Hindawi
Mathematical Problems in Engineering
Volume 2017, Article ID 3818949, 11 pages
https://doi.org/10.1155/2017/3818949

Research Article

Least Square Support Tensor Regression Machine Based on Submatrix of the Tensor

Tuo Shu and Zhi-Xia Yang

College of Mathematics and Systems Science, Xinjiang University, Urumqi 830046, China

Correspondence should be addressed to Zhi-Xia Yang; xjyangzhx@sina.com

Received 14 March 2017; Revised 10 October 2017; Accepted 15 October 2017; Published 9 November 2017

Academic Editor: Gisella Tomasini

For tensor regression problem, a novel method, called least square support tensor regression machine based on submatrix of a tensor (LS-STRM-SMT), is proposed. LS-STRM-SMT is a method which can be applied to deal with tensor regression problem more efficiently. First, we develop least square support matrix regression machine (LS-SMRM) and propose a fixed point algorithm to solve it. And then LS-STRM-SMT for tensor data is proposed. Inspired by the relation between photochrome and the gray pictures, we reformulate the tensor sample training set and form the new model (LS-STRM-SMT) for tensor regression problem. With the introduction of projection matrices and another fixed point algorithm, we turn the LS-STRM-SMT model into several related LS-SMRM models which are solved by the algorithm for LS-SMRM. Since the fixed point algorithm is used twice while solving the LS-STRM-SMT problem, we call the algorithm dual fixed point algorithm (DFPA). Our method (LS-STRM-SMT) has been compared with several typical support tensor regression machines (STRMs). From theoretical point of view, our algorithm has less parameters and its computational complexity should be lower, especially when the rank of submatrix K is small. The numerical experiments indicate that our algorithm has a better performance.

1. Introduction

As we all know, in the past decades, matrices or more generally multiway arrays (tensors) types of data have an increasing number of applications. For example, all raster images are essentially digital readings of a grid of sensors and matrix analysis is widely applied in image processing, for example, photo realistic images of faces [1], palms [2], and medical images [3]. In web search, a large amount of tensors that stand for images [4] can be found easily. Therefore, tensor data analysis [5], particularly regression problem [6, 7], has become one of the most important topics for face recognition [8], palmprint recognition [9], and so on.

Tensor types of data have greatly drawn the attention of people. Recently, several tensor learning for regression approaches [10, 11] appears, but the majority of them dealing with tensor regression problems work on vector spaces that are derived by stacking the original tensor elements in a more or less arbitrary order. This vectorization of data causes many new problems. First, the structural information is destroyed. Second, the vectorization of a tensor may bring an extremely high dimensionality vector which may lead to high computational complexity, overfitting, and large memory requirement. The rest of the methods mainly take advantage of the decomposition of a matrix [12] or tensor [6], which can reduce the high computational complexity as well as high dimensionality at the expense of slight decline of accuracy, but the structural information is destroyed totally. So a more reasonable method which can reserve the underlying structural information and avoid overfitting, high dimensionality, and high computational complexity is needed.

Considering the fact that a colorful photograph can be expressed as a third-order tensor, of which each frontal slice is indeed a gray image that contains almost all the information of the colorful photograph, we can take advantage of this by introducing submatrix of a tensor and abstract vector space when solving tensor learning for regression problems. That means, each tensor data sample can be regarded as an abstract vector [13] whose elements are submatrix types of features. Gathering together the same feature information of different

tensor data sample, we can construct new submatrix training sets and the same number of related training models, from which we can get an equal amount of weight submatrix. And then, the weight tensor is obtained. Besides, we improve the fixed point algorithm [14] via some projection matrices [15], including a series of left projection matrices and a right projection matrix. The improved algorithm is called dual fixed point algorithm (DFPA). The projection matrices can not only join the training models up but also reduce computational complexity and large memory requirement. That is to say, we turn the LS-STRM-SMT problems into solving a battery of least square support matrix machine (LS-SMRM) by fixing the projection matrices and then solve the LS-SMRMs problems with fixed point algorithm. The numerical experiments indicate that our method and algorithm have a better performance.

The paper is organized as follows: in Section 2, notations and preliminaries are introduced, such as definitions related to tensors and notation that will be used. In Section 3, we propose our (LS-SMRM) for matrix regression problems and the fixed point algorithm for them. In Section 4, we propose the LS-STRM-SMT models and develop the DFPA to solve them. Computational comparisons on both UCI data sets and artificial data are done in Section 5 and conclusions in Section 6.

2. Notations and Preliminaries

Here, we will give a brief description of the notations that will be used in the later sections. More specifically, boldface capital letters, for example, $\mathbf{A}$, boldface lowercase letters, for example, $\mathbf{a}$, and lowercase letters, for example, a, are used to denoted matrices, vectors, and scalars, respectively. Tensors regarded as multidimensional arrays will be denoted by Euler script calligraphic letters, for example, $\mathcal{X} \in R^{I_1 \times I_2 \cdots \times I_M}$, where M denotes the order of the tensor. Inspired by the fact that ith element of a vector $\mathbf{x} \in R^n$ is denoted by x_i, $i = 1, 2, \ldots, n$, the elements of an M-order tensor $\mathcal{X}$ will be denoted by $x_{i_1 i_2 \cdots i_M}$, $i_l = 1, 2, \ldots, I_l$, $l = 1, 2, \ldots, M$.

For an M-order tensor $\mathcal{X} \in R^{I_1 \times I_2 \cdots \times I_M}$, the d-mode matricization also known as unfolding or flattening is denoted by

$$\mathbf{X}_{(d)} = \mathrm{mat}_d(\mathcal{X}) \in R^{I_d \times (I_1 I_2 \cdots I_{d-1} I_{d+1} \cdots I_M)}, \quad d = 1, 2 \ldots, M. \tag{1}$$

It is quite clear that we can reorder the elements of the tensor into a matrix in such a way. On the contrary, define a mapping function

$$\cdot^{\langle d \rangle} : R^{I_d \times (I_1 I_2 \cdots I_{d-1} I_{d+1} \cdots I_M)} \longrightarrow R^{I_1 \times I_2 \cdots \times I_M}, \tag{2}$$

to recover a tensor from its unfolding matrix. Particularly, when $M = 3$, we have

$$\begin{aligned} \cdot^{\langle 1 \rangle}(\mathbf{X}_{(1)}) &= \mathcal{X}, \quad \mathbf{X}_{(1)} \in R^{I_1 \times (I_2 I_3)}, \\ \cdot^{\langle 2 \rangle}(\mathbf{X}_{(2)}) &= \mathcal{X}, \quad \mathbf{X}_{(2)} \in R^{I_2 \times (I_1 I_3)}, \\ \cdot^{\langle 3 \rangle}(\mathbf{X}_{(3)}) &= \mathcal{X}, \quad \mathbf{X}_{(3)} \in R^{I_3 \times (I_1 I_2)}. \end{aligned} \tag{3}$$

