

4 Tensor Unsupervised Learning

The second topic we review is tensor unsupervised learning, which involves no external variables. We review two topics: tensor clustering, and tensor graphical model. The former

aims to identify clusters by studying the structure of tensor itself, whereas the latter aims to characterize the dependency structure of the individual mode of tensor-valued data.

4.1 Tensor Clustering

Motivating examples: Consider two motivating examples. One is a digital advertisement example consisting of the click-through rates for advertisements displayed on an internet company's webpages over weeks during the ad campaign. The data is a fourth-order tensor, recording the click-through rate of multiple users over a collection of advertisements by different publishers and published on different devices, and the data was aggregated across time. The goal is to simultaneously cluster users, advertisements, and publishers to improve user behavior targeting and advertisement planning. Another example is dynamic brain connectivity analysis based on fMRI data, where the data is in the form of brain region by region by time tensor, and the goal is to cluster over time, so to better understand the interactions of distinct brain regions and their dynamic patterns over time. Both examples can be formulated as a tensor clustering problem. The prevalent clustering solutions, however, have mainly focused on clustering of vector or matrix-valued data. Notably, biclustering extends the classical clustering along both the observations (rows) and the features (columns) of a data matrix (Madeira and Oliveira, 2004; Chi et al., 2017).

Convex tensor co-clustering: We first review a convex co-clustering method that extends biclustering to tensor co-clustering by solving a convex formulation of the problem. Specifically, without loss of generality, Chi et al. (2018) considered a third-order tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{p_1 \times p_2 \times p_3}$. They assumed that the observed data tensor is a noisy realization of an underlying tensor that exhibits a checkerbox structure modulo some unknown reordering along each of its modes. Suppose that there are K_1, K_2, and K_3 clusters along mode 1, 2, and 3 respectively. If the (i_1, i_2, i_3)th entry in $\boldsymbol{\mathcal{X}}$ belongs to the cluster defined by the r_1th mode-1 group, r_2th mode-2 group, and r_3th mode-3 group, then the observed tensor element $x_{i_1 i_2 i_3}$ is

$$x_{i_1 i_2 i_3} \quad = \quad c^*_{r_1 r_2 r_3} + \epsilon_{i_1 i_2 i_3}, \tag{10}$$

where $c^*_{r_1 r_2 r_3}$ is the mean of the co-cluster defined by the r_1th mode-1 partition, r_2th mode-2 partition, and r_3th mode-3 partition, and $\epsilon_{i_1 i_2 i_3}$ is the noise. Consequently, the observed tensor $\boldsymbol{\mathcal{X}}$ can be written as the sum of a mean tensor $\boldsymbol{\mathcal{U}}^* \in \mathbb{R}^{p_1 \times p_2 \times p_3}$, whose elements are expanded from the co-cluster means tensor $\boldsymbol{\mathcal{C}}^* \in \mathbb{R}^{K_1 \times K_2 \times K_3}$, and a noise tensor $\boldsymbol{\mathcal{E}} \in \mathbb{R}^{p_1 \times p_2 \times p_3}$. Figure 4 illustrates an underlying mean tensor $\mathcal{U}^*$ after permuting the slices along each of the modes to reveal a checkerbox structure. The co-clustering model in (10) is the 3-way analogue of the checkerboard mean model often employed in biclustering data matrices (Madeira and Oliveira, 2004; Chi et al., 2017).

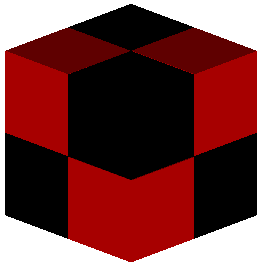

Figure 4: A third-order tensor with a checkerbox structure

Estimating model (10) consists of finding the partitions along each mode and finding the mean values of the $K_1K_2K_3$ co-clusters. The challenge is the first step, i.e., finding the partitions $\mathcal{G}_1, \mathcal{G}_2$, and $\mathcal{G}_3$, which denote the indices of the r_1th mode-1, r_2th mode-2, and r_3th mode-3 groups, respectively. Chi et al. (2018) proposed to solve a convex relaxation to the original combinatorial optimization problem, by simultaneously identifying the partitions along the modes of $\boldsymbol{\mathcal{X}}$ and estimating the co-cluster means through the optimization of the following convex objective function,

$$F_\gamma(\boldsymbol{\mathcal{U}}) \;=\; \frac{1}{2}\|\boldsymbol{\mathcal{X}}-\boldsymbol{\mathcal{U}}\|_{\mathrm{F}}^2 + \gamma \underbrace{\Big[R_1(\boldsymbol{\mathcal{U}}) + R_2(\boldsymbol{\mathcal{U}}) + R_3(\boldsymbol{\mathcal{U}})\Big]}_{R(\boldsymbol{\mathcal{U}})}, \tag{11}$$

where $R_1(\boldsymbol{\mathcal{U}}) = \sum_{i<j} w_{1,ij}\|\boldsymbol{\mathcal{U}}_{i::} - \boldsymbol{\mathcal{U}}_{j::}\|_{\mathrm{F}}$, $R_2(\boldsymbol{\mathcal{U}}) = \sum_{i<j} w_{2,ij}\|\boldsymbol{\mathcal{U}}_{:i:} - \boldsymbol{\mathcal{U}}_{:j:}\|_{\mathrm{F}}$, and $R_3(\boldsymbol{\mathcal{U}}) = \sum_{i<j} w_{3,ij}\|\boldsymbol{\mathcal{U}}_{::i} - \boldsymbol{\mathcal{U}}_{::j}\|_{\mathrm{F}}$. By seeking the minimizer $\widehat{\boldsymbol{\mathcal{U}}}_\gamma \in \mathbb{R}^{p_1\times p_2\times p_3}$ of (11), it casts co-clustering as a signal approximation problem, modeled as a penalized regression, to estimate the true co-cluster mean tensor $\boldsymbol{\mathcal{U}}^*$. The quadratic term in (11) quantifies how well $\boldsymbol{\mathcal{U}}$ approximates $\boldsymbol{\mathcal{X}}$, while the regularization term $R(\boldsymbol{\mathcal{U}})$ penalizes deviations away from a checkerbox pattern. The nonnegative parameter γ tunes the relative emphasis on these two terms and is selected via a BIC-type information criterion. The nonnegative weights $w_{d,ij}$ fine tunes the shrinkage of the slices along the dth mode. Chi et al. (2018) showed that the solution $\widehat{\boldsymbol{\mathcal{U}}}$ for (11) produces an entire solution path of checkerbox co-clustering estimates that varies continuously in γ, from the least smoothed model where $\widehat{\boldsymbol{\mathcal{U}}} = \boldsymbol{\mathcal{X}}$ and each tensor element occupies its own co-cluster, to the most smoothed model where all the elements of $\widehat{\boldsymbol{\mathcal{U}}}$ are identical and all tensor elements belong to a single co-cluster.

Tensor clustering via low-rank decomposition: We next review tensor clustering based on low-rank tensor decompositions (Papalexakis et al., 2013; Sun and Li, 2019). Unlike the convex tensor co-clustering of Chi et al. (2018) that targets a single tensor object, here we target the problem of clustering a collection of tensor samples.

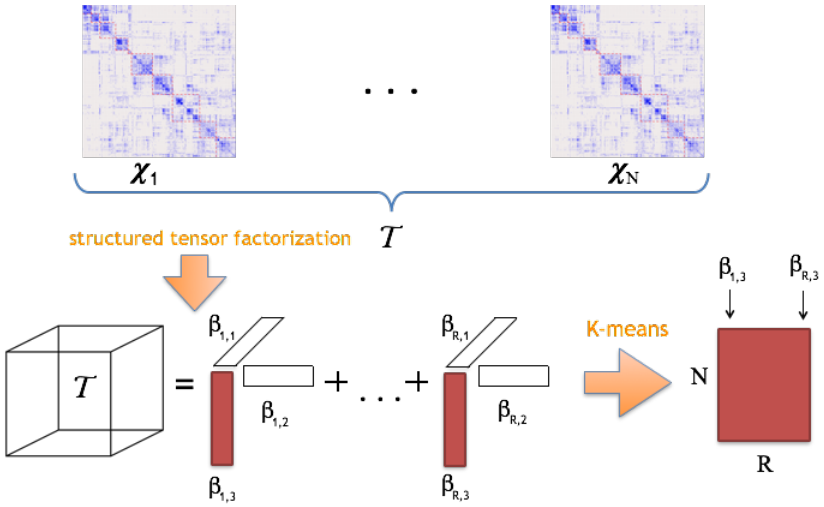

Figure 5: A schematic illustration of the low-rank tensor clustering method.

