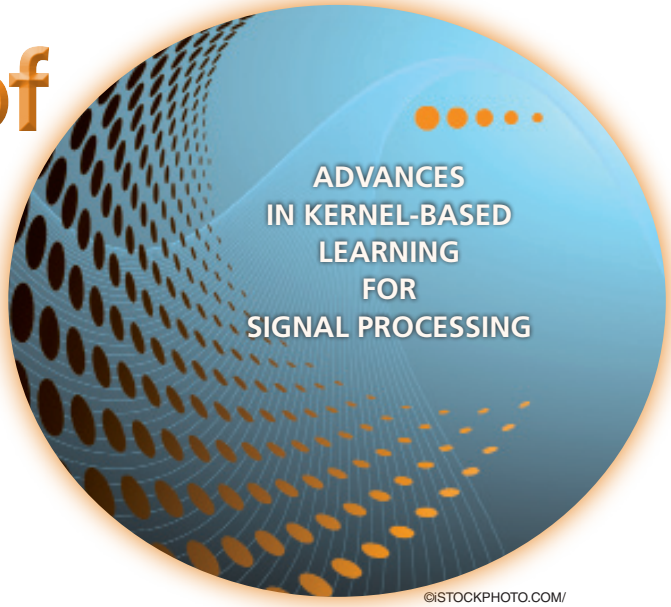
Qibin Zhao, Guoxu Zhou, Tülay Adali, Liqing Zhang, and Andrzej Cichocki

Kernelization of Tensor-Based Models for Multiway Data Analysis



©iSTOCKPHOTO.COM/
ALEKSANDAR VELASEVIC

Processing of multidimensional structured data

Tensors (also called multiway arrays) are a generalization of vectors and matrices to higher dimensions based on multilinear algebra. The development of theory and algorithms for tensor decompositions (factorizations) has been an active area of study within the past decade, e.g., [1] and [2]. These methods have been successfully applied to many problems in unsupervised learning and exploratory data analysis. Multiway analysis enables one to effectively capture the multilinear structure of the data, which is usually available as a priori information about the data. Hence, it might provide advantages over matrix factorizations by enabling one to more effectively use the underlying structure of the data. Besides unsupervised tensor decompositions, supervised tensor subspace regression and classification formulations have been also successfully applied to a variety of fields including chemometrics, signal processing, computer vision, and neuroscience.

There is a growing need for the development and application of machine-learning methods to analyze multidimensional data, such as functional magnetic resonance (fMRI), electrocorticography (ECoG), electroencephalography (EEG) data, and three-dimensional (3-D) video sequences. Tensors provide a natural and efficient way to describe such multiway data, and the corresponding learning methods can explicitly exploit the a priori information of data structure and capture the underlying multimode relations to achieve useful decompositions of the data with good generalization ability. In addition, tensor-based approaches for joint analysis on data ensemble generated from multiple factors or multiple sources, such as facial images from different people, pose, and illuminations, have been demonstrated to be powerful for multilinear dimension reduction, multitask classification, and image synthesis [3].

Kernel methods, on the other hand, have proven successful in many applications, providing an efficient way to solve nonlinear problems by mapping input data space into a high-dimensional feature space [4], where the problem becomes linearly solvable. Recent research has addressed the incorporation of the kernel concept into tensor decompositions [5]–[8], which aims to bring together the desirable properties of kernel methods and tensor decompositions for significant performance gain when the data are structured and nonlinear dependencies among latent variables do exist. Hence, kernel-based tensor decompositions promise to improve our ability to investigate multiway nonlinear dependencies among structured data.

Digital Object Identifier 10.1109/MSP.2013.2255334
Date of publication: 12 June 2013

In this article, we first review the principle of kernel machines and tensor decompositions, then present two fundamental models for kernel-based extension of tensor decompositions, together with an illustrative application. The key issue in developing a kernel-based framework for tensorial data is the kernel function with tensor-valued inputs, which can take multiway structure into account. We propose a family of tensor kernels based on multimode product kernels and probabilistic generative models and introduce two novel tensor-based learning methods, kernel tensor partial least squares (KTPLS) and kernel tensor canonical correlation analysis (KTCCA), which can be applied to nonlinear tensor-based regression and classification problems. In addition, we extend the Gaussian processes (GPs) model to a tensor-valued input space and provide a probabilistic interpretation of KTPLS and KTCCA using a GPs latent variable model. The effectiveness of these methods is demonstrated by concrete applications including reconstruction of 3-D movement trajectories from ECoG signals recorded from a monkey's brain, and human action classification based on video sequences.

KERNEL MACHINES

Kernel machines have gained considerable popularity during the last few decades, providing attractive solutions to a variety of problems including those in signal processing. In general, they have two parts: a module that performs a nonlinear mapping into the embedding or feature space implicitly through a kernel function and a specific learning algorithm in a dual form designed to discover linear relations in the feature space. The basic assumption in the development is that if two data points are close in the feature space they also have close outputs, hence the only information that is required is the similarity measure in the feature space, which leads us to avoid explicitly having to know the nonlinear mapping function. Instead, the similarity measure of two data points in the feature space, i.e., an inner product, should be appropriately defined by a reproducing kernel formulated in the input space, which is called a *kernel trick*. Kernel methods were introduced into machine learning in 1964 [9] and were subsequently applied successfully to many problems such as ridge regression, Fisher discriminant analysis, support vector machines (SVMs), partial least squares (PLS), and canonical correlation analysis (CCA) [4].

The main ingredients of kernel methods are elucidated through a nonlinear extension of principal component analysis (PCA), i.e., kernel PCA (KPCA) [10], [11]. PCA is a simple second-order approach that has been extensively applied for dimensionality reduction, feature extraction, data compression, and denoising [12], [45]. The principle is to seek a low-dimensional representation of the data such that the expected residual is as small as possible through projection onto mutually orthogonal directions of maximum variance. For instance, given a set of centered I-dimensional observations $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]$, the first principal component is defined as $y_n = \mathbf{w}^T\mathbf{x}_n$, where the parameter $\mathbf{w}$ can be estimated as the leading eigenvector of sample covariance matrix $\Sigma = (1/N)\mathbf{X}\mathbf{X}^T$ satisfying $\lambda\mathbf{w} = \Sigma\mathbf{w}$, which indicates that $\mathbf{w}$ can be also expressed as a linear combination of the training points, i.e., $\mathbf{w} = \sum_{n=1}^{N}\alpha_n\mathbf{x}_n = \mathbf{X}\boldsymbol{\alpha}$. Thus, the dual representation of PCA is $N\lambda\boldsymbol{\alpha} = \mathbf{K}\boldsymbol{\alpha}$ where $\mathbf{K} = \mathbf{X}^T\mathbf{X}$ with $k_{nn'} = \langle \mathbf{x}_n, \mathbf{x}_{n'}\rangle$ referred to as the *Gram matrix* with dimension $N \times N$. After estimation of $\boldsymbol{\alpha}$, the principal component of a novel example $\mathbf{x}^*$ is obtained by $y^* = \mathbf{w}^T\mathbf{x}^* = \sum_{n=1}^{N}\alpha_n\langle \mathbf{x}_n, \mathbf{x}^*\rangle$.

Let us now consider a nonlinear mapping $\phi : \mathbf{x} \in \mathbb{R}^I \longmapsto \phi(\mathbf{x}) \in \mathcal{H}$, that converts data into a feature space. Observe that for the dual representation of PCA, all information from training data is given by the Gram matrix $\mathbf{K}$ consisting of inner products between all pairs of training points. Hence, KPCA can be solved using the Gram matrix $\mathbf{K} = \phi^T(\mathbf{X})\phi(\mathbf{X})$ defined in the feature space with entries $k_{nn'} = \langle \phi(\mathbf{x}_n), \phi(\mathbf{x}_{n'})\rangle$, and the principal component of $\phi(\mathbf{x}^*)$ is computed by $\sum_{n=1}^{N}\alpha_n\langle \phi(\mathbf{x}_n), \phi(\mathbf{x}^*)\rangle = \boldsymbol{\alpha}^T\mathbf{k}^*$, where vector $\mathbf{k}^*$ is of size N with entries $k_n^* = \langle \phi(\mathbf{x}_n), \phi(\mathbf{x}^*)\rangle$. Note that the nonlinear mapping $\phi(\cdot)$ is used only in the form of inner products to yield the Gram matrix. By defining a kernel function in the original input space as an alternative way for inner products in feature space, i.e., $k(\mathbf{x}, \mathbf{x}') = \langle \phi(\mathbf{x}), \phi(\mathbf{x}')\rangle$, the Gram matrix $\mathbf{K}$ can be computed without explicit knowledge of $\phi(\cdot)$, and is thus also called the kernel matrix.

Kernel functions play a key role for nonlinear learning algorithms since they implicitly define the feature space, called the reproducing kernel Hilbert space (RKHS), and enable us to access the flexible high—or even infinite—dimensional feature spaces at low computational cost, which is particularly useful for addressing curse of dimensionality [4]. A popular kernel function is the Gaussian radial basis function (RBF) corresponding to an infinite feature space, defined by $k(\mathbf{x}, \mathbf{x}') = \exp[-(\|\mathbf{x} - \mathbf{x}'\|^2/2\beta^2)]$ with β controlling the width of the RBF kernel.

MULTILINEAR DATA ANALYSIS

For the development to follow, we first introduce the notation adopted in this article. Tensors are denoted by calligraphic letters, e.g., $\mathcal{X}$; matrices by boldface capital letters, e.g., $\mathbf{X}$; and vectors by boldface lowercase letters, e.g., $\mathbf{x}$. The order of a tensor is the number of dimensions, also knows as ways or modes. The element $(i_1, i_2, \ldots, i_M)$ of an Mth-order tensor $\mathcal{X}$ is denoted by $x_{i_1 i_2 \ldots i_M}$ or $(\mathcal{X})_{i_1 i_2 \ldots i_M}$, in which indices typically range from 1 to their capital version, e.g., $i_M = 1, 2, \ldots, I_M$. The mth element in a sequence is denoted by a superscript in parentheses, e.g., $\mathbf{U}^{(m)}$. Matricization, also known as unfolding, is the process of reordering the elements of a tensor into a matrix. More specifically, the mode-m matricization of a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_M}$ is denoted by $\mathbf{X}_{(m)} \in \mathbb{R}^{I_m \times I_1 \cdots I_{m-1} I_{m+1} \cdots I_M}$, while the vectorization of a tensor is denoted as $\mathrm{vec}(\mathcal{X})$. The inner product of two same-sized tensors $\mathcal{X}, \mathcal{X}'$ is defined by $\langle \mathcal{X}, \mathcal{X}'\rangle = \sum_{i_1 i_2 \ldots i_M} x_{i_1 i_2 \ldots i_M} x'_{i_1 i_2 \ldots i_M}$, and the squared Frobenius norm by $\|\mathcal{X}\|_F^2 = \langle \mathcal{X}, \mathcal{X}\rangle$. The m-mode product of a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_m \times \cdots \times I_M}$ with a matrix $\mathbf{U} \in \mathbb{R}^{J \times I_m}$ is denoted by $\mathcal{X} \times_m \mathbf{U}$ and is a tensor $\mathcal{Y}$ of size $I_1 \times \cdots \times I_{m-1} \times J \times I_{m+1} \times \cdots \times I_M$ defined by $(\mathcal{Y})_{i_1 \cdots i_{m-1} j i_{m+1} \cdots i_M} = \sum_{i_m=1}^{I_m} x_{i_1 i_2 \cdots i_M} u_{j i_m}$. For more detailed descriptions of multilinear algebra, see, e.g., Kolda and Bader [2]. The two popular tensor decompositions are the Tucker model and CANDECOMP/PARAFAC (CP) model, both of which can be regarded as higher-order generalizations of the matrix singular value decomposition (SVD) [1].