The inner product of the two same size tensors $\mathcal{X}, \mathcal{Y} \in R^{I_1 \times I_2 \cdots \times I_M}$ is defined as

$$\langle \mathcal{X}, \mathcal{Y} \rangle = \sum_{i_1=1}^{I_1} \cdots \sum_{i_M=1}^{I_M} x_{i_1 i_2 \cdots i_n} y_{i_1 i_2 \cdots i_n}. \tag{4}$$

The Frobenius norm of a tensor is thus defined as

$$\|\mathcal{X}\|_{\mathrm{Fro}} = \sqrt{\langle \mathcal{X}, \mathcal{X} \rangle}. \tag{5}$$

And it can be shown that

$$\|\mathcal{X}\|_{\mathrm{Fro}} = \left\|\mathbf{X}_{(d)}\right\|_{\mathrm{Fro}} = \sqrt{\mathrm{tr}\left(\mathbf{X}_{(d)} \mathbf{X}_{(d)}^T\right)}, \quad d = 1, 2, \ldots, M. \tag{6}$$

The Contrast Pyramid, referred to as CP, decomposition factorizes an M order tensor $\mathcal{X} \in R^{I_1 \times I_2 \cdots \times I_M}$ into a linear combination of R rank-one tensors, written as

$$\mathcal{X} \approx \sum_{r=1}^{R} \mathbf{u}_r^1 \circ \mathbf{u}_r^2 \cdots \circ \mathbf{u}_r^M \triangleq \left[\left|\mathbf{U}^1, \mathbf{U}^2, \ldots, \mathbf{U}^M\right|\right], \tag{7}$$

where the operator $\circ$ is the outer product of vectors and the factor matrix $\mathbf{U}^k = [\mathbf{u}_1^k, \mathbf{u}_2^k, \ldots, \mathbf{u}_R^k] \in R^{I_k \times R}$, $k = 1, 2, \ldots, M$, of the size $I_k \times R$, $k = 1, 2, \ldots, M$. For the convenience, the mentioned tensor is the third-order tensor in the following content if there is no special instruction.

3. Least Square Support Matrix Regression Machine (LS-SMRM)

In the section, we propose least square support matrix regression machine, shorten as LS-SMRM, for the regression problem with matrix input.

Give a training set

$$T = \{(\mathbf{X}_1, y_1), (\mathbf{X}_2, y_2), \ldots, (\mathbf{X}_L, y_L)\}, \tag{8}$$

where $\mathbf{X}_i \in R^{I_1 \times I_2}$ is the input and $y_i \in R$ is the output, $i = 1, 2, \ldots, L$. Our task is to find a predictor

$$y = \langle \mathbf{W}, \mathbf{X} \rangle + b, \tag{9}$$

where $\mathbf{W}$ denotes the weight matrix and b is the bias. For a new input matrix, we can predict its output through the above-mentioned predictor.

In order to get predictor (9), we develop the following optimization problem:

$$\begin{aligned} \min_{\mathbf{W}, b, \eta} \quad & \frac{1}{2} \|\mathbf{W}\|_{\mathrm{Fro}}^2 + \frac{C}{2} \sum_{i=1}^{L} \eta_i^2 \\ \text{s.t.} \quad & y_i - (\langle \mathbf{W}, \mathbf{X}_i \rangle + b) = \eta_i, \quad i = 1, 2, \ldots, L, \end{aligned} \tag{10}$$

where $C > 0$ is penalty parameter.

According to the CP decomposition (7), the matrices $\mathbf{V}$ and $\mathbf{U}$ can be found to make

$$\mathbf{W} \approx \sum_{r=1}^{K} \mathbf{v}_r \circ \mathbf{u}_r \triangleq [|\mathbf{V}, \mathbf{U}|], \tag{11}$$

where $\mathbf{V} = (\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_K)$, $\mathbf{U} = (\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_K)$, $\mathbf{v}_i \in R^{I_1}$, $\mathbf{u}_i \in R^{I_2}$, $i = 1, 2, \ldots, K$, and $K = 2, 3, \ldots, \min\{I_1, I_2\}$. Then, optimization problem (10) can be turned into as follows:

$$\begin{aligned} \min_{\mathbf{V},\mathbf{U},\eta,b} \quad & \frac{1}{2}\left\|\mathbf{V}\mathbf{U}^T\right\|_{\text{Fro}}^2 + \frac{C}{2}\sum_{i=1}^{L}\eta_i^2, \\ \text{s.t.} \quad & y_i - \left(\sum_{j=1}^{K}\mathbf{v}_j^T\mathbf{X}_i\mathbf{u}_j + b\right) = \eta_i, \quad i = 1, 2, \ldots, L. \end{aligned} \tag{12}$$

The fixed point algorithm is applied to solve optimization problem (12). When $\mathbf{U} = (\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_K)$ is fixed, we need to compute a set of $\mathbf{v}_j$ for $j = 1, 2, \ldots, K$ and b. Firstly, denote

$$\begin{aligned} \widehat{\mathbf{f}}_i &= \begin{pmatrix} \mathbf{X}_i\mathbf{u}_1 \\ \mathbf{X}_i\mathbf{u}_2 \\ \vdots \\ \mathbf{X}_i\mathbf{u}_K \end{pmatrix}_{I_1K\times 1}, \\ \widehat{\mathbf{v}} &= \begin{pmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \vdots \\ \mathbf{v}_K \end{pmatrix}_{I_1K\times 1} \\ \mathbf{D} &= \left(\mathbf{U}^T\mathbf{U}\right) \otimes I_{I_1\times I_1}, \end{aligned} \tag{13}$$

and the optimization problems (12) are equivalent to

$$\begin{aligned} \min_{\widehat{\mathbf{v}},\eta,b} \quad & \frac{1}{2}\widehat{\mathbf{v}}^T D\widehat{\mathbf{v}} + \frac{C}{2}\sum_{i=1}^{L}\eta_i^2 \\ \text{s.t.} \quad & y_i - \left(\widehat{\mathbf{v}}^T\widehat{\mathbf{f}}_i + b\right) = \eta_i, \quad i = 1, 2, \ldots, L. \end{aligned} \tag{14}$$

And let

$$\begin{aligned} \mathbf{v} &= \mathbf{D}^{1/2}\widehat{\mathbf{v}}, \\ \mathbf{f}_i &= \mathbf{D}^{-1/2}\widehat{\mathbf{f}}_i; \end{aligned} \tag{15}$$

optimization problems (14) are reformulated as

$$\begin{aligned} \min_{\mathbf{v},\eta,b} \quad & \frac{1}{2}\mathbf{v}^T\mathbf{v} + \frac{C}{2}\sum_{i=1}^{L}\eta_i^2, \\ \text{s.t.} \quad & y_i - \left(\mathbf{v}^T\mathbf{f}_i + b\right) = \eta_i, \quad i = 1, 2, \ldots, L. \end{aligned} \tag{16}$$