Given N copies of Dth-order tensors, $\boldsymbol{\mathcal{X}}_1, \ldots, \boldsymbol{\mathcal{X}}_N \in \mathbb{R}^{p_1 \times \ldots \times p_D}$, Papalexakis et al. (2013); Sun and Li (2019) aimed to uncover the underlying cluster structures of the N samples, with K clusters, and an equal number of $l = N/K$ samples per cluster, for simplicity. Sun and Li (2019) proposed to first stack all n tensor samples into a $(D+1)$th-order tensor $\boldsymbol{\mathcal{T}} \in \mathbb{R}^{p_1 \times \cdots \times p_D \times N}$, then consider a structured decomposition of $\boldsymbol{\mathcal{T}}$, and finally apply a usual clustering algorithm, e.g., K-means, to the matrix from the tensor decomposition that corresponds to the last mode to obtain the cluster assignment. Figure 5 shows a schematic plot of this method. Specifically, assume that the tensor $\boldsymbol{\mathcal{T}}$ is observed with noise, i.e., $\boldsymbol{\mathcal{T}} = \boldsymbol{\mathcal{T}}^* + \boldsymbol{\mathcal{E}}$, where $\boldsymbol{\mathcal{E}}$ is an error tensor, and $\boldsymbol{\mathcal{T}}^*$ is the true tensor with a rank-R CP decomposition structure, $\boldsymbol{\mathcal{T}}^* = \sum_{r=1}^{R} w_r^* \boldsymbol{\beta}_{r,1}^* \circ \cdots \circ \boldsymbol{\beta}_{r,D+1}^*$, where $\boldsymbol{\beta}_{r,j}^* \in \mathbb{R}^{p_j}$, $\|\boldsymbol{\beta}_{j,r}^*\|_2 = 1$, $w_r^* > 0$, $j = 1, \ldots, D+1, r = 1, \ldots, R$. Then the cluster structure of samples along the last mode of the tensor $\boldsymbol{\mathcal{T}}$ is fully determined by the matrix that stacks the decomposition components, i.e.,

$$\boldsymbol{B}_{D+1}^* = (\boldsymbol{\beta}_{1,D+1}^*, \ldots, \boldsymbol{\beta}_{R,D+1}^*) = \Big(\underbrace{\boldsymbol{\mu}_1^{*\top}, \ldots, \boldsymbol{\mu}_1^{*\top}}_{l \text{ samples}}, \ldots, \underbrace{\boldsymbol{\mu}_K^{*\top}, \ldots, \boldsymbol{\mu}_K^{*\top}}_{l \text{ samples}}\Big)^\top \in \mathbb{R}^{N \times R},$$

where $\boldsymbol{\mu}_k^* = (\mu_{1,k}^*, \ldots, \mu_{R,k}^*) \in \mathbb{R}^R$, $k = 1, \ldots, K$, indicates the cluster assignment. Accordingly, the true cluster means of the tensor samples $\boldsymbol{\mathcal{X}}_1, \ldots, \boldsymbol{\mathcal{X}}_N$ can be written as,

$$\underbrace{\boldsymbol{\mathcal{M}}_1 := \sum_{r=1}^{R} w_r^* \boldsymbol{\beta}_{r,1}^* \circ \cdots \circ \boldsymbol{\beta}_{r,D}^* \mu_{r,1}^*}_{\text{cluster center 1}}, \quad \ldots, \quad \underbrace{\boldsymbol{\mathcal{M}}_K := \sum_{r=1}^{R} w_r^* \boldsymbol{\beta}_{r,1}^* \circ \cdots \circ \boldsymbol{\beta}_{r,D}^* \mu_{r,K}^*}_{\text{cluster center K}}.$$

This reveals the key structure, i.e., each cluster mean is a linear combination of the outer

product of R rank-1 basis tensors, and all the cluster means share the same R basis tensors.

Sun and Li (2019) further introduced the sparsity and smoothness fusion structures in tensor decomposition to capture the sparsity and dynamic properties of the tensor samples. They proposed an optimization algorithm consisting of an unconstrained tensor decomposition step followed by a constrained optimization step. They established theoretical guarantee for their proposed dynamic tensor clustering approach, by deriving the corresponding non-asymptotic error bound, the rate of convergence, and the cluster recovery consistency.

Additional tensor clustering approaches: We briefly discuss some additional tensor clustering methods. Zhang et al. (2015a) unfolded tensor in each mode to construct an affinity matrix, then applied spectral clustering algorithm on this affinity matrix to obtain the cluster structure. Wu et al. (2016) utilized super-spacey random walk to propose a tensor spectral co-clustering algorithm for a nonnegative three-mode tensor. More recently, Luo and Zhang (2020) studied high-order clustering with planted structures for testing whether a cluster exists and identifying the support of cluster.

Future directions: In the model $\boldsymbol{\mathcal{T}} = \boldsymbol{\mathcal{T}}^* + \boldsymbol{\mathcal{E}}$ considered by Sun and Li (2019), no distributional assumption is imposed on the error tensor $\boldsymbol{\mathcal{E}}$. If one further assumes that $\boldsymbol{\mathcal{E}}$ is a standard Gaussian tensor, then the method reduces to a tensor version of Gaussian mixture model with identity covariance matrix. One possible future direction is to consider a more general tensor Gaussian mixture model with non-identity covariance matrix. The tensor cluster means and the covariance matrices can be estimated using a high-dimensional expectation-maximization algorithm (Hao et al., 2018), in which the maximization-step solves a penalized weighted least squares. Moreover, in the theoretical analysis of all the aforementioned tensor clustering projects, the true number of clusters was assumed to be given. It is of great interests to study the property of the tensor clustering when the number of cluster is estimated (Wang, 2010).

4.2 Tensor Graphical Model

Motivating examples: Tensor graphical model aims to characterize the dependency structure of the individual mode of the tensor-valued data. As an example, consider the microarray study for aging (Zahn et al., 2007), where multiple gene expression measurements are recorded on multiple tissue types of multiple mice with varying ages, which forms a set of third-order gene-tissue-age tensors. It is of scientific interest to study the dependency structure across different genes, tissues, and ages.

Gaussian graphical model: Similar to the vector-valued graphic model, He et al. (2014); Sun et al. (2015) assumed that the Dth-order tensor $\boldsymbol{\mathcal{T}} \in \mathbb{R}^{p_1 \times \ldots \times p_D}$ follows a tensor

normal distribution with zero mean and covariance matrices $\mathbf{\Sigma}_1, \ldots, \mathbf{\Sigma}_D$. Denote it by $\mathcal{T} \sim \mathrm{TN}(\mathbf{0}; \mathbf{\Sigma}_1, \ldots, \mathbf{\Sigma}_D)$, and its probability density function is given by

$$p(\mathcal{T}|\mathbf{\Sigma}_1, \ldots, \mathbf{\Sigma}_D) = (2\pi)^{-p/2} \left\{ \prod_{d=1}^{D} |\mathbf{\Sigma}_d|^{-p/(2p_d)} \right\} \exp\left(-\|\mathcal{T} \times \mathbf{\Sigma}^{-1/2}\|_F^2/2\right), \tag{12}$$

where $p = \prod_{d=1}^{D} p_d$, and $\mathbf{\Sigma}^{-1/2} = \left\{\mathbf{\Sigma}_1^{-1/2}, \ldots, \mathbf{\Sigma}_D^{-1/2}\right\}$. When $D = 1$, it reduces to the vector normal distribution with zero mean and covariance $\mathbf{\Sigma}_1$. Following Kolda and Bader (2009), $\mathcal{T} \sim \mathrm{TN}(\mathbf{0}; \mathbf{\Sigma}_1, \ldots, \mathbf{\Sigma}_D)$ if and only if $\mathrm{vec}(\mathcal{T}) \sim \mathrm{N}(\mathrm{vec}(\mathbf{0}); \mathbf{\Sigma}_D \otimes \ldots \otimes \mathbf{\Sigma}_1)$, where $\otimes$ denotes the Kronecker product.