Let $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_M}$ denote an Mth-order tensor, then Tucker model, illustrated in Figure 1, is defined as [2]

$$\mathcal{X} = \mathcal{G} \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} \cdots \times_M \mathbf{U}^{(M)}, \tag{1}$$

where $\mathcal{G} \in \mathbb{R}^{R_1 \times \cdots \times R_M}$ is called the core tensor and $\mathbf{U}^{(m)} \in \mathbb{R}^{I_m \times R_m}$ denotes a factor matrix in m-mode. The matricized version of (1) is

$$\mathbf{X}_{(m)} = \mathbf{U}^{(m)} \mathbf{G}_{(m)} (\mathbf{U}^{(M)} \otimes \cdots \otimes \mathbf{U}^{(m+1)} \otimes \mathbf{U}^{(m-1)} \otimes \cdots \otimes \mathbf{U}^{(1)})^T, \tag{2}$$

where "$\otimes$" denotes the Kronecker product operator between matrices. Such a decomposition becomes more interesting as some specific constraints are imposed. For example, when all factor matrices $\{\mathbf{U}^{(m)}\}_{m=1}^{M}$ are columnwise orthonormal and the core tensor $\mathcal{G}$ is all-orthogonal (i.e., any subtensors are orthogonal, see definition in [13]) and ordered, this model is called higher-order singular value decomposition (HOSVD) [13], which provides us many useful properties. For instance, the core tensor can be simply computed by $\mathcal{G} = \mathcal{X} \times_1 \mathbf{U}^{(1)T} \times_2 \mathbf{U}^{(2)T} \cdots \times_M \mathbf{U}^{(M)T}$. A more restricted case of Tucker is the CP model where the number of components in all factor matrices is same (i.e., $R_1 = \cdots = R_M$) and the core tensor $\mathcal{G}$ is superdiagonal (i.e., $g_{r_1 \cdots r_M} \neq 0$ only if $r_1 = \cdots = r_M$) [2]. The CP model can be also defined as a sum of rank-one tensors

$$\mathcal{X} = \sum_{r=1}^{R} \lambda_r \mathbf{u}_r^{(1)} \circ \mathbf{u}_r^{(2)} \circ \cdots \circ \mathbf{u}_r^{(M)}, \tag{3}$$

where the symbol "$\circ$" represents the vector outer product. The rank of a tensor $\mathcal{X}$ is defined as the smallest number of rank-one tensors in an exact CP decomposition, i.e., $\text{rank}(\mathcal{X}) = R$ [2]. An important property of CP is its essential uniqueness up to the indeterminacy of scaling and permutation under mild assumptions.

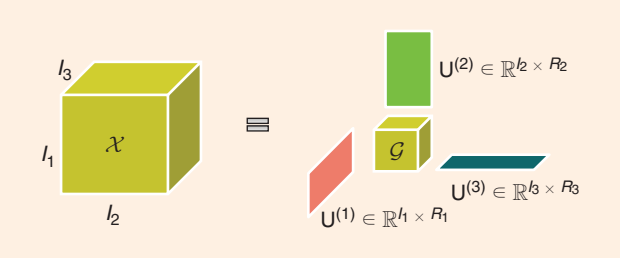
KERNEL-BASED TENSOR DECOMPOSITIONS

Tensor decompositions consist of several linear transformations collaboratively performed in different modes, using multilinear algebra, which enables us to capture the underlying interactions among multiple modes. As a result, their extension to capture nonlinear multimode interactions of data is highly desirable.

KERNEL-BASED MULTIFACTOR ANALYSIS

In practical applications, a tensor-based framework is suitable for analyzing a set of observations generated by multiple factors. For instance, facial images are affected by multiple factors such as belonging to different persons, and having different poses, illuminations, and viewpoints. Tensor representation is able to construct a multifactor structure of image ensembles while tensor decompositions allow us to extract multiple low-dimensional components in different modes, which can be employed for image recognition and image synthesis. This approach was successfully demonstrated by tensor faces [14] and was later extended to kernel-based multifactor analysis [3], [15].

Let $\mathcal{X} \in \mathbb{R}^{N_1 \times N_2 \times N_3 \times I}$ denote a fourth-order tensor constructed from a set of training samples affected by three factors,



[FIG1] The representation for a third-order Tucker decomposition.

where a mode-4 vector $(\mathcal{X})_{n_1 n_2 n_3 :}$ denoted by $\mathbf{x}^{(n_1, n_2, n_3)} \in \mathbb{R}^I$ is one data sample with factor indices of n_1, n_2, n_3. For example, facial images are generated from N_1 people, N_2 poses, and N_3 illuminations with each facial example denoted by an I-pixel image. The tensor $\mathcal{X}$ can be decomposed by HOSVD as

$$\mathcal{X} = \mathcal{G} \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} \times_3 \mathbf{U}^{(3)} \times_4 \mathbf{U}^{(4)}, \tag{4}$$

where $\mathcal{G} \times_4 \mathbf{U}^{(4)}$ can be written as a new core tensor $\mathcal{W} \in \mathbb{R}^{R_1 \times R_2 \times R_3 \times I}$ for simplicity and $\{\mathbf{U}^{(m)} \in \mathbb{R}^{N_m \times R_m}\}_{m=1}^{3}$ are orthogonal matrices. Thus, a training image $\mathbf{x}^{(n_1, n_2, n_3)}$ from the n_1th person, the n_2th pose, and the n_3th illumination condition is represented by

$$\mathbf{x}^{(n_1, n_2, n_3)} = \mathcal{W} \times_1 \mathbf{u}_{n_1}^{(1)T} \times_2 \mathbf{u}_{n_2}^{(2)T} \times_3 \mathbf{u}_{n_3}^{(3)T}, \tag{5}$$

where $\mathbf{u}_{n_1}^{(1)T}$ of size $1 \times R_1$ is the n_1th-row vector of $\mathbf{U}^{(1)}$, representing the low-dimensional components or coefficients with respect to the corresponding factor, i.e., n_1th person-identity. Similarly, $\mathbf{u}_{n_2}^{(2)T}, \mathbf{u}_{n_3}^{(3)T}$ are components with respect to n_2th pose and n_3th illumination conditions, respectively [3].

This model provides us with several interesting applications:

1) *Multitask classification*: Given a test sample of face image $\mathbf{x}^*$ without information on the person, pose, and illumination, we might seek the optimal components $\{\mathbf{u}_*^{(m)} \,|\, m = 1, 2, 3\}$ with fixed $\mathcal{W}$ according to (5), which are regarded as the low-dimensional features that can be used individually for different tasks such as face recognition, pose, and illumination classification, or can be used jointly for multitask classification.

2) *Robust face recognition*: If a priori knowledge about test sample $\mathbf{x}^*$ is available, such as n_2th pose and n_3th illumination, hence only $\mathbf{u}_*^{(1)}$ corresponding to person-identity needs to be optimized with the fixed $\mathbf{u}_{n_2}^{(2)}$ and $\mathbf{u}_{n_3}^{(3)}$ obtained from training data. This usually results in improved robustness for face recognition [15].

3) *Image synthesis*: For a new example $\mathbf{x}^*$, after all the components $\{\mathbf{u}_*^{(m)}\}_{m=1}^{3}$ have been estimated, $\mathbf{x}^*$ can be translated into any known pose and illumination by

$$\mathbf{y}^* = \mathcal{W} \times_1 \mathbf{u}_*^{(1)T} \times_2 \mathbf{u}_{n_2}^{(2)T} \times_3 \mathbf{u}_{n_3}^{(3)T}, \tag{6}$$

where $\mathbf{u}_{n_2}^{(2)}, \mathbf{u}_{n_3}^{(3)}$ are parameters obtained from training data, and $\mathbf{y}^*$ is a synthetic image representing the translation of $\mathbf{x}^*$ to the condition of n_2th pose and n_3th illumination [3].

This multifactor analysis model can be extended to its nonlinear version based on the kernel method. Let us assume that all data points are nonlinearly mapped to H-dimensional space by $\phi(\mathbf{x}^{(n_1,n_2,n_3)})$, implying that nonlinear mapping is performed on mode-4 vectors denoted by $\phi_4(\mathcal{X}) \in \mathbb{R}^{N_1\times N_2\times N_3\times H}$ or tensor $\tilde{\mathbf{\Phi}}$ for simplicity. Then HOSVD can be performed in the feature space as $\tilde{\mathbf{\Phi}} = \tilde{\mathcal{W}} \times_1 \tilde{\mathbf{U}}^{(1)} \times_2 \tilde{\mathbf{U}}^{(2)} \times_3 \tilde{\mathbf{U}}^{(3)}$, where the orthogonal factor matrices $\{\tilde{\mathbf{U}}^{(m)}\}_{m=1}^{3}$ are computed by the SVD of mode-m matricization $\tilde{\mathbf{\Phi}}_{(m)}\tilde{\mathbf{\Phi}}_{(m)}^T$ denoted by $\mathbf{K}^{(m)} \in \mathbb{R}^{N_m\times N_m}$, which consists of inner products of $\phi(\mathbf{x}^{(n_1,n_2,n_3)})$ and is regarded as mode-m kernel matrix, thus it can be computed using the kernel trick. For example, $\mathbf{K}^{(1)}$ is computed by

$$(\mathbf{K}^{(1)})_{n_1 n_1'} = \sum_{n_2=1}^{N_2}\sum_{n_3=1}^{N_3} k(\mathbf{x}^{(n_1,n_2,n_3)}, \mathbf{x}^{(n_1',n_2,n_3)}). \tag{7}$$

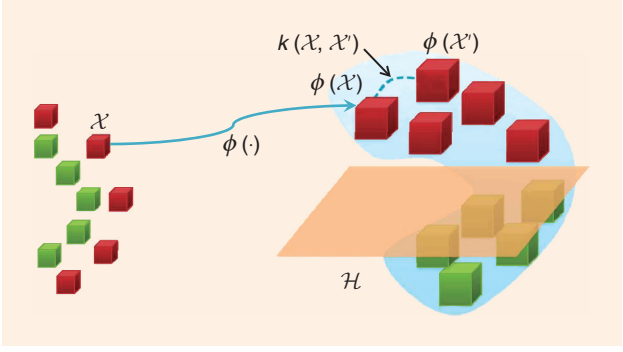
For a new face image $\mathbf{x}^*$, the goal is to seek its nonlinear multifactor latent representations $\{\tilde{\mathbf{u}}_*^{(m)}\}_{m=1}^{3}$ satisfying (5). To this end, we need to compute the projection $\tilde{\mathcal{W}} \times_4 \phi(\mathbf{x}^*)^T$, where the core tensor $\tilde{\mathcal{W}}$ can be represented as a multilinear combination of all data points in $\tilde{\mathbf{\Phi}}$, i.e.,

$$\tilde{\mathcal{W}} = \tilde{\mathbf{\Phi}} \times_1 \tilde{\mathbf{U}}^{(1)T} \times_2 \tilde{\mathbf{U}}^{(2)T} \times_3 \tilde{\mathbf{U}}^{(3)T}. \tag{8}$$

We obtain $\tilde{\mathcal{W}} \times_4 \phi(\mathbf{x}^*)^T = \mathcal{K}^* \times_1 \tilde{\mathbf{U}}^{(1)T} \times_2 \tilde{\mathbf{U}}^{(2)T} \times_3 \tilde{\mathbf{U}}^{(3)T}$, where $\mathcal{K}^* = \tilde{\mathbf{\Phi}} \times_4 \phi(\mathbf{x}^*)^T$ is a third-order kernel tensor with entries denoted as $(\mathcal{K}^*)_{n_1 n_2 n_3} = k(\mathbf{x}^{(n_1,n_2,n_3)}, \mathbf{x}^*)$, i.e., kernel evaluation between a new data point and all training data points. The details of the algorithm can be found in [15]. Similar to principal components in KPCA, $\{\tilde{\mathbf{u}}_*^{(m)}\}_{m=1}^{3}$ are multiple low-dimensional latent components extracted by nonlinear transformations, which may capture more complicated interaction information among multiple factors.