The Lagrange function of optimization problems (16) can be expressed as

$$L = \frac{1}{2}\mathbf{v}^T\mathbf{v} + \frac{C}{2}\sum_{i=1}^{L}\eta_i^2 + \sum_{i=1}^{L}\alpha_i\left(y_i - \mathbf{v}^T\mathbf{f}_i - b - \eta_i\right), \tag{17}$$

where $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_L)^T$ is Lagrangian multiplier vector. Then, the KKT system of (17) is

$$\begin{aligned} \frac{\partial L}{\partial \mathbf{v}} &= \mathbf{v} - \sum_{i=1}^{L}\alpha_i\mathbf{f}_i = 0 \Longrightarrow \\ \mathbf{v} &= \sum_{i=1}^{L}\alpha_i\mathbf{f}_i, \end{aligned} \tag{18}$$

$$\begin{aligned} \frac{\partial L}{\partial b} &= -\sum_{i=1}^{L}\alpha_i = 0 \Longrightarrow \\ \sum_{i=1}^{L}\alpha_i &= 0, \end{aligned} \tag{19}$$

$$\begin{aligned} \frac{\partial L}{\partial \eta_i} &= C\sum_{i=1}^{L}\eta_i - \sum_{i=1}^{L}\alpha_i = 0 \Longrightarrow \\ \eta_i &= \frac{1}{C}\alpha_i. \end{aligned} \tag{20}$$

Rewrite (18), (19), and (20) into the form of equation; we can get

$$\begin{pmatrix} \mathbf{H} & \mathbf{F}_1 \\ 0 & \mathbf{H}^T \end{pmatrix}\begin{pmatrix} b \\ \alpha \end{pmatrix} = \begin{pmatrix} \mathbf{y} \\ 0 \end{pmatrix}, \tag{21}$$

where

$$\begin{aligned} &\mathbf{y} = \left(y_1, y_2, \ldots, y_L\right)^T, \\ &H = \text{ones}\,(1, L)^T, \\ &\mathbf{F}_1 \\ &= \begin{pmatrix} \langle\mathbf{f}_1, \mathbf{f}_1\rangle + \dfrac{1}{C} & \langle\mathbf{f}_1, \mathbf{f}_2\rangle & \cdots & \langle\mathbf{f}_1, \mathbf{f}_L\rangle \\ \langle\mathbf{f}_2, \mathbf{f}_1\rangle & \langle\mathbf{f}_2, \mathbf{f}_2\rangle + \dfrac{1}{C} & \cdots & \langle\mathbf{f}_2, \mathbf{f}_L\rangle \\ \vdots & \vdots & \vdots & \\ \langle\mathbf{f}_L, \mathbf{f}_1\rangle & \langle\mathbf{f}_L, \mathbf{f}_2\rangle & \cdots & \langle\mathbf{f}_L, \mathbf{f}_L\rangle + \dfrac{1}{C} \end{pmatrix}. \end{aligned} \tag{22}$$

b and α can be got by solving linear system (21). Then, $\mathbf{v}$ is obtained according to (18) and the right projection matrix $\mathbf{U}$ is accessed through the relation among $\mathbf{v}$, (13) and (15). In summary, when we fix $\mathbf{v}_i$, $i = 1, 2, \ldots, K$, the solution of optimization problems (12) can be computed by solving linear system (21) directly.

Similarly, when $\mathbf{V} = (\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_K)$ is fixed, we can also change the formulation of our algorithm in optimization problems (12) and derive its optimal $\mathbf{U}$ and b by solving another linear system. That is to say, we need to compute a set of $\mathbf{u}_j$ for $j = 1, 2, \ldots, K$ and b. Firstly, denote

$$\widehat{\mathbf{g}}_i = \begin{pmatrix} \mathbf{X}_i^T\mathbf{v}_1 \\ \mathbf{X}_i^T\mathbf{v}_2 \\ \vdots \\ \mathbf{X}_i^T\mathbf{v}_K \end{pmatrix}_{I_2K\times 1},$$

$$\hat{\mathbf{u}} = \begin{pmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \\ \vdots \\ \mathbf{u}_K \end{pmatrix}_{I_2K\times 1}, \quad \mathbf{Q} = \left(\mathbf{V}^T\mathbf{V}\right) \otimes I_{I_2\times I_2}; \tag{23}$$

optimization problems (12) is equivalent to

$$\begin{aligned} \min_{\hat{\mathbf{u}},\eta,b} \quad & \frac{1}{2}\hat{\mathbf{u}}^T Q\hat{\mathbf{u}} + \frac{C}{2}\sum_{i=1}^{L}\xi_i^2 \\ \text{s.t.} \quad & y_i - \left(\hat{\mathbf{u}}^T\hat{\mathbf{g}}_i + b\right) = \xi_i, \quad i = 1, 2, \ldots, L. \end{aligned} \tag{24}$$

And let

$$\begin{aligned} \mathbf{u} &= \mathbf{Q}^{1/2}\hat{\mathbf{u}}, \\ \mathbf{g}_i &= \mathbf{Q}^{-1/2}\hat{\mathbf{g}}_i; \end{aligned} \tag{25}$$

the optimization problems (24) are reformulated as

$$\begin{aligned} \min_{\mathbf{v},\eta,b} \quad & \frac{1}{2}\mathbf{u}^T\mathbf{u} + \frac{C}{2}\sum_{i=1}^{L}\xi_i^2, \\ \text{s.t.} \quad & y_i - \left(\mathbf{u}^T\mathbf{g}_i + b\right) = \xi_i, \quad i = 1, 2, \ldots, L. \end{aligned} \tag{26}$$

The Lagrange function of optimization problems (26) can be expressed as

$$G = \frac{1}{2}\mathbf{u}^T\mathbf{u} + \frac{C}{2}\sum_{i=1}^{L}\eta_i^2 + \sum_{i=1}^{L}\beta_i\left(y_i - \mathbf{u}^T\mathbf{g}_i - b - \xi_i\right), \tag{27}$$

where $\beta = (\beta_1, \beta_2, \ldots, \beta_L)^T$ is Lagrangian multiplier vector. Then, the KKT system of (27) is

$$\begin{aligned} \frac{\partial G}{\partial \mathbf{u}} &= \mathbf{u} - \sum_{i=1}^{L}\beta_i\mathbf{g}_i = 0 \Longrightarrow \\ \mathbf{u} &= \sum_{i=1}^{L}\beta_i\mathbf{g}_i, \end{aligned} \tag{28}$$

$$\begin{aligned} \frac{\partial G}{\partial b} &= -\sum_{i=1}^{L}\beta_i = 0 \Longrightarrow \\ \sum_{i=1}^{L}\beta_i &= 0, \end{aligned} \tag{29}$$

$$\begin{aligned} \frac{\partial G}{\partial \xi_i} &= C\sum_{i=1}^{L}\xi_i - \sum_{i=1}^{L}\beta_i = 0 \Longrightarrow \\ \xi_i &= \frac{1}{C}\beta_i. \end{aligned} \tag{30}$$