Given n copies of i.i.d. samples $\mathcal{T}_1, \ldots, \mathcal{T}_n$ from $\mathrm{TN}(\mathbf{0}; \mathbf{\Sigma}_1^*, \ldots, \mathbf{\Sigma}_D^*)$, the goal of tensor graphical modeling is to estimate the true covariance matrices $\mathbf{\Sigma}_1^*, \ldots, \mathbf{\Sigma}_D^*$, and the corresponding true precision matrices $\mathbf{\Omega}_1^*, \ldots, \mathbf{\Omega}_D^*$, where $\mathbf{\Omega}_d^* = \mathbf{\Sigma}_d^{*-1}, d = 1, \ldots, D$. For identifiability, assume that $\|\mathbf{\Omega}_d^*\|_F = 1$ for $d = 1, \ldots, D$. This renormalization does not change the graph structure of the original precision matrix. A standard solution is the penalized maximum likelihood estimation which minimizes

$$\frac{1}{p}\mathrm{tr}[\mathbf{S}(\mathbf{\Omega}_D \otimes \ldots \otimes \mathbf{\Omega}_1)] - \sum_{d=1}^{D} \frac{1}{p_d} \log|\mathbf{\Omega}_d| + \sum_{d=1}^{D} P_{\lambda_d}(\mathbf{\Omega}_d), \tag{13}$$

where $\mathbf{S} = n^{-1} \sum_{i=1}^{n} \mathrm{vec}(\mathcal{T}_i)\mathrm{vec}(\mathcal{T}_i)^\top$, and $P_{\lambda_d}(\cdot)$ is a penalty function indexed by the tuning parameter λ_d. Adopting the usual lasso penalty used in vector graphical model, let $P_{\lambda_d}(\mathbf{\Omega}_d) = \lambda_d \|\mathbf{\Omega}_d\|_{1,\mathrm{off}}$, where $\|\cdot\|_{1,\mathrm{off}}$ means the sparsity penalty is applied to the off-diagonal elements of the matrix. The problem reduces to the classical sparse vector graphical model (Yuan and Lin, 2007; Friedman et al., 2008) when $D = 1$, and the sparse matrix graphical model (Leng and Tang, 2012; Yin and Li, 2012; Tsiligkaridis et al., 2013; Zhou, 2014) when $D = 2$. He et al. (2014) showed that the global minimizer of (13) enjoys nice theoretical properties.

Note that the objective function in (13) is bi-convex, in the sense that it is convex in terms of $\mathbf{\Omega}_d$ when the rest of $D - 1$ precision matrices are fixed. Exploring this bi-convex property, Sun et al. (2015) proposed to solve (13) by alternatingly updating one precision matrix while fixing the rest, which is equivalent to minimizing

$$\frac{1}{p_d}\mathrm{tr}(\mathbf{S}_d\mathbf{\Omega}_d) - \frac{1}{p_d}\log|\mathbf{\Omega}_d| + \lambda_d\|\mathbf{\Omega}_d\|_{1,\mathrm{off}}, \tag{14}$$

where $\mathbf{S}_d = p_d/(np) \sum_{i=1}^{n} \mathbf{V}_i^d \mathbf{V}_i^{d\top}$, $\mathbf{V}_i^d = \left[\mathcal{T}_i \times \left\{\mathbf{\Omega}_1^{1/2}, \ldots, \mathbf{\Omega}_{d-1}^{1/2}, 1_{p_d}, \mathbf{\Omega}_{d+1}^{1/2}, \ldots, \mathbf{\Omega}_D^{1/2}\right\}\right]_{(d)}$, $\times$ denotes the tensor product operation, and $[\cdot]_{(d)}$ denotes the mode-d matricization operation. Minimizing (14) corresponds to estimating the vector-valued Gaussian graphical model, which can be efficiently solved (Yuan and Lin, 2007; Friedman et al., 2008). Sun et al. (2015) further

showed that the estimator of their tensor lasso algorithm is able to achieve the desirable optimal statistical rates. In particular, their estimator $\widehat{\mathbf{\Omega}}_d$ satisfies

$$\left\|\widehat{\mathbf{\Omega}}_d - \mathbf{\Omega}_d^*\right\|_F = O_P\left(\sqrt{\frac{p_d(p_d+s_d)\log p_d}{np}}\right);\ \left\|\widehat{\mathbf{\Omega}}_d - \mathbf{\Omega}_d^*\right\|_\infty = O_P\left(\sqrt{\frac{p_d \log p_d}{np}}\right).$$

where $p = \prod_{d=1}^D p_d$ and the sparsity parameter s_d is the number of nonzero entries in the off-diagonal component of $\mathbf{\Omega}_d^*$. The above error bound implies that when the mode $D \geq 3$, the estimator from the tensor lasso algorithm can achieve estimation consistency even if we only have access to one observation, i.e., $n = 1$. This is because the estimation of the dth precision matrix takes advantage of the information from all other modes of the tensor data. This phenomenon only exists in tensor graphical model when $D \geq 3$, which reveals an interesting blessing of dimensionality phenomenon. Moreover, this rate is minimax-optimal since it is the best rate one can obtain even when $\mathbf{\Omega}_j^*$ $(j \neq d)$ were known.

As a follow-up, Lyu et al. (2019) further proposed a de-biased statistical inference procedure for testing hypotheses on the true support of the sparse precision matrices, and employed it for testing a growing number of hypothesis with false discovery rate (FDR) control. They also established the asymptotic normality of the test statistic and the consistency of the FDR controlled multiple testing procedure.

Variation in the Kronecker structure: In addition to the Kronecker product structure considered in (12), Greenewald et al. (2019) considered a Kronecker sum structure of $\Omega = \Psi_1 \oplus \Psi_2 = (\Psi_1 \otimes \mathbb{I}) + (\Psi_2 \otimes \mathbb{I})$. They showed that the new structure on the precision matrix leads to a non-separable covariance matrix that provides a richer model than the Kronecker product structure. Alternatively, Wang et al. (2020) proposed a Sylvester-structured graphical model to estimate precision matrices associated with tensor data, and used a Kronecker sum model for the square root factor of the precision matrix.

Future directions: All the existing works have assumed that the tensor data follows a tensor normal distribution. A natural future direction is to relax this normal distribution requirement, extend to the higher-order nonparanormal distribution (Liu et al., 2009), and utilize a robust rank-based likelihood estimation. When the order of the tensor is $D = 2$, it reduces to the semiparametric bigraphical model considered in Ning and Liu (2013).

5 Tensor Reinforcement Learning

The third topic we review is tensor reinforcement learning. Reinforcement learning (RL) is an area of machine learning that focuses on how an agent interacts with and takes actions in an environment in order to maximize the notion of cumulative rewards. It is a fast

growing field; see Sutton and Barto (2018) for a review and the references therein. We highlight two topics that involve tensor learning in RL: stochastic low-rank tensor bandit, and learning Markov decision process via tensor decomposition. In both cases, tensor methods serve as a powerful dimension reduction tool, which efficiently reduces the complexity of the reinforcement learning problems.

5.1 Stochastic Low-rank Tensor Bandit

Motivating examples: The growing availability of tensor data provides an unique opportunity for decision-makers to efficiently develop multi-dimensional decisions for individuals (Ge et al., 2016; Frolov and Oseledets, 2017; Bi et al., 2018; Song et al., 2019). For instance, consider a marketer who wants to design an advertising campaign for products with promotion offers across different marketing channels and user segments. This marketer needs to estimate the probability of user i clicking offer j in channel k for any (i, j, k) combination so that the most relevant users will be targeted for a chosen product and channel. Figure 6 gives a graphic illustration.

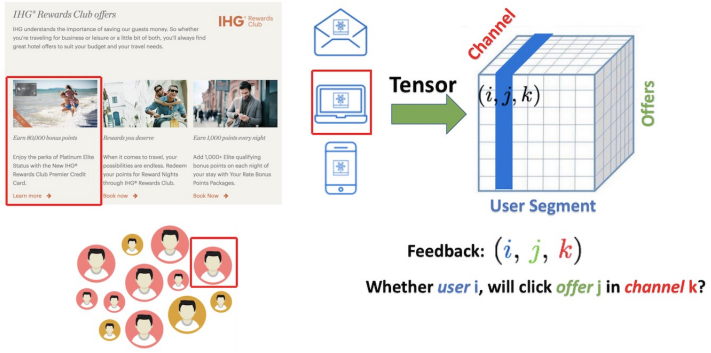

Figure 6: The tensor formulation of multi-dimensional advertising decisions.

Traditional static recommendation systems using tensor methods (Frolov and Oseledets, 2017; Bi et al., 2018; Song et al., 2019) do *not* interact with the environment to update the estimation. Besides, they usually suffer from cold-start in the absence of information from new customers, new products, or new contexts. An interactive recommendation system for multi-dimensional decisions is urgently needed.

Reinforcement learning offers a dynamic and interactive policy of recommendations. One of the fundamental problems in RL is the exploration-exploitation trade-off, in the sense that the agent must balance between exploiting existing information to accrue immediate reward, while investing in exploratory behavior that may increase future reward. Multi-armed bandit (Lattimore and Szepesvári, 2020) can be viewed as a simplified version of RL that

exemplifies this exploration-exploitation trade-off, and itself has plenty of applications in online advertising and operations research (Li et al., 2010). We review the problem of stochastic low-rank tensor bandit, a class of bandits whose mean reward can be represented as a low-rank tensor.