KERNEL-BASED FRAMEWORK FOR TENSORIAL DATA

The method described above is still based on vector representation and vector-based kernels, while the tensor structures allow us to exploit multilinear interaction among multiple factors. However, in many applications, different types of structural data such as images, videos, fMRI, and EEG data admit a natural tensorial representation. Next, we introduce a tensor-based model by encoding the structural information embedded in the multiway data into the kernels.



[FIG2] Tensor observations are mapped into RKHS space $\mathcal{H}$ by a nonlinear mapping function $\phi(\cdot)$. The kernel function is particularly defined as a similarity measure between two tensors.

Suppose we have N observations of third-order tensors $\{\mathcal{X}^{(n)} \in \mathbb{R}^{I_1\times I_2\times I_3}\}_{n=1}^{N}$, which can be concatenated as a fourth-order tensor $\mathcal{X} \in \mathbb{R}^{I_1\times I_2\times I_3\times N}$. We shall assume that the tensorial data points are mapped into the Hilbert space $\mathcal{H}$ by

$$\phi: \quad \mathcal{X}^{(n)} \longmapsto \phi(\mathcal{X}^{(n)}) \in \mathbb{R}^{H_1\times H_2\times H_3}. \tag{9}$$

For simplicity, we denote $\phi(\mathcal{X})$ by tensor $\mathbf{\Phi}$, and then perform HOSVD in the feature space according to (1), which can be rewritten as

$$\mathbf{\Phi} = \mathcal{G} \times_1 \mathbf{U}^{(1)} \cdots \times_4 \mathbf{U}^{(4)}, \quad \text{and} \quad \phi(\mathcal{X}^{(n)}) = \mathcal{W} \times_4 \mathbf{u}_n^{(4)T}, \tag{10}$$

where $\mathbf{U}^{(4)} \in \mathbb{R}^{N\times R}$ consists of R principal components with its nth-row vector denoted by $\mathbf{u}_n^{(4)T}$ and the core tensor $\mathcal{W} = \mathcal{G} \times_1 \mathbf{U}^{(1)} \cdots \times_3 \mathbf{U}^{(3)}$, regarded as tensor subspace basis, is obtained by $\mathbf{\Phi} \times_4 \mathbf{U}^{(4)T}$, implying that $\mathcal{W}$ can be represented as a linear combination of N observations $\{\phi(\mathcal{X}^{(n)})\}_{n=1}^{N}$. To avoid computing high-dimensional $\mathcal{W}$, we need to compute the factor matrix $\mathbf{U}^{(4)}$ by applying SVD on $\mathbf{\Phi}_{(4)}\mathbf{\Phi}_{(4)}^T$. Note that each element in $\mathbf{\Phi}_{(4)}\mathbf{\Phi}_{(4)}^T$ is an inner product between two mapped tensors in a vectorized form, indicating that it can be defined as a kernel matrix with (n, n')-entry denoted by

$$(\mathbf{K})_{nn'} = k(\text{vec}(\mathcal{X}^{(n)}), \text{vec}(\mathcal{X}^{(n')})), \tag{11}$$

where $k(\cdot,\cdot)$ could be any standard kernel function. Given a new example $\mathcal{X}^*$, the corresponding principal components can be obtained by $\mathbf{u}^* = \mathbf{U}^{(4)T}\mathbf{K}^{-1}\mathbf{k}^*$, where $(\mathbf{k}^*)_n = k(\mathcal{X}^{(n)}, \mathcal{X}^*)$. Note that this naive kernel in (11) is a simple way to generalize kernel function to tensors via the vectorization operation, but such kernel actually neglects the structural information conveyed by tensorial representations. To overcome this problem, we need to define special kernels for tensorial data.

KERNEL FUNCTION AS A SIMILARITY MEASURE BETWEEN TENSORS

The definition of kernels should take into account the a priori knowledge about invariance in the input space. For instance, translations and rotations of handwritten characters should leave their labels unchanged in a character recognition task, indicating that these transformed images, though distant in the original metric, should be close in the topology defined by the kernel. In this section, we discuss definition of kernels for tensor-valued inputs, which can take multiway structure into account for similarity measures (see Figure 2).

Although a number of kernels have been designed for structured objects, few approaches exploit the structure of tensorial representations. Recently, Signoretto et al. [5], [8] proposed a tensorial kernel exploiting algebraic geometry of spaces of tensors and a similarity measure between different

subspaces spanned by higher-order tensors. In addition, they showed that spaces of finite-dimensional tensors can be regarded as RKHSs associated to product kernels, while the Hilbert space of multilinear functions associated to general product kernels can be regarded as a space of infinite-dimensional tensors. There are some valid reproducing kernels toward a straightforward generalization to Mth-order tensors, such as the kernel functions $k: \mathcal{X} \times \mathcal{X} \longrightarrow \mathbb{R}$ given as [5]

$$\begin{aligned} \text{Linear kernel:} \quad & k(\mathcal{X}, \mathcal{X}') = \langle \text{vec}(\mathcal{X}), \text{vec}(\mathcal{X}') \rangle, \\ \text{Gaussian- RBF kernel:} \quad & k(\mathcal{X}, \mathcal{X}') = \exp\left(-\frac{1}{2\beta^2}\|\mathcal{X} - \mathcal{X}'\|_F^2\right). \end{aligned} \tag{12}$$

To define the similarity measure that directly exploits multilinear algebraic structure of input tensors, a product kernel can be defined by M factor kernels, e.g., $k(\mathcal{X}, \mathcal{X}') = \prod_{m=1}^{M} k(\mathbf{X}_{(m)}, \mathbf{X}'_{(m)})$, where each factor kernel represents a similarity measure between mode-m matricization of two tensors. One such similarity measure between matrices is Chordal distance (projection Frobenius norm) on the Grassmannian manifolds [5]. More specifically, let $\mathcal{X}$ denote an Mth-order tensor, when SVD is applied on mode-m unfoldings as $\mathbf{X}_{(m)} = \mathbf{U}_{\mathrm{X}}^{(m)} \Sigma_{\mathrm{X}}^{(m)} \mathbf{V}_{\mathrm{X}}^{(m)T}$, then the Chordal distance-based kernel for tensorial data can be defined as

$$k(\mathcal{X}, \mathcal{X}') = \prod_{m=1}^{M} \exp\left(-\frac{1}{2\beta_m^2}\| \mathbf{V}_{\mathrm{X}}^{(m)} \mathbf{V}_{\mathrm{X}}^{(m)T} - \mathbf{V}_{\mathrm{X}'}^{(m)} \mathbf{V}_{\mathrm{X}'}^{(m)T} \|_F^2\right). \tag{13}$$

This kernel provides us with rotation and reflection invariance for elements on the Grassmann manifold [5].

TENSOR KERNELS USING INFORMATION DIVERGENCE

Probabilistic kernels have been also investigated based on generative models and information divergences that are measures of dissimilarity between probability distributions, such as the Fisher kernel and Kullback-Leibler (KL) kernel [16]. The Fisher kernel assumes a generative model that thoroughly explains all data samples and maps each sample into a Fisher score computed by the gradient of the log-likelihood with respect to model parameters. Here we propose a new probabilistic kernel for tensors based on the assumption that each Mth-order tensor observation (e.g., $\mathcal{X}^{(n)} \in \mathbb{R}^{I_1 \times \cdots \times I_M}$) is considered individually as M different generative models. More specifically, mode-m matricization $\mathbf{X}_{(m)}^{(n)}$ is regarded as an ensemble of multivariate instances with dimensionality of I_m and number of instances of $I_1 I_2 \cdots I_{m-1} I_{m+1} \cdots I_M$, generated from a parametric model $p(\mathbf{x} \,|\, \boldsymbol{\Omega})$. In this manner, $\mathcal{X}^{(n)}$ has been successfully mapped into M-dimensional model-based probability distribution function space, i.e., $\{p(\mathbf{x} \,|\, \boldsymbol{\Omega}_m^{(n)}) \,|\, m = 1, \ldots, M\}$. Subsequently, the similarity measure between two tensors $\mathcal{X}$ and $\mathcal{X}'$ in mode-m is defined as

$$S_m(\mathcal{X} \,\|\, \mathcal{X}') = D(p(\mathbf{x} \,|\, \boldsymbol{\Omega}_m^{\mathcal{X}}) \,\|\, q(\mathbf{x} \,|\, \Omega_m^{\mathcal{X}'})), \tag{14}$$

where p, q represents probability density function for $\mathcal{X}$ and $\mathcal{X}'$ respectively, and $D(p \,\|\, q)$ is an information divergence between two distributions. One popular information divergence is the symmetric KL (sKL) divergence [16]

$$\begin{aligned} D_{sKL}(p(\mathbf{x} \,|\, \boldsymbol{\Omega}) \,\|\, q(\mathbf{x} \,|\, \boldsymbol{\Omega}')) = & \frac{1}{2}\int_{-\infty}^{+\infty} p(\mathbf{x} \,|\, \boldsymbol{\Omega}) \log \frac{p(\mathbf{x} \,|\, \boldsymbol{\Omega})}{q(\mathbf{x} \,|\, \boldsymbol{\Omega}')} d\mathbf{x} \\ & + \frac{1}{2}\int_{-\infty}^{+\infty} q(\mathbf{x} \,|\, \boldsymbol{\Omega}') \log \frac{q(\mathbf{x} \,|\, \boldsymbol{\Omega}')}{p(\mathbf{x} \,|\, \boldsymbol{\Omega})} d\mathbf{x}. \end{aligned} \tag{15}$$

Another possibility is the Jensen-Shannon (JS) divergence [17], [18] expressed by

$$D_{JS}(p \,\|\, q) = \frac{1}{2}\mathbf{KL}(p \,\|\, r) + \frac{1}{2}\mathbf{KL}(q \,\|\, r), \tag{16}$$

where $\mathbf{KL}(\cdot \,\|\, \cdot)$ denotes Kullback-Leibler divergence and $r(\mathbf{x}) = (p(\mathbf{x}) + q(\mathbf{x}))/2$. JS divergence can be interpreted as the average KL divergence between each probability distribution and the average distribution, or equivalently as the diversity of two distributions with equal priors. Finally, a probabilistic product kernel for tensors is defined as

$$k(\mathcal{X}, \mathcal{X}') = \alpha^2 \prod_{m=1}^{M} \exp\left(-\frac{1}{2\beta_m^2} S_m(\mathcal{X} \,\|\, \mathcal{X}')\right), \tag{17}$$

where α denotes a magnitude parameter and $[\beta_1, \ldots, \beta_M]$ are length-scales parameters. As isotropic RBF kernel, $\{\beta_m\}_{m=1}^{M}$ in (13) and (17) could also be the same. All kernel parameters are usually denoted by $\boldsymbol{\theta} = \{\alpha, \beta_m \,|\, m = 1, \ldots, M\}$. It can be shown that both sKL and JS divergences are nonnegative and equal to zero when $p(\mathbf{x}) = q(\mathbf{x})$, while they do not fulfill the triangle inequality [i.e., we do not have $D(p \,\|\, q) \le D(p \,\|\, r) + D(r \,\|\, q)$]. However, it has been proven in [17] and [18] that $[D_{sKL}(p \,\|\, q)]^{1/2}$ and $[D_{JS}(p \,\|\, q)]^{1/2}$ fulfill the triangle inequality thus is a metric, implying that the tensor kernel defined in (17) is a metric kernel. For simplicity, the Gaussian model assumption can be employed with model parameters including a mean vector and a full covariance matrix, i.e., $\boldsymbol{\Omega}_m = \{\boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m\}$ that can be estimated by the maximum likelihood from $\mathbf{X}_{(m)}$. The detailed algorithms using sKL and JS divergences between two multivariate Gaussian distributions are given in [16], [19], and [20]. In practice, because of the absence of closed-form solutions for probabilistic kernels, one may end up with a kernel matrix that is not positive-definite due to inaccuracies in the approximations.