Rewrite (28), (29), and (30) into the form of equation; we can get

$$\begin{pmatrix} \mathbf{H} & \mathbf{F}_2 \\ 0 & \mathbf{H}^T \end{pmatrix}\begin{pmatrix} b \\ \beta \end{pmatrix} = \begin{pmatrix} \mathbf{y} \\ 0 \end{pmatrix}, \tag{31}$$

where

$$\mathbf{y} = (y_1, y_2, \ldots, y_L)^T,$$

$$\mathbf{H} = \text{ones}(1, L)^T,$$

$$\mathbf{F}_2 = \begin{pmatrix} \langle \mathbf{g}_1, \mathbf{g}_1\rangle + \frac{1}{C} & \langle \mathbf{g}_1, \mathbf{g}_2\rangle & \cdots & \langle \mathbf{g}_1, \mathbf{g}_L\rangle \\ \langle \mathbf{g}_2, \mathbf{g}_1\rangle & \langle \mathbf{g}_2, \mathbf{g}_2\rangle + \frac{1}{C} & \cdots & \langle \mathbf{g}_2, \mathbf{g}_L\rangle \\ \vdots & \vdots & \vdots & \\ \langle \mathbf{g}_L, \mathbf{g}_1\rangle & \langle \mathbf{g}_L, \mathbf{g}_2\rangle & \cdots & \langle \mathbf{g}_L, \mathbf{g}_L\rangle + \frac{1}{C} \end{pmatrix}. \tag{32}$$

Repeating the iterative operation until convergence, the weight matrix $\mathbf{W}$ is obtained by (11), and the predictor is

$$y_i = \langle \mathbf{W}, \mathbf{X}_i\rangle + b = \left\langle \mathbf{V}\mathbf{U}^T, \mathbf{X}_i\right\rangle + b = \sum_{j=1}^{K}\mathbf{v}_j^T\mathbf{X}_i\mathbf{u}_j + b. \tag{33}$$

According to the above description, we can summarize the following algorithm.

Algorithm 1 (LS-SMRM).

(1) Training Process

Input. Training set (8).

Output. Left and right projection matrices $\mathbf{V}$, $\mathbf{U}$, and bias b.

(a) Initialize $\mathbf{U} = (\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_K)$, $u_i \in R^{I_1}$, $i = 1, 2, \ldots, K$, $K = 2, 3, \ldots, \min\{I_1, I_2\}$.

(b) Formulate $\mathbf{f}_i$, $i = 1, 2, \ldots, L$. by (13).

(c) Alternatively update $\mathbf{V}$ and $\mathbf{U}$ until convergence.

(1) Update $\mathbf{V}$ and b.

(1.1) Get α and b by solving linear system (21).
(1.2) Get $\mathbf{v}$ from (19).
(1.3) Get $\hat{\mathbf{v}}$ from (15).
(1.4) Get $\mathbf{v}$ from (13).

(2) Update $\mathbf{U}$ and b.

(2.1) Get β and b by solving linear system (31).
(2.2) Get $\mathbf{u}$ from (28).
(2.3) Get $\hat{\mathbf{u}}$ from (25).
(2.4) Get $\mathbf{u}$ from (23).

(2) Testing Process

Input. Testing point $\mathbf{X}$, left and right projection matrices $\mathbf{V}$, $\mathbf{U}$, and bias b.

Output. The value of testing set: y

$$y = \langle \mathbf{W}, \mathbf{X}\rangle + b = \left\langle \mathbf{V}\mathbf{U}^T, \mathbf{X}\right\rangle + b = \sum_{j=1}^{K}\mathbf{v}_j^T\mathbf{X}\mathbf{u}_j + b. \tag{34}$$

4. LS-STRM Based on Submatrix of the Tensor (LS-STRM-SMT)

In this section, we propose least square support tensor regression machine, shorten as LS-STRM, for the regression problem with the tensor input. In fact, with the introduction of submatrix of the tensor, the LS-STRM problem is turned into I_3 LS-SMRM problems which are independent. However, the right project matrices should be made equal to fit the practical situation. That is to say, we need to solve I_3 LS-SMRM problems with the same right projection matrix. To show the effectiveness of the proposed algorithm, we will provide some deep analyses in the section.

4.1. Formulation. Give a training set

$$T = \{(\mathcal{X}_1, y_1), (\mathcal{X}_2, y_2), \ldots, (\mathcal{X}_L, y_L)\}, \tag{35}$$

where $\mathcal{X}_i \in R^{I_1 \times I_2 \times I_3}$ is the input and $y_i \in R$ is the output, $i = 1, 2, \ldots, L$. Our task is to find a predictor

$$y = \langle \mathcal{W}, \mathcal{X} \rangle + b, \tag{36}$$

where $\mathcal{W}$ denotes the weight tensor and b is the bias. For a new input tensor, we can predict its output through predictor (36). For convenience, we set $b = 0$.

Definition 2. For an M-order tensor $\mathcal{X} \in R^{I_1 \times I_2 \cdots \times I_M}$, the submatrix of the tensor is defined as

$$\mathrm{mat}_1 (\mathcal{X}(:, :, \ldots, :, k)), \quad k = 1, 2, \ldots, I_M. \tag{37}$$

Particularly, when $M = 3$, the submatrices of a third-order tensor are indeed the frontal slices.

However, we do not construct the model for training set (35) directly. We transform the training set (35) into I_3 training sets similar to training set (8) by the introduction of the submatrix of the tensor and then construct I_3 regression problems. That means, for the training set (35), every input tensor $\mathcal{X}_i \in R^{I_1 \times I_2 \times I_3}$, $i = 1, 2, \ldots, L$, can be regarded as an abstract vector

$$\left(\mathbf{X}_{i(1)}, \mathbf{X}_{i(2)}, \ldots, \mathbf{X}_{i(I_3)}\right), \quad i = 1, 2, \ldots, L, \tag{38}$$

where $\mathbf{X}_{i(j)}$, $j = 1, 2, \ldots, I_3$, is the jth frontal slice of the tensor $\mathcal{X}_i$. Next, we take the jth frontal slice of each tensor $\mathcal{X}_i$, $i = 1, 2, \ldots, L$, out and construct the following training set:

$$\mathbf{T}_j = \left\{\left(\mathbf{X}_{i(j)}, Y_{i(j)}\right)\right\}, \quad i = 1, 2, \ldots, L, \tag{39}$$

where $Y_{i(j)} \in R$ denotes the jth element of Y_i, $Y_i = (y_{i(1)}, y_{i(2)}, \ldots, y_{i(I_3)})^T$ is a vector obey normal distribution with mean y_i and variance σ.