Low-rank tensor bandit problem formulation: We begin with a brief introduction of basic notations and concepts of multi-armed bandit. For more details, we refer to Lattimore and Szepesvári (2020). In the vanilla K-armed bandit, the agent interacts with the environment for n rounds. At round $t \in [n]$, the agent faces a multi-dimensional decision set $\boldsymbol{\mathcal{A}} \subseteq \mathbb{R}^{p_1 \times \ldots \times p_d}$, and the cardinality of $\boldsymbol{\mathcal{A}}$ can be either finite or infinite. The agent pulls an arm $I_t \in [K]$, and observes its reward y_{I_t}, which is drawn from a distribution associated with the arm I_t, denoted by P_{I_t} with an mean reward μ_{I_t}. It is important to point out in multi-armed bandit problems, the objective is to minimize the expected cumulative regret, which is defined as

$$R_n = n \max_{k \in [K]} \mu_k - \mathbb{E}\left[\sum_{t=1}^{n} y_t\right], \tag{15}$$

where the expectation is with respect to the randomness in the environment and policy.

Next, we introduce the low-rank tensor bandit. The classical vanilla multi-armed bandit can be treated as a special case of tensor bandit where the order of the tensor is one, and the action set $\boldsymbol{\mathcal{A}}$ only consists of canonical basis vectors, e.g., e_i that has 1 on its ith coordinate and 0 anywhere else. At round $t \in [n]$, based on historical information, the agent selects an action $\boldsymbol{\mathcal{A}}_t$ from $\boldsymbol{\mathcal{A}}$ and observes a noisy reward y_t, which can be written as

$$y_t = \langle \boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{A}}_t \rangle + \epsilon_t, \tag{16}$$

where $\boldsymbol{\mathcal{X}}$ is an unknown tensor parameter that admits a low-rank structure, and ϵ_t is a random noise. Model (16) can be viewed as a special case of the so-called stochastic linear bandit (Dani et al., 2008; Rusmevichientong and Tsitsiklis, 2010; Chu et al., 2011; Abbasi-Yadkori et al., 2011), where the mean reward can be parametrized into a linear form. However, naively implementing the existing linear bandit algorithms is to suffer high regret, since none of them utilizes the intrinsic low-rank structure of $\boldsymbol{\mathcal{X}}$.

At a glance, the tensor bandit model (16) looks similar to the tensor predictor regression model (3) in tensor supervised learning. However, the two have some fundamental distinctions. First, (16) considers a sequential setting, in the sense that $\boldsymbol{\mathcal{A}}_t$ has to be sequentially collected by the agent rather than given ahead. Consequently, $\boldsymbol{\mathcal{A}}_t$ and $\boldsymbol{\mathcal{A}}_{t-1}$ may be highly dependent, and the dependency structure is extremely difficult to characterize. By contrast, (3) can be viewed as corresponding to the offline setting where $\boldsymbol{\mathcal{A}}_t$ is fully observed. Second, instead of minimizing the mean square error as in tensor supervised learning, the objective in tensor

bandit is to minimize the cumulative regret,

$$R_n = \sum_{t=1}^{n} \langle \boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{A}}^* \rangle - \sum_{t=1}^{n} \langle \boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{A}}_t \rangle, \tag{17}$$

where $\boldsymbol{\mathcal{A}}^* = \text{argmax}_{\boldsymbol{\mathcal{A}}} \langle \boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{X}} \rangle$. As commonly observed in the bandit literature, even though $\boldsymbol{\mathcal{A}}^*$ may not be optimally estimated, the optimal regret is still achievable.

Rank-one bandit: Several existing RL methods can be categorized into the framework of (16), and they differ in terms of the structure of the action set $\boldsymbol{\mathcal{A}}$ and the assumptions placed on $\boldsymbol{\mathcal{X}}$. Particularly, Katariya et al. (2017b,a); Trinh et al. (2020) considered stochastic rank-1 matrix bandit, where $\boldsymbol{\mathcal{X}}$ is a rank-1 matrix and $\text{vec}(\boldsymbol{\mathcal{A}}_t)$ is a basis vector. The rank-1 structure greatly alleviates the difficulty of the problem, since one only needs to identify the largest values of the left-singular and right-singular vectors to find the largest entry of a non-negative rank-1 matrix. Alternatively, Katariya et al. (2017b,a) proposed special elimination-based algorithms, and Trinh et al. (2020) viewed rank-1 bandit as a special instance of unimodal bandit (Combes and Proutiere, 2014). However, neither of these solutions is applicable for general-rank matrices.

General-rank bandit: Kveton et al. (2017); Lu et al. (2018) studied the extension of stochastic general low-rank matrix bandit, and Hao et al. (2020) further generalized to stochastic low-rank tensor bandit. In particular, Kveton et al. (2017) relied on a strong hot-topic assumption on the mean reward matrix, and their algorithm was computationally expensive. Lu et al. (2018) utilized the ensemble sampling for low-rank matrix bandit, but did not provide any regret guarantee due to the theoretical challenges in handling sampling-based exploration. Hao et al. (2020) proposed a version of epoch-greedy algorithm (Langford and Zhang, 2008) and a tensor elimination algorithm to handle both data-poor regime and data-rich regime. The corresponding worse-case regret bounds were derived, though it is unclear if those bounds are optimal. In addition, Jun et al. (2019); Lu et al. (2020) studied stochastic contextual low-rank matrix bandit, where $\text{vec}(\boldsymbol{\mathcal{A}}_t)$ can be an arbitrary feature vector, and Hamidi et al. (2019) considered linear contextual bandit with a low-rank structure.

Future directions: The key principle to design an algorithm for low-rank tensor bandit is to efficiently utilize the low-rank information while balancing the exploration-exploitation trade-off. Unfortunately, there is no consensus about what types of algorithm can explore the low-rank information in both a provable and practical fashion. Actually, there is no direct upper confidence bound or Thompson sampling type algorithm for low-rank tensor bandit that is justified both empirically and theoretically for different structured bandit problems. The challenge is to construct a valid confidence bound, or the posterior distribution of a non-convex estimator in the sequential setting. In theory, although several regret upper

bounds have been derived (Jun et al., 2019; Hao et al., 2020; Lu et al., 2020), the minimax lower bound of low-rank tensor bandit remains unestablished.

5.2 Learning Markov Decision Process via Tensor Decomposition

Motivating examples: We next turn to full reinforcement learning of how an agent takes actions in an environment. A classical application is robotics, where a robot is to autonomously discover an optimal behavior through trial-and-error interactions with its environment; see (Kober et al., 2013) for a survey of reinforcement learning in robotics. In particularly, (Kober et al., 2013) noted that a key challenge facing robotics RL is the high dimensionality of both the action space and the state space, due to many degrees of freedom of modern anthropomorphic robots. Tensor methods again offer useful dimension reduction tools.

Dimension reduction of Markov decision process: Markov decision process (MDP) is a fundamental model in RL that characterizes the interactions between an agent and an environment. We first briefly introduce some basic notations about MDP. For more details, we refer to Puterman (2014). An instance of MDP $\boldsymbol{\mathcal{M}}$ can be specified by a tuple $(\boldsymbol{\mathcal{S}}, \boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{P}}, \boldsymbol{R})$, where $\boldsymbol{\mathcal{S}}$ and $\boldsymbol{\mathcal{A}}$ are the state and action spaces, $\boldsymbol{\mathcal{P}} \in \mathbb{R}^{|\boldsymbol{\mathcal{S}}|\times|\boldsymbol{\mathcal{A}}|\times|\boldsymbol{\mathcal{S}}|}$ is the transition probability tensor, $\boldsymbol{R} \in \mathbb{R}^{|\boldsymbol{\mathcal{S}}|\times|\boldsymbol{\mathcal{A}}|}$ is a matrix whose entries represent the reward after taking a certain action under a certain state. A policy $\pi \in \mathbb{R}^{|\boldsymbol{\mathcal{S}}|\times|\boldsymbol{\mathcal{A}}|}$ is a set of probability distributions on actions conditioned on each state. In addition, $|\boldsymbol{\mathcal{S}}| = p, |\boldsymbol{\mathcal{A}}| = q$.

In most applications, e.g., the robotics, the exact transition probability tensor of the MDP is unknown, and only a batch of empirical transition trajectories are available to the learner. Then one of the key tasks is to efficiently estimate the MDP transition tensor from the batch data. A challenge, however, is the scale of the data, which makes both model estimation and policy optimization intractable (Sutton and Barto, 2018).

Dimension reduction of MDP through matrix or tensor decompositions appears in a variety of RL solutions, including the Markov decision process with rich observations (Azizzadenesheli et al., 2016), the state aggregation model (Bertsekas et al., 2005; Zhang and Wang, 2019; Duan et al., 2019), the hidden Markov model (Hsu et al., 2012), among others.