The tensor kernels described above have some interesting properties. An intuitive interpretation is that Mth-order tensor observations are first mapped into an M-dimensional model space, then information divergence is applied as a similarity measure in the model space. Hence, such a kernel combines generative models with discriminative ones when used in conjunction with a specific discriminative method such as GPs. The probabilistic tensor kernels can deal with multiway data with missing values and variable lengths. Since it provides a way to model one tensor from M different viewpoints that correspond to different low-dimensional vector spaces, multiway relations can be captured in the similarity measure. Furthermore, the number of kernel parameters in (17) is much smaller than that

of an RBF kernel performed on unfolded tensors, hence making the tensor kernel less prone to overfitting.

KERNEL-BASED LEARNING ALGORITHMS FOR TENSOR-VALUED INPUTS

KERNEL-BASED TENSOR PLS REGRESSION

Kernel partial least squares (KPLS) has been successfully used in diverse applications both for regression and classification [21], [22]. For tensor input data, higher-order partial least squares (HOPLS) [23] was introduced with the goal of predicting a tensor $\mathcal{Y}$ from a tensor $\mathcal{X}$ through multilinear projection of the data onto the latent space followed by regression against the latent variables. Here, we introduce kernel-based tensor PLS (KTPLS) as an extension of HOPLS to kernel spaces.

Given N pairs of tensor observations $\{(\mathcal{X}^{(n)}, \mathcal{Y}^{(n)})\}_{n=1}^{N}$, $\mathcal{X}^{(n)}$ denotes an Mth-order independent tensor and $\mathcal{Y}^{(n)}$ denotes an Lth-order dependent tensor, which can be concatenated to form an $(M+1)$th-order tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{N \times I_1 \times \cdots \times I_M}$ and $(L+1)$th-order tensor $\boldsymbol{\mathcal{Y}} \in \mathbb{R}^{N \times J_1 \times \cdots \times J_L}$. We then let $\boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{Y}}$ to be mapped into the Hilbert space as described by (9). For simplicity, we denote $\phi(\boldsymbol{\mathcal{X}})$ by $\boldsymbol{\Phi}$ and $\phi(\boldsymbol{\mathcal{Y}})$ by $\boldsymbol{\Psi}$. KTPLS finds tensor decompositions such that

$$\begin{aligned} \boldsymbol{\Phi} &= \mathcal{G}_{\mathcal{X}} \times_1 \mathbf{T} \times_2 \mathbf{P}^{(1)} \cdots \times_{M+1} \mathbf{P}^{(M)} + \mathcal{E}_{\mathcal{X}}, \\ \boldsymbol{\Psi} &= \mathcal{G}_{\mathcal{Y}} \times_1 \mathbf{U} \times_2 \mathbf{Q}^{(1)} \cdots \times_{L+1} \mathbf{Q}^{(L)} + \mathcal{E}_{\mathcal{Y}}, \\ \mathbf{U} &= \mathbf{TD} + \mathbf{E}_U, \end{aligned} \tag{18}$$

where $\{\mathbf{P}^{(m)}, \mathbf{Q}^{(l)}\}$ denote factor matrices and $\{\mathcal{G}_{\mathcal{X}}, \mathcal{G}_{\mathcal{Y}}\}$ denote core tensors. $\mathcal{E}_{\mathcal{X}}, \mathcal{E}_{\mathcal{Y}}$, and $\mathbf{E}_U$ represent residuals and errors. Here, $\mathbf{D}$ is a diagonal matrix denoting the inner relation between latent vectors. Since $\mathcal{G}_{\mathcal{X}} \times_2 \mathbf{P}^{(1)} \cdots \times_{M+1} \mathbf{P}^{(M)}$ denoted by $\tilde{\mathcal{G}}_{\mathcal{X}}$ and $\mathcal{G}_{\mathcal{Y}} \times_2 \mathbf{Q}^{(1)} \cdots \times_{L+1} \mathbf{Q}^{(L)}$ denoted by $\tilde{\mathcal{G}}_{\mathcal{Y}}$ can be represented as a linear combination of $\{\phi(\mathcal{X}^{(n)})\}$ and $\{\phi(\mathcal{Y}^{(n)})\}$ respectively, i.e., $\tilde{\mathcal{G}}_{\mathcal{X}} = \boldsymbol{\Phi} \times_1 \mathbf{T}^T$ and $\tilde{\mathcal{G}}_{\mathcal{Y}} = \boldsymbol{\Psi} \times_1 \mathbf{U}^T$, we only need to explicitly find the latent vectors of $\mathbf{T} = [\mathbf{t}_1, \ldots, \mathbf{t}_R]$ and $\mathbf{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_R]$ with pairwise maximum covariance through solving an optimization problem sequentially by applying deflation, which is expressed by

$$\max_{\{\mathbf{w}_r^{(m)}, \mathbf{v}_r^{(l)}\}} [\mathrm{cov}(\mathbf{t}_r, \mathbf{u}_r)]^2, r = 1, 2, ..., \mathrm{R}, \tag{19}$$

where

$$\mathbf{t}_r = \boldsymbol{\Phi} \times_2 \mathbf{w}_r^{(1)T} \cdots \times_{M+1} \mathbf{w}_r^{(M)T}, \ \mathbf{u}_r = \boldsymbol{\Psi} \times_2 \mathbf{v}_r^{(1)T} \cdots \times_{L+1} \mathbf{v}_r^{(L)T}.$$

Rewriting (19) in matrix form, it becomes $\mathbf{t}_r = \boldsymbol{\Phi}_{(1)} \tilde{\mathbf{w}}_r$, $\mathbf{u}_r = \boldsymbol{\Psi}_{(1)} \tilde{\mathbf{v}}_r$, which can be solved by a kernelized version of the eigenvalue problem, i.e., $\boldsymbol{\Phi}_{(1)} \boldsymbol{\Phi}_{(1)}^T \boldsymbol{\Psi}_{(1)} \boldsymbol{\Psi}_{(1)}^T \mathbf{t}_r = \lambda \mathbf{t}_r$ and $\mathbf{u}_r = \boldsymbol{\Psi}_{(1)} \boldsymbol{\Psi}_{(1)}^T \mathbf{t}_r$ [21]. Note that $\boldsymbol{\Phi}_{(1)} \boldsymbol{\Phi}_{(1)}^T$ contains only the inner products between vectorized input tensors, which can be replaced by an $N \times N$ kernel matrix $\mathbf{K}_{\mathcal{X}}$. Thus, we have $\mathbf{K}_{\mathcal{X}} \mathbf{K}_{\mathcal{Y}} \mathbf{t}_r = \lambda \mathbf{t}_r$ and $\mathbf{u}_r = \mathbf{K}_{\mathcal{Y}} \mathbf{t}_r$. To take the multilinear structure into account, the kernel matrices should be computed using the kernel functions for tensors, i.e., $(\mathbf{K}_{\mathcal{X}})_{nn'} = k(\mathcal{X}^{(n)}, \mathcal{X}^{(n')})$ and $(\mathbf{K}_{\mathcal{Y}})_{nn'} = k(\mathcal{Y}^{(n)}, \mathcal{Y}^{(n')})$. Finally, the prediction of a novel data point $\mathcal{X}^*$ can be achieved by [21]

$$\mathbf{y}^{*T} = \mathbf{k}^{*T} \mathbf{U} (\mathbf{T}^T \mathbf{K}_{\mathcal{X}} \mathbf{U})^{-1} \mathbf{T}^T \mathbf{Y}_{(1)}, \tag{20}$$

where $(\mathbf{k}^*)_n = k(\mathcal{X}^{(n)}, \mathcal{X}^*)$ and $\mathbf{y}^{*T}$ should be reorganized to tensor form $\mathcal{Y}^*$.

The significance of (20) can be explained in several ways. First, it is a linear combination of N observations $\{\mathcal{Y}^{(n)}\}$ with the coefficients $\mathbf{k}^{*T} \mathbf{U} (\mathbf{T}^T \mathbf{K}_{\mathcal{X}} \mathbf{U})^{-1} \mathbf{T}^T$; the second interpretation is that y_j^* is predicted by a linear combination of N kernels, each one centered on a training point, i.e., $y_j^* = \sum_{n=1}^{N} \alpha_n k(\mathcal{X}^{(n)}, \mathcal{X}^*)$, where $\alpha_n = (\mathbf{U} (\mathbf{T}^T \mathbf{K}_{\mathcal{X}} \mathbf{U})^{-1} \mathbf{T}^T \mathbf{Y}_{(1)})_{nj}$. Finally, a third interpretation is that $\mathbf{t}^*$ is obtained by nonlinearly projecting $\mathcal{X}^*$ onto the latent space, i.e., $\mathbf{t}^{*T} = \mathbf{k}^{*T} \mathbf{U} (\mathbf{T}^T \mathbf{K}_{\mathcal{X}} \mathbf{U})^{-1}$, then $\mathbf{y}^{*T}$ is predicted by a linear regression against $\mathbf{t}^*$, i.e., $\mathbf{y}^{*T} = \mathbf{t}^{*T} \mathbf{C}$ where regression coefficient is $\mathbf{C} = \mathbf{T}^T \mathbf{Y}_{(1)}$. In general, to ensure the strict linear relationship between latent vectors and output in original spaces, the kernel function on data $\boldsymbol{\mathcal{Y}}$ is restricted to linear kernels.

KERNEL-BASED TENSOR CCA

CCA can be also used to determine the linear relationship between two sets of observations. In essence, CCA finds the directions of maximum correlation while PLS finds the directions of maximum covariance. Covariance and correlation are two different statistical measures for quantifying how variables covary. CCA is a popular choice for many applications, such as supervised dimensionality reduction, multiview learning, and multilabel classification [24]. Kernel canonical correlation analysis (KCCA) has been proposed in [25] and [26], and its multilinear extension for third-order tensors in [27].