According to training set (39), I_3 optimization problems are constructed as follows:

$$\begin{aligned} \min_{\mathbf{W}_j} \quad & \frac{1}{2} \left\|\mathbf{W}_j\right\|_F^2 + \frac{C_j}{2} \sum_{i=1}^{L} \eta_{i(j)}^2 \\ \text{s.t.} \quad & y_{i(j)} - \mathrm{tr}\left(\mathbf{W}_j^T \mathbf{X}_{i(j)}\right) = \eta_{i(j)}, \quad j = 1, 2, \ldots, I_3, \end{aligned} \tag{40}$$

where $\mathbf{W}_j = \mathcal{W}(:, :, j) = \mathbf{V}_j \mathbf{U}_j^T$, $\mathbf{X}_{i(j)} = \mathcal{X}_i(:, :, j)$, $j = 1, 2, \ldots, I_3$, $i = 1, 2, \ldots, L$. So we can get $\mathbf{W}_{(1)} = (\mathbf{W}_1, \mathbf{W}_2, \ldots, \mathbf{W}_{I_3})$, and then the weight tensor can be obtained by

$$\mathcal{W} = \cdot^{\langle 1 \rangle} \left(\mathbf{W}_{(1)}\right). \tag{41}$$

It is clear that the I_3 model is independent, but this is contrary to the truth. In order to reflect the relation among them, we set

$$\mathbf{U}_1 = \mathbf{U}_2 = \cdots = \mathbf{U}_{I_3} = \mathbf{U}, \tag{42}$$

so that the models can fit the practical situation better. The I_3 models can be expressed together as follows:

$$\begin{aligned} \min_{\mathbf{U}, \mathcal{W}} \quad & \frac{1}{2} \|\mathcal{W}\|_F^2 + \frac{C_j}{2} \sum_{i=1}^{L} \sum_{j=1}^{I_3} \eta_{i(j)}^2 \\ \text{s.t.} \quad & y_{i(j)} - \mathrm{tr}\left(\mathbf{W}_j^T \mathbf{X}_{i(j)}\right) = \eta_{i(j)}, \\ & \qquad i = 1, 2, \ldots, L, \ \ j = 1, 2, \ldots, I_3, \end{aligned} \tag{43}$$

where $\mathbf{W}_j = \mathcal{W}(:, :, j) = [\mathbf{V}_j \mathbf{U}^T]$, $\mathbf{X}_{i(j)} = \mathcal{X}_i(:, :, j)$, $C_j = C$, $i = 1, 2, \ldots, L$, $j = 1, 2, \ldots, I_3$. That means what we need to solve are optimization problems (43).

4.2. Dual Fixed Point Algorithm (DFPA) for LS-STRM-SMT. Fixing $\mathbf{V}_1, \mathbf{V}_2, \ldots, \mathbf{V}_{j-1}, \mathbf{V}_{j+1}, \ldots, \mathbf{V}_{I_3}, \mathbf{U}$ and optimizing $\mathbf{V}_j$, optimization problems (43) can be reformulated as follows:

$$\begin{aligned} \min_{\mathbf{V}_j} \quad & \frac{1}{2} \left\|\mathbf{V}_j \mathbf{U}^T\right\|_F^2 + \frac{C}{2} \sum_{i=1}^{L} \eta_{ij}^2 \\ \text{s.t.} \quad & y_{ij} - \left(\sum_{k=1}^{K} \mathbf{v}_{kj}^T \mathbf{X}_{i(j)} \mathbf{u}_k\right) = \eta_{ij}, \quad i = 1, 2, \ldots, L, \end{aligned} \tag{44}$$

where $\mathbf{V}_j = (\mathbf{v}_{1j}, \mathbf{v}_{2j}, \ldots, \mathbf{v}_{Kj})$, $\mathbf{U} = (\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_K)$.

That is to say, a series of problems similar to optimization problems (44) which are indeed LS-SMRMs rather than optimization problems (43) are needed to be solved.

Fixing $\mathbf{V}_j$, $j = 1, 2, \ldots, I_3$ and optimizing $\mathbf{U}$. Similarly, when $\mathbf{V}_j$, $j = 1, 2, \ldots, I_3$, is fixed, optimization problems (43) can be reformulated as follows:

$$\min_{\mathbf{U}} \quad \frac{1}{2} \left\|\mathbf{P}\mathbf{U}^T\right\|_F^2 + \frac{C}{2} \sum_{i=1}^{L} \eta_i^2 \tag{45}$$

$$\begin{aligned} \text{s.t.} \quad & y_i - \left(\sum_{k=1}^{K} \mathbf{p}_k^T \mathrm{mat}_1(\mathcal{X})\, \mathbf{q}_k\right) = \eta_i, \\ & \qquad i = 1, 2, \ldots, L, \end{aligned} \tag{46}$$

where

$$\mathbf{P} = [\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_K] = \begin{pmatrix} \mathbf{V}_1 \\ \mathbf{V}_2 \\ \vdots \\ \mathbf{V}_k \end{pmatrix}_{I_1 I_3 \times K}, \quad \mathbf{Q} = [\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_K] = \begin{pmatrix} \mathbf{W}_1 \\ \mathbf{W}_2 \\ \vdots \\ \mathbf{W}_k \end{pmatrix}_{I_1 I_3 \times K}. \tag{47}$$

It is clear that $\mathbf{U}$ can be obtained by solving optimization problems (45)-(46) with Algorithm 1. This will lead Algorithm 3.

Algorithm 3 (LS-STRM-SMT).

(1) Training Process

Input. Training set (35).

Output. $\mathbf{V}_1, \mathbf{V}_2, \ldots, \mathbf{V}_{I_3}$, and $\mathbf{U}$.

(a) Construct the I_3 number of new submatrix training sets (39).

(b) Give a positive integer K $(K = 2, 3, \ldots, \min\{I_1, I_2\})$.

(c) Initialize $\mathbf{V}_1, \mathbf{V}_2, \ldots, \mathbf{V}_{i-1}, \mathbf{V}_{i+1}, \ldots, \mathbf{V}_{I_3}$ and $\mathbf{U}, V_j \in R^{I_1 \times K}$, $j \in \{1, 2, \ldots, i-1, i+1, \ldots, I_3\}$, $\mathbf{U} \in R^{I_2 \times K}$.

(d) Alternatively update $\mathbf{V}_1, \mathbf{V}_2, \ldots, \mathbf{V}_{I_3}$ and $\mathbf{U}$ until convergence.

(1) Update $\mathbf{V}_i$, $i = 1, 2, \ldots, \mathrm{I}_3$, by solving optimization problems (44) with Algorithm 1.

(2) Update $\mathbf{U}$ by solving optimization problems (45)-(46) with Algorithm 1.

(2) Testing Process

Input. Testing point $\mathcal{X}$, $\mathbf{V}_1, \mathbf{V}_2, \ldots, \mathbf{V}_i, \mathbf{V}_{i+1}, \ldots, \mathbf{V}_{I_3}$, and $\mathbf{U}$.

Output. The value of testing point: y

$$y = \langle \mathcal{W}, \mathcal{X} \rangle = \sum_{j=1}^{I_3} \langle \mathcal{W}(:, :, j), \mathbf{X}(:, :, j) \rangle = \sum_{j=1}^{I_3} \left\langle \mathbf{V}_j \mathbf{U}^T, \mathbf{X}(:, :, j) \right\rangle. \tag{48}$$

4.3. Extension. For a more general tensor, that means $M > 3$. We can also take advantage of the submatrix of the tensor to turn the tensor learning for regression problems into I_M LS-SMRM problems that can be solved by Algorithm 1. The details can be shown as follows.