Maximum likelihood estimation and Tucker decomposition: Ni and Wang (2019) proposed a joint dimension reduction method for both the action and state spaces of the MDP transition tensor through the Tucker decomposition (2),

$$\boldsymbol{\mathcal{P}} = [\![\widetilde{\boldsymbol{\mathcal{P}}}; \boldsymbol{U}_1, \boldsymbol{U}_2, \boldsymbol{U}_3]\!],$$

where $\widetilde{\boldsymbol{\mathcal{P}}} \in \mathbb{R}^{r_1\times r_2\times r_3}$ is the core tensor, and $\boldsymbol{U}_1 \in \mathbb{R}^{p\times r_1}, \boldsymbol{U}_2 \in \mathbb{R}^{q\times r_2}, \boldsymbol{U}_3 \in \mathbb{R}^{p\times r_3}$ are the factor matrices. The Tucker rank (r_1, r_2, r_3) can be viewed as the intrinsic dimension of the MDP. When $q = 1$, the MDP reduces to a Markov chain and the Tucker decomposition

reduces to the spectral decomposition of the Markov chain (Li et al., 2018a; Zhang and Wang, 2019). The factor matrices provide natural features for representing functions and operators on the action and state spaces, which can be applied together with feature-based reinforcement learning methods (Ernst et al., 2005).

A natural way to estimate the low-rank MDP transition tensor from the batch data is through maximum likelihood estimation. Suppose there are n independent state-action transition triplets $\{(s_k, a_k, s'_k)\}_{k\in[n]}$. For $1 \le s, s' \le p, 1 \le a \le q$, Define the empirical count as $n_{sas'} = \sum_{k=1}^{n} \mathbf{1}_{\{s_k=s, a_k=a, s'_k=s'\}}$. Given a fixed policy π, the negative log-likelihood based on the state-action transition triples $\{(s_k, a_k, s'_k)\}_{k\in[n]}$ is

$$L(P) = -\sum_{s=1}^{p}\sum_{a=1}^{q}\sum_{s'=1}^{p} n_{sas'} \log(P_{(s,a,s')}) + C,$$

where C is some constant unrelated with $\boldsymbol{\mathcal{P}}$. To estimate the MDP from sample transitions, Ni and Wang (2019) proposed the following Tucker-constrained maximum likelihood estimator,

$$\text{minimize } L(\boldsymbol{\mathcal{Q}}), \quad \text{such that } \boldsymbol{\mathcal{Q}}_{(\cdot,a,\cdot)}\mathbf{1}_p = \mathbf{1}_p, \ \text{Tuker-rank}(\boldsymbol{\mathcal{Q}}) \le (r_1, r_2, r_3), \ \text{and } a \in \boldsymbol{\mathcal{A}}.$$

Theoretically, Ni and Wang (2019) showed that the maximum likelihood estimator $\widehat{\boldsymbol{\mathcal{P}}}$ satisfies the following bound with a high probability,

$$\|\widehat{\boldsymbol{\mathcal{P}}} - \boldsymbol{\mathcal{P}}\|_F^2 \lesssim \left(\frac{\widetilde{p}^2\widetilde{r}^2}{n} + q\sqrt{\frac{\log(\widetilde{p})}{n}}\right), \tag{18}$$

where $\widetilde{p} = \max(p, q), \widetilde{r} = \max(r_1, r_2, r_3)$. The bound in (18) suggests that the estimation error is largely determined by the Tucker rank of the MDP instead of its actual dimension. This makes model compression possible with a limited number of data observations.

Future directions: Many questions in MDP remain open. For instance, it is unclear if the error bound (18) is minimax optimal. After obtaining the low-rnak representations of the MDP, it remains unclear how to embed them into the existing RL planning algorithms, and how the approximation error would affect the planning phase.

6 Tensor Deep Learning

The last topic we review is tensor deep learning. Deep learning represents a broad family of machine learning methods based on artificial neural networks (LeCun et al., 2015). It has received enormous attention in recent years thanks to its remarkable successes in a large variety of applications, including but not limited to image classification (Krizhevsky et al., 2012), speech recognition (Hinton et al., 2012), and game playing (Silver et al., 2016). We review two topics that connect tensors with deep learning: tensor-based compression of deep neural networks, and deep learning theory through tensor representation.

6.1 Tensor-based Deep Neural Network Compression

Motivating examples: Convolutional neural network (CNN) is perhaps the most common network structure in deep learning. It typically consists of a large number of convolutional layers, followed by a few fully-connected layers. Therefore, it often requires a vast number of parameters, and an enormous amount of training time even on the modern GPU clusters. For instance, the well-known VGG-19 network architecture (Simonyan and Zisserman, 2015) contains 10^8 parameters and requires over 15G floating-point operations to classify a single image. On the other hand, there is a growing interest to deploy CNNs on mobile devices, e.g., smartphones and self-driving cars, to implement real-time image recognition and conversational system. Unfortunately, the expensive computational cost, in both time and memory, of the standard CNN architectures prohibits their deployments on such devices. For that reason, there have recently emerged some promising works to speed up CNNs through tensor-based dimension reduction.

Recurrent neural network (RNN) is another common network structure in deep learning (Hochreiter and Schmidhuber, 1997). It is particularly suitable for modeling temporal dynamics, and has demonstrated excellent performance in sequential prediction tasks, e.g., speech recognition (Graves et al., 2013) and traffic forecasting (Li et al., 2018c). Despite of their effectiveness for smooth and short-term dynamics, however, it is difficult to generalize RNN to capture nonlinear dynamics and long-term temporal dependency. Moreover, the standard version of RNN and its memory-based extension such as the long short-term memory (LSTM) network suffer from an excessive number of parameters, making it difficult to train and also susceptible to overfitting.

Compression of convolutional layers of CNN : Denton et al. (2014); Lebedev et al. (2015); Tai et al. (2016) proposed low-rank approximations for the convolutional layers of CNN. Particularly, Lebedev et al. (2015) applied the CP decomposition (1) for the convolutional layers, while Kim et al. (2016) applied the Tucker decomposition (2) on the convolutional kernel tensors of a pre-trained network, then fine-tuned the resulting network. Meanwhile, which decomposition is better depends on the application domains, tasks, network architectures, and hardware constraints. Recognizing this issue, Hayashi et al. (2019) proposed to characterize a decomposition class specific to CNNs, by adopting a flexible hyper-graphical notion in tensor networks. This class includes modern light-weight CNN layers, such as the bottleneck layers in ResNet (He et al., 2016), the depth-wise separable layers in Mobilenet V1 (Howard et al., 2017), the inverted bottleneck layers in Mobilenet V2 (Sandler et al., 2018), among others. Moreover, this class can also deal with nonlinear activations by combining neural architecture search with the LeNet and ResNet architectures. Furthermore, Kossaifi et al. (2020b) introduced a tensor factorization framework for efficient multi-dimensional

convolutions of higher-order CNNs, with applications to spatiotemporal emotion estimation.

Compression of fully-connected layers of CNN : In a standard CNN architecture, the activation tensors of convolutional layers are first flattened, then connected to the outputs through fully connected layers. This step introduces a large number of parameters, and the flattening operation may also lose multimodal information. As an example, in the VGG-19 network architecture, about 80% of its parameters come from the fully-connected layers (Simonyan and Zisserman, 2015). Motivated by these observations, Novikov et al. (2015) applied the tensor-train decomposition, Ye et al. (2020) applied the block-term decomposition, and Kossaifi et al. (2020a) applied the Tucker decomposition, all focusing on reducing the number of parameters in the fully-connected layers.

Figure 7 provides an outline of the tensor-based CNN compression strategy from Kossaifi et al. (2020a). Built upon a standard CNN architecture, it consists of two new layers, a tensor contraction layer and a tensor regression layer, as the end-to-end trainable components of deep neural networks. After the standard convolutional layer and activation step, the tensor contraction layer reduces the dimensionality of the original activation tensor $\boldsymbol{\mathcal{X}}_i$ via a Tucker decomposition to obtain a core tensor $\boldsymbol{\mathcal{X}}'_i$. The tensor regression layer then directly associates $\boldsymbol{\mathcal{X}}'_i$ with the response y_i via a low-rank Tucker structure on the coefficient $\boldsymbol{\mathcal{B}}$, which helps avoid the flattening operation in the traditional fully-connected layer. All the parameters can be efficiently learned via end-to-end back-propagation.