Here, we introduce kernel-based tensor CCA (KTCCA) as a multiway extension of KCCA. Given N observations of two random tensors denoted as $\boldsymbol{\mathcal{X}}$ and $\boldsymbol{\mathcal{Y}}$, the objective function of KTCCA can be written as

$$\max_{\{\mathbf{w}, \mathbf{v}\}} \frac{\mathbf{w}^T \mathbf{K}_{\mathcal{X}} \mathbf{K}_{\mathcal{Y}} \mathbf{v}}{\sqrt{\mathbf{w}^T \mathbf{K}_{\mathcal{X}}^2 \mathbf{w} \mathbf{v}^T \mathbf{K}_{\mathcal{Y}}^2 \mathbf{v}}}, \tag{21}$$

where the kernel matrices $\mathbf{K}_{\mathcal{X}}, \mathbf{K}_{\mathcal{Y}}$ are computed using the kernel function for tensorial data as defined in (13) and (17), $\{\mathbf{w}, \mathbf{v}\} \in \mathbb{R}^N$ are weight coefficients in kernel space corresponding to $\boldsymbol{\mathcal{X}}$ and $\boldsymbol{\mathcal{Y}}$, respectively. The solution can be obtained by solving $\mathbf{w} = (1/\lambda) \mathbf{K}_{\mathcal{X}}^{-1} \mathbf{K}_{\mathcal{Y}} \mathbf{v}, \mathbf{K}_{\mathcal{Y}}^2 \mathbf{v} - \lambda^2 \mathbf{K}_{\mathcal{Y}}^2 \mathbf{v} = \mathbf{0}$, which has the same form as KCCA [26].

GAUSSIAN PROCESSES IN TENSOR VARIATE SPACE

TENSOR-BASED GAUSSIAN PROCESS REGRESSION

GPs are a class of probabilistic models that specify a distribution over functions and perform inference directly in the function space [28]. We introduce a GP model in tensor-valued input space, called tensor-based GPs (tensor-GPs), by taking into account the tensor structure of data. Given a set of paired observations $\{\mathcal{X}^{(n)}, y^{(n)} | n = 1, \ldots, N\}$, the tensor regression model is defined as $y^{(n)} = f(\mathcal{X}^{(n)}) + \epsilon$, where, for example for multilinear

regression, the latent function can be represented as $f(\mathcal{X}) = \mathcal{X} \times_1 \mathbf{w}^{(1)T} \times \cdots \times_M \mathbf{w}^{(M)T}$. For nonlinear extensions, GP prior for latent function is specified by

$$f(\mathcal{X}) \sim \mathcal{GP}(\mu(\mathcal{X}), k(\mathcal{X}, \mathcal{X}')), \tag{22}$$

where $\mu(\mathcal{X})$ denotes mean function and covariance function $k(\mathcal{X}, \mathcal{X}')$ with tensor-valued inputs is predefined by a tensor kernel. Thus, assuming additive independent and identically distributed (i.i.d.) Gaussian noise with variance σ^2, the likelihood of noisy observations becomes $\mathbf{y} \sim \mathcal{N}(\mathbf{0}, \mathbf{K} + \sigma^2\mathbf{I})$. The predictive distribution of y^* corresponding to $\mathcal{X}^*$ can be inferred as $y^* \mid \mathcal{X}^*, \mathcal{X}, \mathbf{y} \sim \mathcal{N}(\bar{y}^*, \mathrm{cov}(y^*))$ where $\bar{y}^* = \mathbf{k}^{*T}(\mathbf{K} + \sigma^2\mathbf{I})^{-1}\mathbf{y}$ and $\mathrm{cov}(y^*) = k(\mathcal{X}^*, \mathcal{X}^*) - \mathbf{k}^{*T}(\mathbf{K} + \sigma^2\mathbf{I})^{-1}\mathbf{k}^* + \sigma^2$ [28]. Comparing this expression with (20), we see that the relationship between GP and KTPLS regression is that GP computes an exact inverse of kernel matrix with noise variance added to the diagonal, while KTPLS approximates kernel matrix through projections on the latent vectors $\mathbf{T}$ and $\mathbf{U}$ and then computes its inverse.

The covariance function for tensors defined in (17) contains hyperparameters that can be learned from observed data based on maximum a posterior (MAP) estimation, given by

$$\{\hat{\boldsymbol{\theta}}, \hat{\sigma}\} = \arg\min_{\boldsymbol{\theta}, \sigma} \{-\log p(\mathcal{D} \mid \boldsymbol{\theta}, \sigma) - \log p(\boldsymbol{\theta} \mid \gamma_\theta) - p(\sigma \mid \gamma_\sigma)\}. \tag{23}$$

$\boldsymbol{\theta} = [\alpha, \beta_1, \ldots, \beta_M]$ denotes hyperparameters of covariance function and implements automatic relevance determination (ARD) [28], while σ represents the noise variance. For simplicity, the hyperpriors for $\boldsymbol{\theta}$ and σ with hyperparameters $\gamma_\theta, \gamma_\sigma$ are usually set to noninformative distributions. Simple gradient-based optimization methods can be used to solve (23), which requires the computation of the marginal likelihood and its partial derivatives w.r.t. the hyperparameters, i.e.,

$$\frac{\partial}{\partial \theta_m} \log p(\mathbf{y} \mid \mathcal{X}, \boldsymbol{\theta}, \sigma) = \frac{1}{2}\mathbf{y}^T\mathbf{K}_y^{-1}\frac{\partial \mathbf{K}_y}{\partial \theta_m}\mathbf{K}_y^{-1}\mathbf{y} - \frac{1}{2}\mathrm{tr}\left(\mathbf{K}_y^{-1}\frac{\partial \mathbf{K}_y}{\partial \theta_m}\right), \tag{24}$$

where $\mathbf{K}_y = \mathbf{K} + \sigma^2\mathbf{I}$ [28].

PROBABILISTIC INTERPRETATION OF KTPLS AND KTCCA

Many classical methods have been interpreted under a probabilistic model such that overfitting and model selection can be addressed, while also enabling the use of the powerful Bayesian framework. For instance, the probabilistic interpretations of PCA and CCA have been presented in [29] and [30], while KPCA and KCCA are presented as a GP latent variable model (GPLVM) [31]–[33], and nonlinear Tucker decomposition of tensors as a nonparametric Bayesian model using GPs [6].

We introduce a probabilistic interpretation of KTPLS and KTCCA through the generative model shown in Figure 3. Let $\mathcal{X} = \{\mathcal{X}^{(n)}\}_{n=1}^N \in \mathbb{R}^{N \times I_1 \times \cdots \times I_M}$ and $\mathcal{Y} = \{\mathcal{Y}^{(n)}\}_{n=1}^N \in \mathbb{R}^{N \times J_1 \times \cdots \times J_L}$ with their mode-1 matricizations denoted by $\mathbf{X}_{(1)} \in \mathbb{R}^{N \times I}$ and $\mathbf{Y}_{(1)} \in \mathbb{R}^{N \times J}$, we assume both $\mathcal{X}$ and $\mathcal{Y}$ are generated from shared latent variables $\{\mathbf{t}_n\}_{n=1}^N \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ via nonlinear functions given by

$$\mathcal{X}^{(n)} = \mathcal{F}_X(\mathbf{t}_n) + \mathcal{E}_X, \quad \mathcal{Y}^{(n)} = \mathcal{F}_Y(\mathbf{t}_n) + \mathcal{E}_Y, \tag{25}$$

where each element in $\mathcal{E}_X$ and $\mathcal{E}_Y$ is from i.i.d. noise $\mathcal{N}(0, \Lambda_X^{-1})$ and $\mathcal{N}(0, \Lambda_Y^{-1})$, respectively. Note that $\mathcal{F}_X, \mathcal{F}_Y$ are tensor-valued nonlinear functions $\mathcal{F}_X : \mathbf{t} \mapsto \mathcal{X}$, $\mathcal{F}_Y : \mathbf{t} \mapsto \mathcal{Y}$ and $(\mathcal{X}^{(n)})_{i_1 \ldots i_M}$ is generated from $f^X_{i_1 \ldots i_M}(\mathbf{t}_n)$. By specifying GP priors for latent functions $f^X_{i_1 \ldots i_M}$ with a covariance function $k(\mathbf{t}, \mathbf{t}') = \alpha_X \exp(-(\beta_X/2)\|\mathbf{t} - \mathbf{t}'\|^2)$ and for $f^Y_{j_1 \ldots j_L}$ with $k(\mathbf{t}, \mathbf{t}') = \alpha_Y \exp(-(\beta_Y/2)\|\mathbf{t} - \mathbf{t}'\|^2)$, we obtain the marginalized likelihood by integrating latent functions out [31], [33], i.e.,

$$\begin{aligned} p(\mathcal{X} \mid \mathbf{T}, \boldsymbol{\theta}_X) &= \prod_{i=1}^{I} \mathcal{N}(\mathbf{x}_i \mid \mathbf{0}, \mathbf{K}_X), \\ p(\mathcal{Y} \mid \mathbf{T}, \theta_Y) &= \prod_{j=1}^{J} \mathcal{N}(\mathbf{y}_j \mid \mathbf{0}, \mathbf{K}_Y), \end{aligned} \tag{26}$$

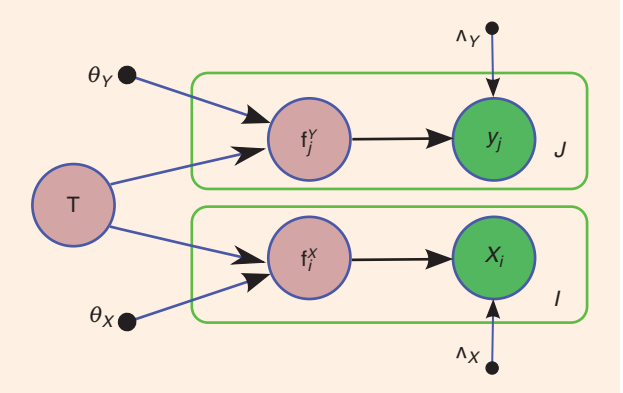
where $\mathbf{x}_i$ denotes the ith column of $\mathbf{X}_{(1)}$, $\mathbf{y}_j$ denotes the jth column of $\mathbf{Y}_{(1)}$, and $I = \prod_m I_m$, $J = \prod_l J_l$. $\mathbf{K}_X = \mathbf{K}_{\mathcal{F}_X} + \Lambda_X^{-1}\mathbf{I}$ is governed by hyperparameters $\boldsymbol{\theta}_X = \{\alpha_X, \beta_X, \Lambda_X\}$, and $\mathbf{K}_Y = \mathbf{K}_{\mathcal{F}_Y} + \Lambda_Y^{-1}\mathbf{I}$ by $\boldsymbol{\theta}_Y = \{\alpha_Y, \beta_Y, \Lambda_Y\}$. From the graphical model in Figure 3, we can see that observations $\mathcal{X}, \mathcal{Y}$ are conditionally independent given latent variables $\mathbf{T}$, thus resulting in the joint distribution

$$p(\mathcal{X}, \mathcal{Y} \mid \mathbf{T}, \boldsymbol{\theta}_X, \boldsymbol{\theta}_Y) = p(\mathcal{X} \mid \mathbf{T}, \boldsymbol{\theta}_X) p(\mathcal{Y} \mid \mathbf{T}, \boldsymbol{\theta}_Y) p(\mathbf{T}), \tag{27}$$