Table 1: Characters of different data sets.

Data	#trainings	#attributes
Artificial data	50	31
Slump	103	9
Ticdata2000	200	85
ConcreteData	100	9
BlogData	300	280

Give a training set

$$T = \{(\mathcal{X}_1, y_1), (\mathcal{X}_2, y_2), \ldots, (\mathcal{X}_L, y_L)\}, \tag{49}$$

where $\mathcal{X}_i \in R^{I_1 \times I_2 \times \cdots \times I_M}$, $(M > 3)$ is the input and $y_i \in R$ is the output, $i = 1, 2, \ldots, L$. Our task is to find a predictor

$$y = \langle \mathcal{W}, \mathcal{X} \rangle + b, \tag{50}$$

where $\mathcal{W} \in R^{I_1 \times I_2 \times \cdots \times I_M}$, $(M > 3)$ denotes the weight tensor, b is the bias. For a new input tensor, we can predict its output through predictor (50). The new training set is constructed through (1) as follows:

$$\mathbf{T}_j = \left\{ \left(\mathrm{mat}_1 (\mathcal{X}(:, :, \ldots, :, i)), Y_{i(j)} \right) \right\}, \quad i = 1, 2, \ldots, L, \ j = 1, 2, \ldots, I_M, \tag{51}$$

where $\mathrm{mat}_1(\mathcal{X}(:, :, \ldots, :, i)) \in R^{I_1 \times (I_2 I_3 \cdots I_M)}$ denotes the jth submatrix of the ith sample, $Y_{i(j)}$ denotes the jth element of Y_i which is a L-dimension vector obey normal distribution with mean y_i and variance σ. Then we can get a I_M-dimension abstract vector whose elements are matrices of the size $I_1 \times (I_2 I_3 \cdots I_{M-1})$. According to mapping function (2), the matrices can be reformulated as equal number of $(M-1)$-order tensors. And then the abstract vector with $(M-1)$-order tensor elements is indeed a M-order tensor that we need to get.

5. Numerical Experiment

In the following numerical experiments, we use four groups of vector data from the UCI database and an artificial tensor data for evaluation of our algorithm. The data are Slump, Ticdata2000, ConcreteData, and BlogData from the UCI database. The artificial data is given by the function "rand" in Matlab. We reformulate the vector data into matrices or tensors by rearranging the order of vector's elements. The detailed statistical characters are listed in Table 1.

Since our method is solved in a fixed point algorithm or in other words an alternative way, the final solution has close relationship with initialization. We would like to introduce two different kinds of initialization strategies in this part. The first initialization is random. We generate some random elements to form the matrices with corresponding sizes. In the following experiments, this kind of initialization is used without specification. The second initialization is an empirical method. That means we can set $\mathbf{V}_i = [\mathbf{I}_{K \times K}, \mathbf{0}_{K \times (I_i - K)}]$, $i = 1, 2, \ldots, M$.

Table 2: MSE of different K for Ticdata2000.

Size	$K = 2$	$K = 3$	$K = 4$	$K = 5$	$K = 6$
$3 \times 4 \times 7$	0.0079 ($\pm 2.097e-05$)	0.0064 ($\pm 8.997e-05$)	0.0042 ($\pm 5.627e-05$)	0.0061 ($\pm 4.244e-05$)	**0.0037** (**$\pm 1.173e-05$**)
$3 \times 7 \times 4$	0.0070 ($\pm 2.225e-06$)	**0.0055** (**$\pm 3.918e-06$**)	0.0073 ($\pm 9.876e-06$)	0.0082 ($\pm 6.824e-06$)	**0.0055** (**$\pm 9.405e-07$**)
$4 \times 3 \times 7$	0.0074 ($\pm 6.088e-06$)	0.0061 ($\pm 1.237e-05$)	**0.0049** (**$\pm 7.482e-06$**)	0.0075 ($\pm 1.843e-05$)	0.0079 ($\pm 4.441e-06$)
$4 \times 7 \times 3$	0.0076 ($\pm 4.419e-06$)	0.0085 ($\pm 4.998e-06$)	0.0066 ($\pm 3.798e-06$)	0.0060 ($\pm 7.888e-06$)	**0.0059** (**$\pm 1.367e-05$**)
$7 \times 3 \times 4$	0.0076 ($\pm 8.519e-06$)	0.0077 ($\pm 2.120e-05$)	0.0065 ($\pm 1.374e-06$)	0.0056 ($\pm 6.836e-06$)	**0.0051** (**$\pm 1.365e-05$**)
$7 \times 4 \times 3$	**0.0059** (**$\pm 9.144e-06$**)	0.0090 ($\pm 1.603e-06$)	0.0078 ($\pm 2.004e-05$)	0.0078 ($\pm 6.602e-06$)	0.0083 ($\pm 1.071e-05$)

Table 3: MSE of different K for artificial data.

Size	$K = 2$	$K = 3$	$K = 4$	$K = 5$	$K = 6$
$2 \times 3 \times 5$	0.2302 (± 0.0151)	**0.1722** (**± 0.0025**)	0.2106 (± 0.0106)	0.2432 (± 0.0011)	0.2640 (± 0.0057)
$2 \times 5 \times 3$	0.2270 (± 0.0011)	0.2143 (± 0.0026)	0.1975 (± 0.0110)	**0.1540** (**± 0.0019**)	0.1857 (± 0.0084)
$3 \times 2 \times 5$	0.1787 (± 0.0071)	0.2177 (± 0.0053)	**0.1519** (**± 0.0038**)	0.1607 (± 0.0056)	0.1865 (± 0.0029)
$3 \times 5 \times 2$	**0.1601** (**± 0.0017**)	0.1685 (± 0.0046)	0.1734 (± 0.0006)	0.1886 (± 0.0242)	0.1880 (± 0.0029)
$5 \times 2 \times 3$	0.2301 (± 0.0091)	0.2104 (± 0.0032)	0.1884 (± 0.0055)	0.1902 (± 0.0096)	**0.1839** (**± 0.0067**)
$5 \times 3 \times 2$	0.1999 (± 0.0039)	0.2033 (± 0.0135)	0.1976 (± 0.0052)	0.1413 (± 0.0002)	**0.1114** (**± 0.0023**)

Table 4: MSE of different algorithm for different data set.