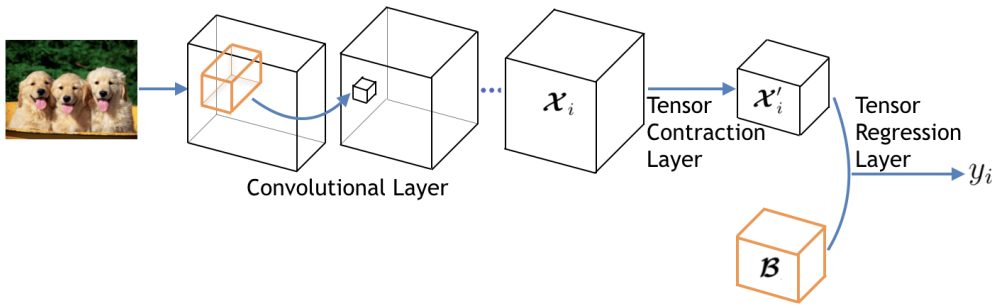

Figure 7: Illustration of the tensor-based CNN compression from Kossaifi et al. (2020a).

Compression of all layers of CNN: In addition to compression of the convolutional layers and fully-connected layers separately, there is the third category of compression methods targeting all layers. This enables to learn the correlations between different tensor dimensions. Moreover, the low-rank structure on the weight tensor acts as an implicit regularization, and can substantially reduce the number of parameters. Specifically, Kasiviswanathan et al. (2018) incorporated the randomized tensor sketching technique and developed a unified framework to approximate the operations of both the convolutional and fully connected layers

in CNNs. Kossaifi et al. (2019) proposed to fully parametrize all layers of CNNs with a single high-order low-rank tensor, where the modes of the tensor represent the architectural design parameters of the network, including the number of convolutional blocks, depth, number of stacks, and input features.

Compression of RNN: Yang et al. (2017); Yu et al. (2019); Su et al. (2020) utilized the tensor-train decomposition to efficiently learn the nonlinear dynamics of RNNs, by directly using high-order moments and high-order state transition functions. In addition, Ye et al. (2018) proposed a compact and flexible structure called the Block-Term tensor decomposition for dimension reduction in RNNs, and showed that it is not only more concise but also able to attain a better approximation to the original RNNs with much fewer parameters.

Future directions: Although the tensor-based DNN compression methods have shown great empirical success, the theoretical properties are still not yet fully understood. Moreover, the existing solutions have been focusing on the low-rank structure for dimension reduction. It is potentially useful to consider the additional sparsity structure, e.g., the sparse tensor factorization (Sun et al., 2017), to further reduce the number of parameters and to improve the interpretability of the tensor layers in CNNs or RNNs.

6.2 Deep Learning Theory through Tensor Methods

Motivating examples: Despite the wide empirical success of deep neural networks models, their theoretical properties are much less understood. Next, we review a few works that use tensor representations to facilitate the understanding of the expressive power, compressibility, generalizability, and other properties of deep neural networks.

Expressive power, compressibility and generalizability: Cohen et al. (2016) used tensor as an analytical tool to study the expressive power of deep neural networks, where the expressive power refers to the representation ability of a neural network architecture. They established an equivalence between the neural network and hierarchical tensor factorization, and showed that a shallow network corresponds to a rank-1 CP decomposition, whereas a deep network corresponds to a hierarchical Tucker decomposition. Through this connection, they further proved that, other than a measure zero negligible set, all functions that can be implemented by a deep network of the polynomial order would require an exponential order shallow network to realize. Built on this general tensor tool, various recent works have extended the study of expressive power to the overlapping architecture of deep learning (Sharir and Shashua, 2018), RNNs with multiplicative recurrent cells (Khrulkov et al., 2018), and RNNs with rectifier nonlinearities (Khrulkov et al., 2019).

Li et al. (2020) employed tensor analysis to derive a set of data dependent and easily measurable properties that tightly characterize the compressibility and generalizability of

neural networks. Specifically, the compressibility measures how much the original network can be compressed without compromising the performance on a training dataset more than a certain range. The generalizability measures the performance of a neural network on the unseen testing data. Compared to the generalization bounds via compression scheme (Arora et al., 2018), Li et al. (2020) provided a much tighter bound for the layer-wise error propagation, by exploiting the additional structures in the weight tensor of a neural network.

Additional connections: There are other connections between deep learning theory and tensors. Janzamin et al. (2015) provided a polynomial-time algorithm based on tensor decomposition for learning one-hidden-layer neural networks with twice-differential activation function and known input distributions. Moreover, Ge et al. (2018) considered learning a one-hidden-layer neural network and proved that the population risk of the standard squared loss implicitly attempts to decompose a sequence of low-rank tensors simultaneously. Mondelli and Montanari (2019) also established connections between tensor decomposition and the problem of learning a one-hidden-layer neural network with activation functions given by low-degree polynomials. They provided evidence that in certain regimes, and for certain data distributions the one-hidden-layer neural network cannot be learnt in polynomial time. So similar to Ge et al. (2018), they also considered the case when the data distribution is normal.

Future directions: Aforementioned works (Janzamin et al., 2015; Ge et al., 2018; Mondelli and Montanari, 2019) provide theoretical foundations for the connection between tensor decomposition and learning one-hidden-layer neural network. It is of interest to study how such a connection can be extended to more general deep neural network architectures and more general data distributions. It is also of interest to investigate if the theoretical results of Li et al. (2020) can be extended to study the compressibility and generalizability of more deep neural network architectures.

References

Abbasi-Yadkori, Y., Pál, D. and Szepesvári, C. (2011). Improved algorithms for linear stochastic bandits. In *Advances in Neural Information Processing Systems*.

Arora, S., Ge, R., Neyshabur, B. and Zhang, Y. (2018). Stronger generalization bounds for deep nets via a compression approach. In *35th International Conference on Machine Learning, ICML 2018*.

Azizzadenesheli, K., Lazaric, A. and Anandkumar, A. (2016). Reinforcement learning of pomdps using spectral methods. In *Proceedings of the 29th Annual Conference on Learning Theory (COLT2016)*.

Bertsekas, D. P., Bertsekas, D. P., Bertsekas, D. P. and Bertsekas, D. P. (2005). *Dynamic programming and optimal control*, vol. 1. Athena scientific Belmont, MA.

Bi, X., Qu, A., Shen, X. et al. (2018). Multilayer tensor factorization with applications to recommender systems. *The Annals of Statistics* **46** 3308–3333.

Bi, X., Tang, X., Yuan, Y., Zhang, Y. and Qu, A. (2020). Tensor in statistics. *Annual Review of Statistics and Its Application* **to appear**.

Chen, H., Raskutti, G. and Yuan, M. (2019). Non-convex projected gradient descent for generalized low-rank tensor regression. *The Journal of Machine Learning Research* **20** 172–208.

Chi, E. C., Allen, G. I. and Baraniuk, R. G. (2017). Convex biclustering. *Biometrics* **73** 10–19.

Chi, E. C., Gaines, B. R., Sun, W. W., Zhou, H. and Yang, J. (2018). Provable convex co-clustering of tensors. *arXiv preprint arXiv:1803.06518* .

Chu, W., Li, L., Reyzin, L. and Schapire, R. (2011). Contextual bandits with linear payoff functions. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*.

Clarkson, K. L. and Woodruff, D. P. (2017). Low-rank approximation and regression in input sparsity time. *Journal of the ACM (JACM)* **63** 1–45.

Cohen, N., Sharir, O. and Shashua, A. (2016). On the expressive power of deep learning: A tensor analysis. In *Conference on learning theory*.

Combes, R. and Proutiere, A. (2014). Unimodal bandits: Regret lower bounds and optimal algorithms. In *International Conference on Machine Learning*.

Dani, V., Hayes, T. P. and Kakade, S. M. (2008). Stochastic linear optimization under bandit feedback. In *21st Annual Conference on Learning Theory, COLT 2008*.

De Lathauwer, L., De Moor, B. and Vandewalle, J. (2000). On the best rank-1 and rank-(r 1, r 2,..., rn) approximation of higher-order tensors. *SIAM journal on Matrix Analysis and Applications* **21** 1324–1342.

Denton, E., Zaremba, W., Bruna, J., LeCun, Y. and Fergus, R. (2014). Exploiting linear structure within convolutional networks for efficient evaluation. In *Proceedings of the 27th International Conference on Neural Information Processing Systems*. NIPS'14.

Duan, Y., Ke, T. and Wang, M. (2019). State aggregation learning from markov transition data. In *Advances in Neural Information Processing Systems*.

Ermis, B., Acar, E. and Cemgil, A. T. (2015). Link prediction in heterogeneous data via generalized coupled tensor factorization. In *Data Mining and Knowledge Discovery*.

Ernst, D., Geurts, P. and Wehenkel, L. (2005). Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research* **6** 503–556.

Friedman, J., Hastie, H. and Tibshirani, R. (2008). Sparse inverse covariance estimation with the graphical Lasso. *Biostatistics* **9** 432–441.

Frolov, E. and Oseledets, I. (2017). Tensor methods and recommender systems. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* **7** e1201.