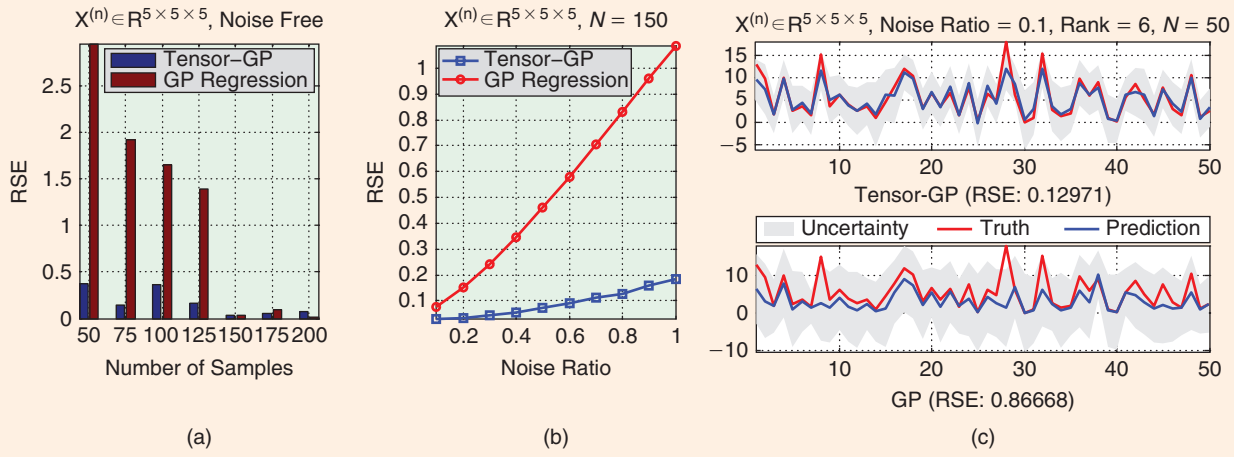
and the corresponding negative log-likelihood [32], [34]

$$\begin{aligned} \mathcal{L} = &\frac{I}{2}\ln|\mathbf{K}_X| + \frac{1}{2}\mathrm{tr}(\mathbf{K}_X^{-1}\mathbf{X}_{(1)}\mathbf{X}_{(1)}^T) \\ &+ \frac{J}{2}\ln|\mathbf{K}_Y| + \frac{1}{2}\mathrm{tr}(\mathbf{K}_Y^{-1}\mathbf{Y}_{(1)}\mathbf{Y}_{(1)}^T) + \mathrm{const.} \end{aligned} \tag{28}$$

To take into account the information on the structure of input data tensors, $\mathbf{X}_{(1)}\mathbf{X}_{(1)}^T$ and $\mathbf{Y}_{(1)}\mathbf{Y}_{(1)}^T$ can be simply replaced by kernel matrices $\tilde{\mathbf{K}}_X$ and $\tilde{\mathbf{K}}_Y$ using the kernel functions $k(\mathcal{X}, \mathcal{X}')$ that are particularly defined for tensors. Based on this model, we can optimize the latent representation $\mathbf{T}$ by maximizing the joint likelihood of the two sets of observations. In general, there is no closed-form solution for latent variables and hyperparameters, and an iterative optimization scheme such as gradient optimization can be used.



[FIG3] Graphical model for probabilistic interpretations of KTPLS/KTCCA. The green nodes represent observed variables and the pink nodes represent the latent variables. The deterministic parameters are shown explicitly by the smaller solid nodes, and the box plates (labeled *I* and *J*) represent independent observations.



[FIG4] The comparison between tensor-GP and standard GP for regression. The predictive performance is illustrated in (a) with varying number of samples and in (b) with varying noise ratio. Part (c) visualizes the predicted and actual output on a specific data set, where the gray area shows uncertainty of predictions.

EXPERIMENTS AND APPLICATIONS

SIMULATIONS ON SYNTHETIC DATA

To illustrate the advantages of tensor-GP for structured data that originally possess multiway relations, $\{\mathcal{X}^{(n)}\}_{n=1}^{N}$ are generated according to the CP model defined in (3) where $\{\mathbf{u}_r^{(m)}, r = 1, \ldots, R, m = 1, \ldots, 3\}$ are drawn from a standard uniform distribution and $\{\lambda_r\}_{r=1}^{R} \sim \mathcal{N}(0,1)$. Here, we set $\mathcal{X}^{(n)} \in \mathbb{R}^{5\times5\times5}$ with $\text{rank}(\mathcal{X}^{(n)}) = R$. Output samples are generated by $y^{(n)} = f(g(\mathcal{X}^{(n)})) + \varepsilon$, where $\varepsilon \sim \mathcal{N}(0, \sigma_\varepsilon^2)$ is additive noise, multilinear transformation $g : \mathcal{X}^{(n)} \mapsto \tilde{\mathbf{x}}^{(n)} \in \mathbb{R}^2$ is defined as $\tilde{x}_i^{(n)} = \mathcal{X}^{(n)} \times_1 \mathbf{w}_i^{(1)T} \cdots \times_3 \mathbf{w}_i^{(3)T}$ with $\{\mathbf{w}_i^{(m)}\}_{m=1}^{3} \sim \mathcal{N}(0,1)$, and a nonlinear transformation is defined as $f(\tilde{\mathbf{x}}^{(n)}) = b_1 + b_2(\tilde{x}_1^{(n)})^2 + b_3(\tilde{x}_2^{(n)})^2$ with coefficients $\mathbf{b} = [0, 1, 3]$. These two successive transformations guarantee that $y^{(n)}$ is neither exactly linear nor mulitilinear related to $\mathcal{X}^{(n)}$. For comparison, tensor-GP with JS divergence covariance function defined in (17) is performed directly on the data set $\{\mathcal{X}, \mathbf{y}\}$, while standard GP with squared exponential covariance function is performed on the unfolded data set $\{\mathbf{X}_{(1)}, \mathbf{y}\}$. The MAP estimation of hyperparameters in (23) is used to implement ARD. Relative squared error, $\text{RSE} = \left[\sum_{n=1}^{N}(\hat{y}_n - y_n)^2\right] / \left[\sum_{n=1}^{N}(\bar{y} - y_n)^2\right]$, is used for performance evaluation.

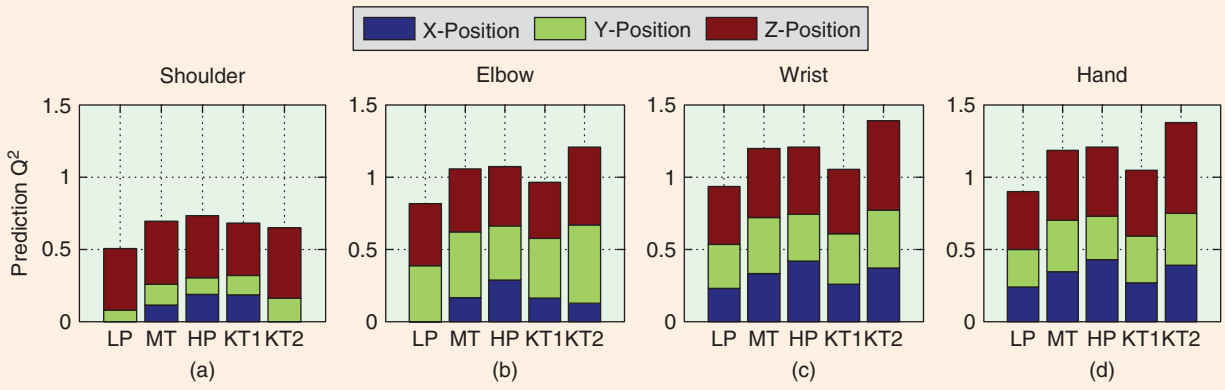
The results comparing the performance of tensor-GP and standard GP are given in Figure 4. In the first simulation, we investigate the performance for an increasing number of samples N (N samples for training and another N samples for testing) with other conditions fixed, i.e., noise ratio $(\sigma_\varepsilon/\sigma_f) = 0$, where σ_f denotes the standard variance of generated $\{f(g(\mathcal{X}^{(n)}))\}_{n=1}^{N}$ and $\text{rank}(\mathcal{X}^{(n)}) = 10$. As shown in Figure 4(a), tensor-GP significantly outperforms standard GP when the number of samples is relatively small compared to the number of variables in $\mathcal{X}^{(n)}$—in this case, 125. In the second simulation, we investigate the performance for varying noise ratio with other conditions fixed for $N = 150$, $\text{rank}(\mathcal{X}^{(n)}) = 10$, since both tensor-GP and GP obtain a similar performance under such conditions in the first simulation. As shown in Figure 4(b), tensor-GP significantly outperforms GP when noise ratio is high. Figure 4(c) illustrates the predictions on a specific data set under conditions setting $N = 50, (\sigma_\varepsilon/\sigma_f) = 0.1, \text{rank}(\mathcal{X}^{(n)}) = 6$. Observe that tensor-GP enhances predictive performance compared with GP, and the confidence region is much smaller by tensor-GP. Note that when the rank of data is smaller, performance is better for both two methods. The MAP estimation of hyperparameters is shown in Table 1, in which the number of hyperparameters for GP are too large to show. In summary, these simulations demonstrate the advantages of tensor kernel with respect to robustness to noise and small number of samples, when data itself has multilinear structure.

[TABLE 1] MAP ESTIMATION OF HYPERPARAMETERS.

MODELS	θ_{MAP}	σ^2_{MAP}
TENSOR-GP	(21.37, 2.29, 1.98, 1.61)	0.0094
GP REGRESSION	LENGTH = 126	0.0495

ECoG DECODING USING KTPLS

To illustrate the effectiveness of tensor kernel, KTPLS regression was applied for decoding of 3-D movement trajectories from ECoG signals recorded from a monkey brain. The data sets and detailed descriptions are freely available from http://neurotycho.org. The movements of a monkey were captured by an optical motion capture system with reflective markers affixed to the left shoulder, elbows, wrists, and hand, thus the dependent data can be represented as a third-order tensor $\mathcal{Y}$ (i.e., samples × 3-D positions × markers). The ECoG signals are transformed to time-frequency domains to represent the discriminative features. As a

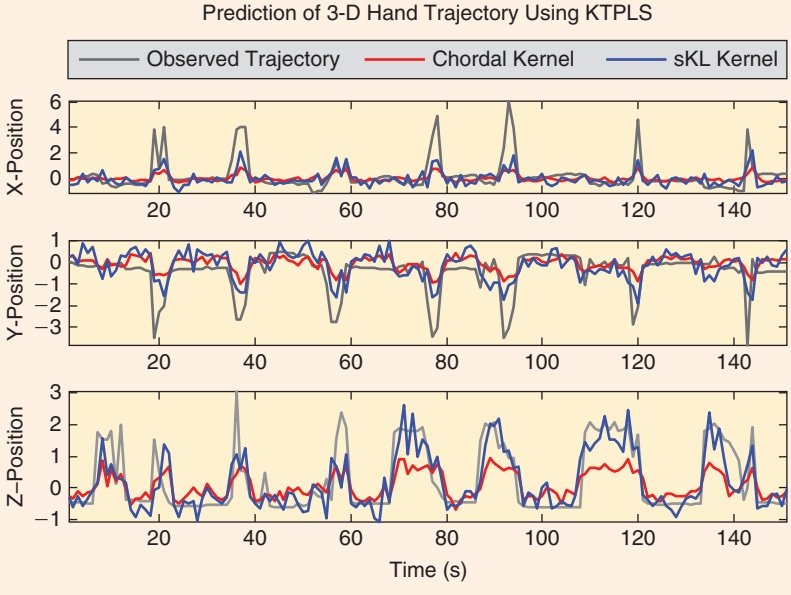


[FIG5] The prediction performance for movement trajectories recorded from a monkey's shoulder, elbow, wrist, and hand using five regression methods: linear PLS (LP), multitask learning (MT), HOPLS (HP), KTPLS with Chordal distance based kernel (KT1), and KTPLS with sKL divergence-based kernel (KT2). The values of evaluation index $Q^2 = 1 - \|\hat{\mathbf{y}} - \mathbf{y}\|^2 / \|\mathbf{y}\|^2$ for (X, Y, Z)-positions are shown in different colors and are accumulated individually for each limb marker, which demonstrates that KT2 achieves the best performance. The fact that X-coordinate is missing for LP indicates that $Q^2 = 0$.