Data	SVR	LS-SVRM	RMR	LASSO	LS-STRM-SMT
Artificial data	1.8015 (± 0.3047)	0.3347 (± 0.0148)	*0.7574* *(± 0.1778)*	0.2190 (± 0.0016)	***0.2043*** (***± 0.0063***)
Slump	27.2552 (± 119.9840)	0.4482 (± 0.0052)	*0.1970* *(± 0.0014)*	0.3126 (± 0.0057)	**0.1931** (**± 0.0011**)
Ticdata2000	0.1149 ($\pm 3.923e-04$)	0.00742 ($\pm 1.193e-04$)	*0.1270* *(± 0.0027)*	0.0557 ($\pm 0.720e-05$)	***0.0072*** (***$\pm 8.561e-05$***)
ConcreteData	0.2114 (± 0.0024)	0.1972 (± 0.0080)	*0.2026* *(± 0.0041)*	0.1823 (± 0.0010)	**0.1459** (**± 0.0115**)
BlogData	2.9480 (± 0.6374)	0.5119 (± 0.0010)	*8.9211* *(± 0.1616)*	0.5212 (± 0.0077)	***0.2899*** (***± 0.0041***)

When solving the LS-STRM-SMT, penalty parameter C is selected from the set $\{2^{-8}, 2^{-7}, \ldots, 2^{7}, 2^{8}\}$. Another important parameter that may influence the result is K. Experiment performances show that the mean square error (MSE) which we used to measure whether the solutions to the LS-STRM-SMT problems are good or not decrease as the growth of the K value. That means the larger K is, the smaller MSEs are. But, this phenomenon is not very obvious and when K increases the computational complexity will rise quickly. We will give the MSE of different K for two data sets. And in the other experiments, we simply set $K = 2$. Besides, the parameter C can be obtained by fivefold cross validation. In this section, we compute the tensor learning regression problems for third-order data with our LS-STRM-SMT algorithm and compare it with some other regression method, three problems are talked about. First, how does the MSE for different data change with the increasing of K? Second, compare our algorithm with support vector regression (SVR) [16], least square support vector regression (LSSVR) [17], regularized matrix regression RMR, [18], and lasso regression [19, 20]. The last problem which is being talked about is the convergence of our algorithm.

The information of the data used is listed in Table 1. Tables 2 and 3 show the influence of different K for artificial data and Ticdata2000, respectively, and the comparison of our algorithm and others are in Table 4. From Tables 2 and 3, we

Table 5: RMR for Slump and ConcreteData of different data size, respectively.

Slump	MSE (variance)	ConcreteData	MSE (variance)
2×4	0.1986 (±0.0011)	2×4	0.2058 (±0.0048)
4×2	0.1954 (±0.0016)	4×2	0.1823 (±0.0010)

Table 6: RMR for artificial data, Ticdata2000, and BlogData of different data size, respectively.

Artificial data	MSE (variance)	Ticdata2000	MSE (variance)	BlogData	MSE (variance)
2×15	0.7001 (±0.1825)	2×42	0.1325 (±0.0011)	4×70	9.0560 (±0.1809)
3×10	0.7624 (±0.0505)	3×28	0.1314 (±0.0121)	7×40	10.2741 (±0.1544)
5×6	0.7056 (±0.2212)	4×21	0.1488 (±0.0019)	10×28	9.7221 (±0.1097)
6×5	0.6659 (±0.0577)	6×14	0.0942 (±0.0008)	28×10	7.7979 (±0.1603)
10×3	0.9848 (±0.4923)	7×12	0.1279 (±0.0007)	40×7	8.9549 (±0.1950)
15×2	0.7258 (±0.0626)	12×7	0.1071 (±0.0016)	70×4	7.7215 (±0.1695)
—	—	14×6	0.1364 (±0.0011)	—	—
—	—	21×4	0.1348 (±0.0052)	—	—
—	—	28×3	0.1264 (±0.0031)	—	—
—	—	42×2	0.1304 (±0.0006)	—	—

Table 7: LS-STRM-SMT for BlogData of different data size.

BlogData	MSE (variance)
$4 \times 7 \times 10$	0.2518 (±0.0037)
$4 \times 10 \times 7$	0.3244 (±0.0068)
$7 \times 4 \times 10$	0.3045 (±0.0058)
$7 \times 10 \times 4$	0.2872 (±0.0032)
$10 \times 4 \times 7$	0.2387 (±0.0037)
$10 \times 7 \times 4$	0.3325 (±0.0013)

know that, with the increasing of K, the MSE tends to reduced gradually. But that is not very obvious and the largest K does not always means the best performance. In Figure 2, (a) and (b) show the relation between K and MSE more intuitively. From (a) and (b) in Figure 3 which is the average MSE of different K for artificial data and Ticdata2000, respectively. We can easily come to the conclusion that when K become large, the change of MSE is not that big, so we can set $K = 2$ to reduce computation complexity.

In Table 4, the variance is represented by the number in parentheses while numbers out of the parentheses denote the MSE of different algorithm for different data set. Since the vector sample can be reformulated as matrix (tensor) of different size, numbers with italic font mean the average value of different data size. The numbers which are bold denote the best performance. It is clear that our LS-STRM-SMT always has a better performance which is obvious in Figure 4. More details are included in Tables 5, 6, and 7.

In order to show the convergence behavior, we computerize the objective value of our method for artificial data and Ticdata2000. After 20 times iterations, the objective function values are shown in Figure 1. There are mainly two observations from the result. First, the objective values of our algorithm are extremely small, which implies that our algorithm performs good to some degree. Second, with the increasing of iteration, our algorithm tends to be a certain number which guarantee the convergence of our algorithm.

In Figure 1, the abscissa and the ordinate represent the iterations and the objective value, respectively, and the curve with different color and mark in the same figure stands for the relationship between objective value and iterations for the same data of different data size.

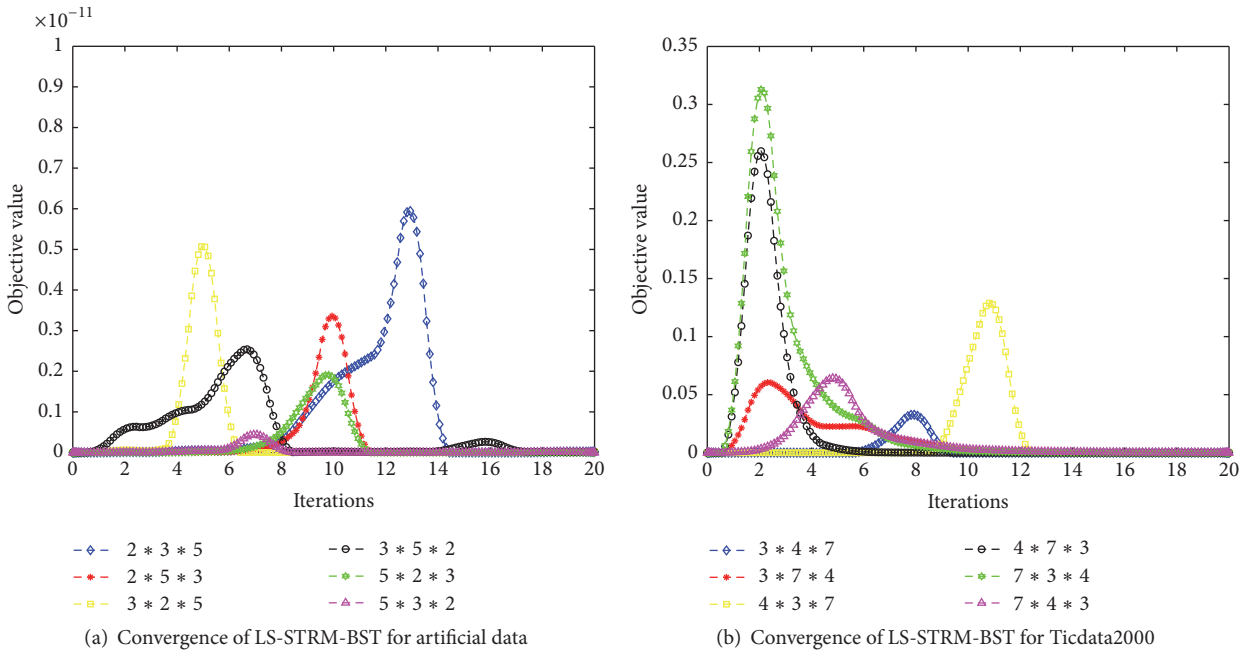
In Figure 2, the abscissa and the ordinate represent the parameter K and the MSE, respectively, and the line graph with different color and mark in the same fig stands for the relationship between the parameter K and the MSE for the same data of different data size.