Ge, H., Caverlee, J. and Lu, H. (2016). Taper: A contextual tensor-based approach for personalized expert recommendation. In *Proceedings of the 10th ACM Conference on Recommender Systems*.

Ge, R., Lee, J. D. and Ma, T. (2018). Learning one-hidden-layer neural networks with landscape design. In *6th International Conference on Learning Representations, ICLR 2018*.

Graves, A., rahman Mohamed, A. and Hinton, G. (2013). Speech recognition with deep recurrent neural networks. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

Greenewald, K., Zhou, S. and Hero III, A. (2019). Tensor graphical lasso (teralasso). *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **81** 901–931.

Guhaniyogi, R., Qamar, S. and Dunson, D. B. (2017). Bayesian tensor regression. *The Journal of Machine Learning Research* **18** 2733–2763.

Hamidi, N., Bayati, M. and Gupta, K. (2019). Personalizing many decisions with high-dimensional covariates. In *Advances in Neural Information Processing Systems*.

Hao, B., Sun, W. W., Liu, Y. and Cheng, G. (2018). Simultaneous clustering and estimation of heterogeneous graphical models. *The Journal of Machine Learning Research* To Appear.

Hao, B., Wang, B., Wang, P., Zhang, J., Yang, J. and Sun, W. W. (2019). Sparse tensor additive regression. *arXiv preprint arXiv:1904.00479* .

Hao, B., Zhou, J., Wen, Z. and Sun, W. W. (2020). Low-rank tensor bandits. *arXiv preprint arXiv:2007.15788* .

Hastie, T. J. and Tibshirani, R. J. (1990). *Generalized additive models*, vol. 43. CRC press.

Hayashi, K., Yamaguchi, T., Sugawara, Y. and Maeda, S.-i. (2019). Exploring unexplored tensor network decompositions for convolutional neural networks. In *Advances in Neural Information Processing Systems*.

He, K., Zhang, X., Ren, S. and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*.

He, S., Yin, J., Li, H. and Wang, X. (2014). Graphical model selection and estimation for high dimensional tensor data. *Journal of Multivariate Analysis* **128** 165–185.

Hinton, G., Deng, L., Yu, D., Dahl, G. E., r. Mohamed, A., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T. N. and Kingsbury, B. (2012). Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine* **29** 82–97.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation* **9** 1735–1780.

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M. and Adam, H. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861* .

Hsu, D., Kakade, S. M. and Zhang, T. (2012). A spectral algorithm for learning hidden markov models. *Journal of Computer and System Sciences* **78** 1460–1480.

Jain, P. and Oh, S. (2014). Provable tensor factorization with missing data. In *Advances in Neural Information Processing Systems*.

Janzamin, M., Ge, R., Kossaifi, J. and Anandkumar, A. (2019). Spectral learning on matrices and tensors. *Foundations and Trends® in Machine Learning* **12** 393–536.

Janzamin, M., Sedghi, H. and Anandkumar, A. (2015). Beating the perils of non-convexity: Guaranteed training of neural networks using tensor methods. *arXiv preprint arXiv:1506.08473* .

Jun, K.-S., Willett, R., Wright, S. and Nowak, R. (2019). Bilinear bandits with low-rank structure. *arXiv preprint arXiv:1901.02470* .

Kanagawa, H., Suzuki, T., Kobayashi, H., Shimizu, N. and Tagami, Y. (2016). Gaussian process nonparametric tensor estimator and its minimax optimality. In *International Conference on Machine Learning*.

Kasiviswanathan, S. P., Narodytska, N. and Jin, H. (2018). Network approximation using tensor sketching. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*.

Katariya, S., Kveton, B., Szepesvári, C., Vernade, C. and Wen, Z. (2017a). Bernoulli rank-1 bandits for click feedback. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*.

Katariya, S., Kveton, B., Szepesvari, C., Vernade, C. and Wen, Z. (2017b). Stochastic rank-1 bandits. In *Artificial Intelligence and Statistics*.

Khrulkov, V., Hrinchuk, O. and Oseledets, I. (2019). Generalized tensor models for recurrent neural networks. In *International Conference on Learning Representations*.

Khrulkov, V., Novikov, A. and Oseledets, I. (2018). Expressive power of recurrent neural networks. In *International Conference on Learning Representations*.

Kim, Y.-D., Park, E., Yoo, S., Choi, T., Yang, L. and Shin, D. (2016). Compression of deep convolutional neural networks for fast and low power mobile applications. *International Conference on Learning Representations* .

Kober, J., Bagnell, J. A. and Peters, J. (2013). Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research* **32** 1238–1274.

Kolda, T. and Bader, B. (2009). Tensor decompositions and applications. *SIAM Review* **51** 455–500.

Kossaifi, J., Bulat, A., Tzimiropoulos, G. and Pantic, M. (2019). T-net: Parametrizing fully convolutional nets with a single high-order tensor. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Kossaifi, J., Lipton, Z. C., Khanna, A., Furlanello, T. and Anandkumar, A. (2020a). Tensor regression networks. *Journal of Machine Learning Research* 1–21.

Kossaifi, J., Toisoul, A., Bulat, A., Panagakis, Y., Hospedales, T. M. and Pantic, M. (2020b). Factorized higher-order cnns with an application to spatio-temporal emotion estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Krizhevsky, A., Sutskever, I. and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*. NIPS'12.

Kveton, B., Szepesvári, C., Rao, A., Wen, Z., Abbasi-Yadkori, Y. and Muthukrishnan, S. (2017). Stochastic low-rank bandits. *arXiv preprint arXiv:1712.04644* .

Langford, J. and Zhang, T. (2008). The epoch-greedy algorithm for multi-armed bandits with side information. In *Advances in neural information processing systems*.

Lattimore, T. and Szepesvári, C. (2020). *Bandit algorithms*. Cambridge University Press.

Lebedev, V., Ganin, Y., Rakhuba, M., Oseledets, I. and Lempitsky, V. (2015). Speeding-up convolutional neural networks using fine-tuned cp-decomposition. In *International Conference on Learning Representations*.

LeCun, Y., Bengio, Y. and Hinton, G. (2015). Deep learning. *Nature* **521** 436–444.

Leng, C. and Tang, C. (2012). Sparse matrix graphical models. *Journal of the American Statistical Association* **107** 1187–1200.

Li, J., Sun, Y., Su, J., Suzuki, T. and Huang, F. (2020). Understanding generalization in deep learning via tensor methods. *International Conference on Artificial Intelligence and Statistics* .

Li, L., Chu, W., Langford, J. and Schapire, R. E. (2010). A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*.

Li, L. and Zhang, X. (2017). Parsimonious tensor response regression. *Journal of the American Statistical Association* **112** 1131–1146.

Li, W., Liu, C.-C., Zhang, T., Li, H., Waterman, M. S. and Zhou, X. J. (2011). Integrative analysis of many weighted co-expression networks using tensor computation. *PLoS Comput Biol* **7** e1001106.

Li, X., Wang, M. and Zhang, A. (2018a). Estimation of markov chain via rank-constrained likelihood. In *35th International Conference on Machine Learning, ICML 2018*. International Machine Learning Society (IMLS).

Li, X., Xu, D., Zhou, H. and Li, L. (2018b). Tucker tensor regression and neuroimaging analysis. *Statistics in Biosciences* **10** 520–545.

Li, Y., Yu, R., Shahabi, C. and Liu, Y. (2018c). Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In *International Conference on Learning Representations*.

Li, Z., Suk, H.-I., Shen, D. and Li, L. (2016). Sparse multi-response tensor regression for alzheimer's disease study with multivariate clinical assessments. *IEEE Transactions on Medical Imaging* **35** 1927–1936.

Liu, H., Lafferty, J. and Wasserman, L. (2009). The nonparanormal: Semiparametric estimation of high dimensional undirected graphs. *Journal of Machine Learning Research* **10**.

Liu, Y., Yao, Q. and Li, Y. (2020). Generalizing tensor decomposition for n-ary relational knowledge bases. In *Proceedings of The Web Conference 2020*.

Lu, X., Wen, Z. and Kveton, B. (2018). Efficient online recommendation via low-rank ensemble sampling. In *Proceedings of the 12th ACM Conference on Recommender Systems*.

Lu, Y., Meisami, A. and Tewari, A. (2020). Low-rank generalized linear bandit problems. *arXiv preprint arXiv:2006.02948* .

Luo, Y. and Zhang, A. R. (2020). Tensor clustering with planted structures: Statistical optimality and computational limits. *arXiv preprint arXiv:2005.10743* .

Lyu, X., Sun, W. W., Wang, Z., Liu, H., Yang, J. and Cheng, G. (2019). Tensor graphical model: Non-convex optimization and statistical inference. *IEEE transactions on pattern analysis and machine intelligence* .