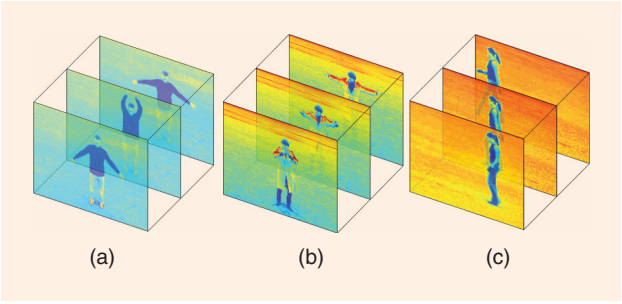
result, the independent data is represented as a fourth-order tensor $\mathcal{X}$ (i.e., epoch × channel × frequency × time), where each ECoG epoch $\mathcal{X}^{(n)}$, containing the most recent past one second time window, corresponds to one sample of movement data $\mathcal{Y}^{(n)}$. Two different tensor kernels using Chordal distance as defined in (13) and using sKL divergence as defined in (17) are employed by KTPLS individually for performance evaluation. The data set is divided into training set (ten minutes) and test set (five minutes), and the optimal tuning parameters, such as kernel parameters $\{\beta_m\}_{m=1}^{3}$ and number of latent vectors, are selected by cross-validation on the training set. The performances on the test set are shown in Figure 5, illustrating the superiority of KTPLS over PLS, HOPLS, and multitask learning with $l_{2,1}$-norm regularization [35]. The best result is achieved by KTPLS using sKL divergence-based tensor kernel. Figure 6 further visualizes the reconstructed trajectories of a monkey hand using two tensor kernels for comparison. This example demonstrates that the proposed generative probabilistic kernel can bring together the advantages of both kernels and multilinear tensor representation. Since a generative model is employed by sKL kernel, KTPLS can be considered to be both a generative and a discriminative method.

VIDEO CLASSIFICATION USING KTCCA

Human action recognition in videos is of high interest for a variety of applications such as video surveillance, human-computer interaction, and video retrieval, where the most competing methods are based on motion estimation [36], local space-time interest points and visual code words [37]–[39], multiple classifiers [40], [41], sparse representation [42], and multiway tensor methods [27], [43]. Tensor representation enables us to directly analyze 3-D video volume and encode global space-time structure information. To illustrate the advantages of tensor kernels, the KTCCA algorithm, described in the section "Kernel-Based Tensor CCA," is used for extracting the global space-time geometrical features and is evaluated on the largest public KTH human action database [38] that contains six types of actions [walking (W), running (R), jogging (J), boxing (B), hand-waving (H-W), and hand-clapping (H-C)] performed by 25 people in four different scenarios



[FIG6] Visualization of reconstructed and actual trajectories of hand movements. Observe that KTPLS using sKL divergence-based kernel obtains better prediction than Chordal distance-based kernel.



[FIG7] Three examples of video sequences for (a) hand waving, (b) hand clapping, and (c) walking actions, which can be represented in a tensor form.

(outdoors, outdoors with scale variation, outdoors with different clothes, and indoors). The total 600 video sequences are divided with respect to the people into a training set (eight people), a validation set (eight people), and a test set (nine people) according to the standard setting in [38]. Each video is space-time aligned and uniformly resized to $20 \times 20 \times 32$, which are then represented by a third-order tensor $\mathcal{X}^{(n)}$ (see Figure 7).

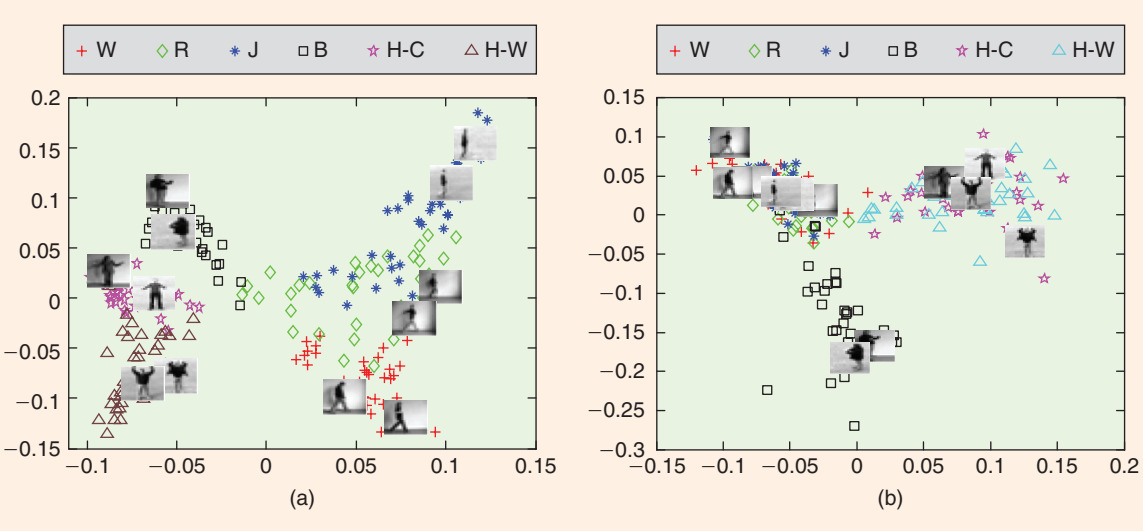
We apply KTCCA to find a low-dimensional latent space between video sequences denoted by $\mathcal{X}$ and the corresponding class membership denoted by $\mathbf{y}$, which can be considered as a supervised feature extraction approach. KTCCA using the Chordal distance kernel defined in (13) is compared with KCCA using an RBF kernel performed on vectorization of tensors $\{\text{vec}(\mathcal{X}^{(n)})\}_{n=1}^{N}$. For comparison, isotropic kernels [i.e., $\beta_1 = \cdots = \beta_M$ in (13) and RBF kernel] are used. The optimal kernel parameter is found according to performance on the validation set and then is used to retrain the models from both training and validation set. The test data is projected onto the latent space by learned models to obtain the discriminative features as shown in Figure 8.

Observe that KTCCA outperforms KCCA with respect to the discriminative ability and six classes are well separated even in two dimensional space. A simple k-nearest neighbor classifier (k-NN) is applied on lower-dimensional features for action classification and the confusion matrices on test set are compared in Table 2, in which rows correspond to the ground truth, and columns correspond to the classification results. It can be seen that KTCCA achieves average accuracy of 98% while KCCA achieves 83%, and the confusion of KTCCA only appears between running and jogging, and between hand-clapping and hand-waving, which is consistent with our intuition that these two pairs of actions are easily confused. The superiority of KTCCA over KCCA indicates that space-time structures of video volumes captured by tensor kernel significantly improves the discriminative performance.

In addition, leave-one-out performance is evaluated for comparison with the state-of-the-art methods on the KTH data set. As shown in Table 3, KTCCA achieves the highest overall classification accuracy followed by product manifold (PM) [43], tensor CCA (TCCA) [27], and boosted exemplar learning (BEL) [41]. In summary, both global and local space-time information are discriminative and promising for action recognition and the results demonstrate the effectiveness and advantages of the tensor kernel when employed by KTCCA for global space-time feature extraction from videos. The MATLAB code for the implementation can be found at http://www.bsp.brain.riken.jp/~qibin/homepage/KernelTensor.html.

SUMMARY

In this article, we have presented a framework that brings powerful kernel methods and tensor decomposition techniques together such that nonlinear kernel-based strategy can be applied to tensor decompositions and tensor subspace



[FIG8] Visualization of test data in the first two-dimensional latent space. (a) represents KTCCA using a tensor kernel while (b) represents KCCA using an RBF kernel performed on the vectorization of tensors. Observe that the features obtained by KTCCA are more discriminative than KCCA features.

[TABLE 2] ACTION RECOGNITION BY KTCCA WITH A TENSOR KERNEL AND KCCA WITH AN RBF KERNEL.

CONFUSION MATRIX BY KTCCA (AVERAGE 98%)

	WALK	RUN	JOG	BOX	H-C	H-W
WALK	1.0	0	0	0	0	0
RUN	0	.97	.03	0	0	0
JOG	0	0	1.0	0	0	0
BOX	0	0	0	1.0	0	0
H-C	0	0	0	0	1.0	0
H-W	0	0	0	0	.08	.92

CONFUSION MATRIX BY KCCA (AVERAGE 83%)

	WALK	RUN	JOG	BOX	H-C	H-W
WALK	.91	.03	.06	0	0	0
RUN	.03	.86	.11	0	0	0
JOG	0	.17	.83	0	0	0
BOX	0	.08	.03	.78	.08	.03
H-C	0	.03	0	0	.83	.14
H-W	0	.08	0	0	.14	.78

[TABLE 3] COMPARISONS OF LEAVE-ONE-OUT CLASSIFICATION ACCURACY ON THE KTH DATA SET.

KTCCA	TCCA [27]	PM [43]	PLSA/ISM [44]	WX/SVM [44]	BEL [41]	MIL [36]	PLSA/LDA [37]	LF/SVM [38]
97.83%	95.33%	97%	83.92%	91.6%	95.33%	87.7%	83.33%	71.72%

models, allowing one to account for both the multilinear structure and nonlinear dependencies within the data. An overview of the state-of-the-art methods relevant to this topic is provided together with specific examples. The definition of effective kernel functions for tensors creates a range of possibilities for the development of machine-learning methods that take the underlying multiway structure into account. In particular, we studied tensor decompositions from a kernel perspective, and kernel-based methods from a tensor representation viewpoint. Several novel models are proposed together with probabilistic interpretations, and their advantages are demonstrated by simulations on synthetic data and two real-world applications.

ACKNOWLEDGMENTS

This work was supported by JSPS KAKENHI (grant 24700154), U.S. National Science Foundation grants CCF-1117056 and IIS-1017718, and the National Natural Science Foundation of China (grants 61202155, 90920014, 91120305, and 61103122).

AUTHORS

Qibin Zhao (qbzhao@brain.riken.jp) is a research scientist in the Laboratory for Advanced Brain Signal Processing at RIKEN Brain Science Institute, Japan, and an assistant researcher in the Department of Computer Science and Engineering, Shanghai Jiao Tong University, China. He received his Ph.D. degree in computer science from Shanghai Jiao Tong University in 2009. His research interests include machine learning, tensor analysis, Bayesian inference, and brain computer interface. He has published more than 40 papers in journals and international conferences.

Guoxu Zhou (zhouguoxu@brain.riken.jp) is a research scientist in the Laboratory for Advanced Brain Signal Processing at RIKEN Brain Science Institute, Japan. He received his Ph.D. degree in intelligent signal and information processing from the School of Electronic and Information Engineering, South China University of Technology, Guangzhou, China, in 2010. His research interests include statistical signal processing, blind source separation, tensor analysis, intelligent information processing, and machine learning. He has published more than 30 papers in journals and international conferences.

Tülay Adali (adali@umbc.edu) received the Ph.D. degree in electrical engineering from North Carolina State University, Raleigh, in 1992 and joined the faculty at the University of Maryland, Baltimore, the same year, where she currently is a Professor in the Department of Computer Science and Electrical Engineering. She assisted in the organization of a number of international conferences and workshops including the International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Machine Learning for Signal Processing (MLSP), and Neural Networks for Signal Processing. She chaired the MLSP Technical Committee (2003–2005, 2011–2013) and served on numerous committees and boards of the IEEE Signal Processing Society (SPS) as well as journals. She is currently serving on the editorial boards of *Proceedings of the IEEE* and *Journal of Signal Processing Systems* and is a member of the MLSP and Signal Processing Theory and Methods Technical Committees. She is a Fellow of the IEEE and the American Institute for Medical and Biological Engineering, recipient of a 2010 IEEE SPS Best Paper Award, 2012–2013 University System of Maryland Regents' Award for Research, and an NSF CAREER Award. She is an IEEE SPS Distinguished Lecturer for 2012 and 2013. Her research interests are in the areas of statistical signal processing, machine learning for signal processing, and biomedical data analysis.

Liqing Zhang (zhang-lq@cs.sjtu.edu.cn) is a professor with the Department of Computer Science and Engineering, Shanghai Jiao Tong University, China. He received the Ph.D. degree from Zhongshan University, Guangzhou, China, in 1988. He was promoted to full professor in 1995 at South China University of Technology. He worked as a research scientist at RIKEN Brain Science Institute, Japan, from 1997 to 2002. His current research interests cover computational theory for cortical networks, brain–computer interface, visual perception and image search, statistical learning, and inference. He has published more than 200 papers in journals and international conferences.

Andrzej Cichocki (a.cichocki@riken.jp) is a senior team leader and head of the Laboratory for Advanced Brain Signal

Processing at RIKEN Brain Science Institute, Japan, and a professor at the Systems Research Institute of the Polish Academy of Science, Warsaw. He received the M.Sc. (honors), Ph.D., and Dr.Sc. (Habilitation) degrees, all in electrical engineering from Warsaw University of Technology. He spent several years at the University Erlangen (Germany) as an Alexander-von-Humboldt Research Fellow and guest professor. He is (co)author of more than 300 technical journal papers and four monographs in English (two of them translated to Chinese). He is an associate editor of *IEEE Transactions on Signal Processing* and *Journal of Neurosciemce Methods*. He was the founding editor-in-chief for *Journal Computational Intelligence and Neuroscience*. Currently, his research focus is on tensor decompositions, EEG hyperscanning, and their practical applications. His publications currently report over 18,000 citations, according to Google Scholar, with an h-index of 58.

REFERENCES

[1] A. Cichocki, R. Zdunek, A. H. Phan, and S. I. Amari, *Nonnegative Matrix and Tensor Factorizations*. Hoboken, NJ: Wiley, 2009.

[2] T. Kolda and B. Bader, "Tensor decompositions and applications," *SIAM Rev.*, vol. 51, no. 3, pp. 455–500, 2009.

[3] Y. Li, Y. Du, and X. Lin, "Kernel-based multifactor analysis for image synthesis and recognition," in *Proc. 10th IEEE Int. Conf. Computer Vision*, 2005, vol. 1, pp. 114–119.

[4] J. Shawe-Taylor and N. Cristianini, *Kernel Methods for Pattern Analysis*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[5] M. Signoretto, L. De Lathauwer, and J. A. Suykens, "A kernel-based framework to tensorial data analysis," *Neural Netw.*, vol. 24, no. 8, pp. 861–874, 2011.

[6] Z. Xu, F. Yan, and A. Qi, "Infinite Tucker decomposition: Nonparametric Bayesian models for multiway data analysis," in *Proc. 29th Int. Conf. Machine Learning (ICML-12)*, pp. 1023–1030.

[7] D. Hardoon and J. Shawe-Taylor, "Decomposing the tensor kernel support vector machine for neuroscience data with structured labels," *Mach. Learn.*, vol. 79, no. 1, pp. 29–46, 2010.

[8] M. Signoretto, E. Olivetti, L. De Lathauwer, and J. Suykens, "Classification of multichannel signals with cumulant-based kernels," *IEEE Trans. Signal Processing*, vol. 60, no. 5, pp. 2304–2314, 2012.

[9] A. Aizerman, E. Braverman, and L. Rozoner, "Theoretical foundations of the potential function method in pattern recognition learning," *Autom. Remote Control*, vol. 25, pp. 821–837, June 1964.

[10] B. Schölkopf, A. Smola, and K. Müller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural Comput.*, vol. 10, no. 5, pp. 1299–1319, 1998.

[11] K. Müller, S. Mika, G. Rätsch, K. Tsuda, and B. Schölkopf, "An introduction to kernel-based learning algorithms," *IEEE Trans. Neural Networks*, vol. 12, no. 2, pp. 181–201, 2001.

[12] I. Jolliffe, *Principal Component Analysis*. Hoboken, NJ: Wiley, 2005.

[13] L. De Lathauwer, B. De Moor, and J. Vandewalle, "A multilinear singular value decomposition," *SIAM J. Matrix Anal. Appl.*, vol. 21, no. 4, pp. 1253–1278, 2000.

[14] M. Vasilescu and D. Terzopoulos, "Multilinear analysis of image ensembles: Tensorfaces," in *Proc. European Conf. Computer Vision*, 2002, pp. 447–460.

[15] S. Park and M. Savvides, "Individual kernel tensor-subspaces for robust face recognition: A computationally efficient tensor framework without requiring mode factorization," *IEEE Trans. Syst. Man Cybern. B*, vol. 37, no. 5, pp. 1156–1166, 2007.

[16] P. Moreno, P. Ho, and N. Vasconcelos, "A Kullback–Leibler divergence based kernel for SVM classification in multimedia applications," in *Advances in Neural Information Processing Systems*, vol. 16. Cambridge, MA: MIT Press, 2003, pp. 1385–1393.

[17] A. Chan, N. Vasconcelos, and P. Moreno, "A family of probabilistic kernels based on information divergence," Univ. California, San Diego, Tech. Rep. SVCL-TR-2004-1, 2004.

[18] D. Endres and J. Schindelin, "A new metric for probability distributions," *IEEE Trans. Inform. Theory*, vol. 49, no. 7, pp. 1858–1860, 2003.

[19] K. Abou-Moustafa and F. Ferrie, "A note on metric properties for some divergence measures: The Gaussian case," *J. Mach. Learn. Res.*, vol. 25, pp. 1–15, Nov. 2012.

[20] D. Schnitzer. (2011). MATLAB/Octave Multivariate Normals Toolbox [Online]. Available: http://www.ofai.at/~dominik.schnitzer/mvn/

[21] R. Rosipal and L. Trejo, "Kernel partial least squares regression in reproducing kernel Hilbert space," *J. Mach. Learn. Res.*, vol. 2, no. 2, pp. 97–123, Mar. 2002.

[22] G. Guo and G. Mu, "Simultaneous dimensionality reduction and human age estimation via kernel partial least squares regression," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2011, pp. 657–664.

[23] Q. Zhao, C. F. Caiafa, D. P. Mandic, Z. C. Chao, Y. Nagasaka, N. Fujii, L. Zhang, and A. Cichocki, "Higher-order partial least squares (HOPLS): A generalized multilinear regression method," *IEEE Trans. Pattern Anal. Mach. Intell*, to be published.

[24] L. Sun, S. Ji, and J. Ye, "Canonical correlation analysis for multilabel classification: A least-squares formulation, extensions, and analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 1, pp. 194–200, 2011.

[25] K. Fukumizu, F. Bach, and A. Gretton, "Statistical convergence of kernel CCA," in *Advances in Neural Information Processing Systems (NIPS)*. Cambridge, MA: MIT Press, 2006, pp. 387–394.

[26] D. Hardoon, S. Szedmak, and J. Shawe-Taylor, "Canonical correlation analysis: An overview with application to learning methods," *Neural Comput.*, vol. 16, no. 12, pp. 2639–2664, 2004.

[27] T. Kim and R. Cipolla, "Canonical correlation analysis of video volume tensors for action categorization and detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 8, pp. 1415–1428, 2009.

[28] C. Rasmussen and C. Williams, *Gaussian Processes for Machine Learning*, vol. 1. Cambridge, MA: MIT Press, 2006.

[29] M. Tipping and C. Bishop, "Probabilistic principal component analysis," *J. Roy. Stat. Soc. B (Stat. Methodol.)*, vol. 61, no. 3, pp. 611–622, 1999.

[30] F. Bach and M. Jordan, "A probabilistic interpretation of canonical correlation analysis," Dept. Stat., Univ. California, Berkeley, Tech. Rep. 688, 2005.

[31] N. Lawrence, "Probabilistic non-linear principal component analysis with Gaussian process latent variable models," *J. Mach. Learn. Res.*, vol. 6, pp. 1783–1816, Nov. 2005.

[32] A. Shon, K. Grochow, A. Hertzmann, and R. Rao, "Learning shared latent structure for image synthesis and robotic imitation," in *Advances in Neural Information Processing Systems*, vol. 18. Cambridge, MA: MIT Press, 2006, pp. 1233–1240.

[33] C. Fyfe, G. Leen, and P. Lai, "Gaussian processes for canonical correlation analysis," *Neurocomputing*, vol. 71, no. 16, pp. 3077–3088, 2008.

[34] M. Salzmann, C. Ek, R. Urtasun, and T. Darrell, "Factorized orthogonal latent spaces," in *Proc. Int. Conf. Artificial Intelligence and Statistics*, vol. 9, May 2010, pp. 701-708.

[35] J. Zhou, J. Chen, and J. Ye. (2011). MALSAR: Multi-task learning via structural regularization [Online]. Available: http://www.public.asu.edu/~jye02/Software/MALSAR

[36] S. Ali and M. Shah, "Human action recognition in videos using kinematic features and multiple instance learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 2, pp. 288–303, 2010.

[37] J. Niebles, H. Wang, and L. Fei-Fei, "Unsupervised learning of human action categories using spatial–temporal words," *Int. J. Comput. Vis.*, vol. 79, no. 3, pp. 299–318, 2008.

[38] C. Schuldt, I. Laptev, and B. Caputo, "Recognizing human actions: A local SVM approach," in *Proc. 17th Int. Conf. Pattern Recognition, ICPR*, 2004, vol. 3, pp. 32–36.

[39] M. Holte, B. Chakraborty, J. Gonzalez, and T. Moeslund, "A local 3-D motion descriptor for multi-view human action recognition from 4-D spatio-temporal interest points," *IEEE J. Select. Topics Signal Processing*, vol. 6, no. 5, pp. 553–565, 2012.

[40] Y. Song, Y. Zheng, S. Tang, X. Zhou, Y. Zhang, S. Lin, and T. Chua, "Localized multiple kernel learning for realistic human action recognition in videos," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 21, no. 9, pp. 1193–1202, 2011.

[41] T. Zhang, J. Liu, S. Liu, C. Xu, and H. Lu, "Boosted exemplar learning for action recognition and annotation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 21, no. 7, pp. 853–866, 2011.

[42] T. Guha and R. Ward, "Learning sparse representations for human action recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 8, pp. 1576–1588, 2012.

[43] Y. Lui, J. Beveridge, and M. Kirby, "Action classification on product manifolds," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2010, pp. 833–839.

[44] S. Wong, T. Kim, and R. Cipolla, "Learning motion categories using both semantic and structural information," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR'07)*, pp. 1–6.

[45] R. Rosipal, M. Girolami, L. J. Trejo, and A. Cichocki, "Kernel PCA for feature extraction and de-noising in nonlinear regression," *Neural Comput. Applicat.*, vol. 10, no. 3, pp. 231–243, 2001.

[SP]