Ma, X., Zhang, P., Zhang, S., Duan, N., Hou, Y., Zhou, M. and Song, D. (2019). A tensorized transformer for language modeling. In *Advances in Neural Information Processing Systems*.

Madeira, S. C. and Oliveira, A. L. (2004). Biclustering algorithms for biological data analysis: A survey. *Computational Biology and Bioinformatics, IEEE/ACM Transactions on* **1** 24–45.

Mondelli, M. and Montanari, A. (2019). On the connection between learning two-layer neural networks and tensor decomposition. In *The 22nd International Conference on Artificial Intelligence and Statistics*.

Ni, C. and Wang, M. (2019). Maximum likelihood tensor decomposition of markov decision process. In *2019 IEEE International Symposium on Information Theory (ISIT)*. IEEE.

Ning, Y. and Liu, H. (2013). High-dimensional semiparametric bigraphical models. *Biometrika* **100** 655–670.

Novikov, A., Podoprikhin, D., Osokin, A. and Vetrov, D. (2015). Tensorizing neural networks. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*. NIPS'15.

Papalexakis, E. E., Sidiropoulos, N. D. and Bro, R. (2013). From K-Means to Higher-Way Co-Clustering: Multilinear Decomposition With Sparse Latent Factors. *IEEE Transactions on Signal Processing* **61** 493–506.

Puterman, M. L. (2014). *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons.

Rabanser, S., Shchur, O. and Günnemann, S. (2017). Introduction to tensor decompositions and their applications in machine learning. *arXiv preprint arXiv:1711.10781* .

Rabusseau, G. and Kadri, H. (2016). Low-rank regression with tensor responses. In *Advances in Neural Information Processing Systems*.

Raskutti, G., Yuan, M., Chen, H. et al. (2019). Convex regularization for high-dimensional multiresponse tensor regression. *The Annals of Statistics* **47** 1554–1584.

Ravikumar, P., Lafferty, J., Liu, H. and Wasserman, L. (2009). Sparse additive models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **71** 1009–1030.

Rendle, S. and Schmidt-Thieme, L. (2010). Pairwise interaction tensor factorization for personalized tag recommendation. In *International Conference on Web Search and Data Mining*.

Rusmevichientong, P. and Tsitsiklis, J. N. (2010). Linearly parameterized bandits. *Mathematics of Operations Research* **35** 395–411.

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A. and Chen, L.-C. (2018). Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*.

Sharir, O. and Shashua, A. (2018). On the expressive power of overlapping architectures of deep learning. In *International Conference on Learning Representations*.

Sidiropoulos, N. D., De Lathauwer, L., Fu, X., Huang, K., Papalexakis, E. E. and Faloutsos, C. (2017). Tensor decomposition for signal processing and machine learning. *IEEE Transactions on Signal Processing* **65** 3551–3582.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T. and Hassabis, D. (2016). Mastering the game of go with deep neural networks and tree search. *Nature* **529** 484–489.

Simonyan, K. and Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*.

Song, Q., Ge, H., Caverlee, J. and Hu, X. (2019). Tensor completion algorithms in big data analytics. *ACM Transactions on Knowledge Discovery from Data (TKDD)* **13** 1–48.

Su, J., Byeon, W., Huang, F., Kautz, J. and Anandkumar, A. (2020). Convolutional tensor-train lstm for spatio-temporal learning. *arXiv preprint arXiv:2002.09131* .

Sun, W. and Li, L. (2017). Sparse tensor response regression and neuroimaging analysis. *Journal of Machine Learning Research* **18** 4908–4944.

Sun, W., Lu, J., Liu, H. and Cheng, G. (2017). Provable sparse tensor decomposition. *Journal of the Royal Statistical Society, Series B* **79** 899–916.

Sun, W., Wang, Z., Liu, H. and Cheng, G. (2015). Non-convex statistical optimization for sparse tensor graphical model. *Advances in Neural Information Processing Systems* .

Sun, W. W. and Li, L. (2019). Dynamic tensor clustering. *Journal of the American Statistical Association* **114** 1894–1907.

Sutton, R. S. and Barto, A. G. (2018). *Reinforcement learning: An introduction*. MIT press.

Suzuki, T., Kanagawa, H., Kobayashi, H., Shimizu, N. and Tagami, Y. (2016). Minimax optimal alternating minimization for kernel nonparametric tensor learning. In *Advances in Neural Information Processing Systems*.

Tai, C., Xiao, T., Zhang, Y., Wang, X. and E, W. (2016). Convolutional neural networks with low-rank regularization. In *International Conference on Learning Representations*.

Trinh, C., Kaufmann, E., Vernade, C. and Combes, R. (2020). Solving bernoulli rank-one bandits with unimodal thompson sampling. In *Algorithmic Learning Theory*.

Trouillon, T., Dance, C. R., Gaussier, É., Welbl, J., Riedel, S. and Bouchard, G. (2017). Knowledge graph completion via complex tensor factorization. *The Journal of Machine Learning Research* **18** 4735–4772.

Tsiligkaridis, T., Hero, A. O. and Zhou, S. (2013). On convergence of Kronecker graphical Lasso algorithms. *IEEE Transactions on Signal Processing* **61** 1743–1755.

Vasilescu, M. and Terzopoulos, D. (2002). Multilinear analysis of image ensembles: Tensorfaces. In *European Conference on Computer Vision*.

Wang, J. (2010). Consistent selection of the number of clusters via cross validation. *Biometrika* **97** 893–904.

Wang, Y., Jang, B. and Hero, A. (2020). The sylvester graphical lasso (syglasso). In *International Conference on Artificial Intelligence and Statistics*.

Wu, T., Benson, A. R. and Gleich, D. F. (2016). General tensor spectral co-clustering for higher-order data. In *Advances in Neural Information Processing Systems*.

Yang, Y., Krompass, D. and Tresp, V. (2017). Tensor-train recurrent neural networks for video classification. In *International Conference on Machine Learning*.

Ye, J., Li, G., Chen, D., Yang, H., Zhe, S. and Xu, Z. (2020). Block-term tensor neural networks. *Neural Networks* 11–21.

Ye, J., Wang, L., Li, G., Chen, D., Zhe, S., Chu, X. and Xu, Z. (2018). Learning compact recurrent neural networks with block-term tensor decomposition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Yin, J. and Li, H. (2012). Model selection and estimation in the matrix normal graphical model. *Journal of Multivariate Analysis* **107** 119–140.

Yu, R. and Liu, Y. (2016). Learning from multiway data: Simple and efficient tensor regression. In *International Conference on Machine Learning*.

Yu, R., Zheng, S., Anandkumar, A. and Yue, Y. (2019). Long-term forecasting using higher-order tensor rnns. *arXiv preprint arXiv:1711.00073v2* .

Yuan, M. and Lin, Y. (2007). Model selection and estimation in the gaussian graphical model. *Biometrika* **94** 19–35.

Zahn, J., Poosala, S., Owen, A., Ingram, D. et al. (2007). AGEMAP: A gene expression database for aging in mice. *PLOS Genetics* **3** 2326–2337.

Zhang, A., Luo, Y., Raskutti, G. and Yuan, M. (2020). Islet: Fast and optimal low-rank tensor regression via importance sketching. *SIAM Journal on Mathematics of Data Science* **2** 444–479.

Zhang, A. and Wang, M. (2019). Spectral state compression of markov processes. *IEEE Transactions on Information Theory* **66** 3202–3231.

Zhang, C., Fu, H., Liu, S., Liu, G. and Cao, X. (2015a). Low-rank tensor constrained multiview subspace clustering. In *Proceedings of the IEEE international conference on computer vision*.

Zhang, X., Li, L., Zhou, H. and Shen, D. (2019). Tensor generalized estimating equations for longitudinal imaging analysis. *Statistica Sinica* **29** 1977–2005.

Zhang, Y., Duchi, J. and Wainwright, M. (2015b). Divide and conquer kernel ridge regression: A distributed algorithm with minimax optimal rates. *The Journal of Machine Learning Research* **16** 3299–3340.

Zhou, H., Li, L. and Zhu, H. (2013). Tensor regression with applications in neuroimaging data analysis. *Journal of the American Statistical Association* **108** 540–552.

Zhou, J., Sun, W. W., Zhang, J. and Li, L. (2020a). Partially observed dynamic tensor response regression. *arXiv preprint arXiv:2002.09735* .

Zhou, S. (2014). Gemini: Graph estimation with matrix variate normal instances. *Annals of Statistics* **42** 532–562.

Zhou, Y., Wong, R. K. W. and He, K. (2020b). Broadcasted nonparametric tensor regression. *arXiv preprint arXiv:2008.12927* .