Figure 3 show the relationship between average MSE and K for the artificial data and Ticdata2000, respectively.

Figure 4 is plotted based on Tables 2, 3, and 4. In (a) the full line stands for the performance of our algorithm while the imaginary line stands for the best performance of other algorithm. In (b) the bar graph in blue represents the performance of our algorithm while the bar graph in brown represents the best performance of other algorithm. From Figure 4, we can easily come to the conclusion that our method has a better performance.

6. Conclusion

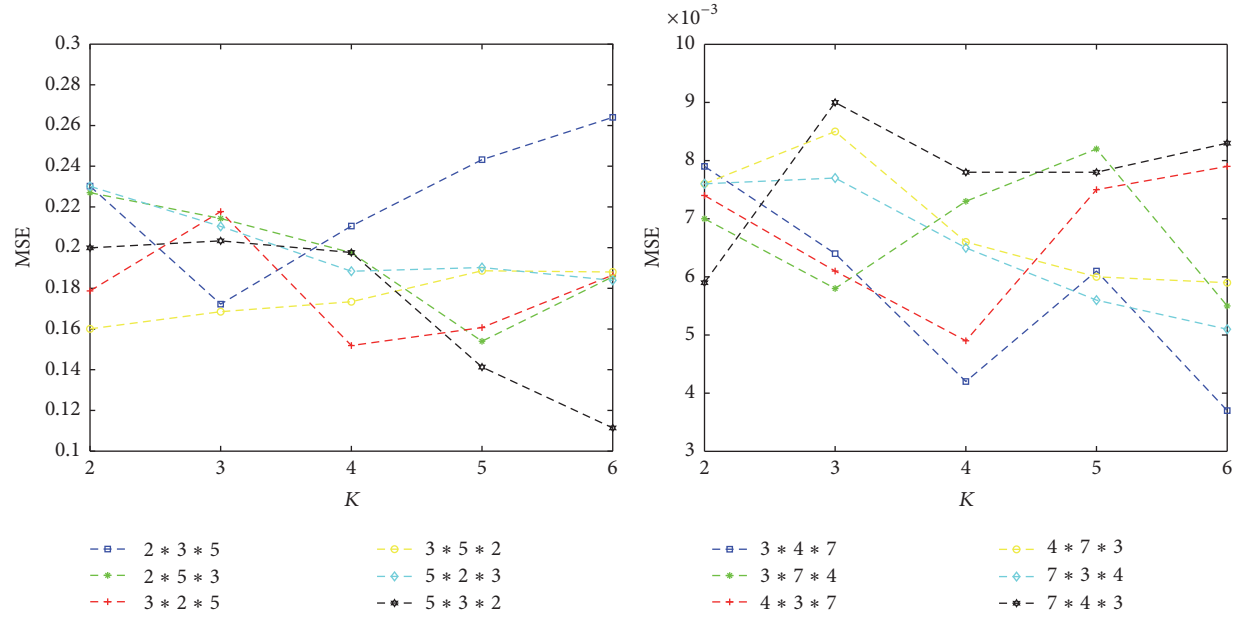
In this paper, we propose a novel method for tensor learning regression problems, called least square support tensor regression machine based on submatrix of the tensor (LS-STRM-SMT), which is inspired by the idea of multiple rank multilinear SVM for matrix data classification (MRMLSVM). And we give our dual fixed point algorithm (DFPA) to solve the problems. However, LS-STRM-SMT is not a straightforward extension of MRMLSVM. On one hand, MRMLSVM is a method for matrix data classification while LS-STRM-SMT is used to solve tensor data regression problems; on the other hand, the LS-STRM-SMT and MRMLSVM are connected



(a) Convergence of LS-STRM-BST for artificial data

(b) Convergence of LS-STRM-BST for Ticdata2000

Figure 1: Convergence of LS-STRM-BST.

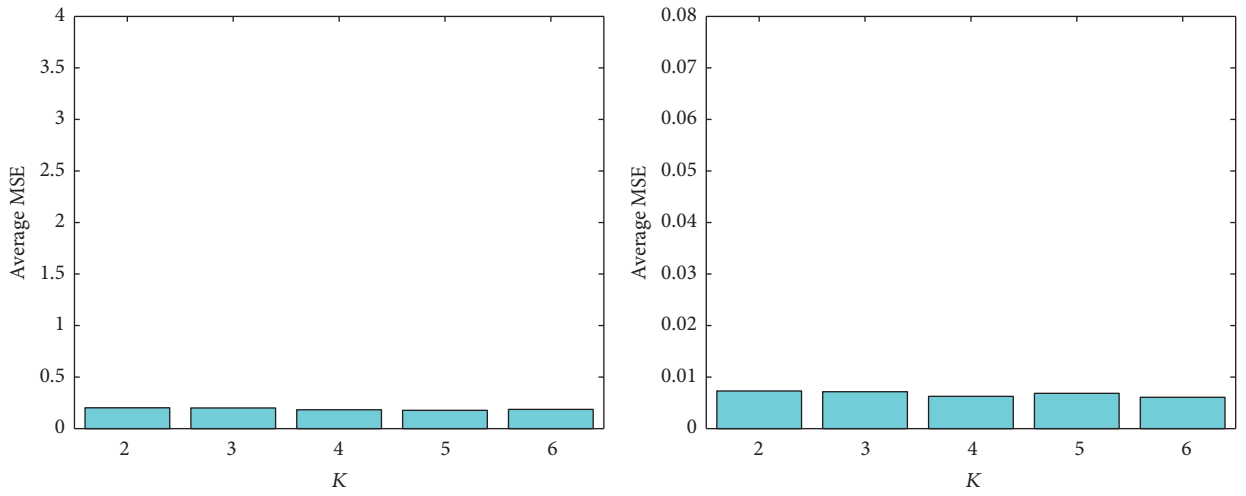


(a) MSE of different K for artificial data

(b) MSE of different K for Ticdata2000

Figure 2: MSE of different K for artificial data and Ticdata2000.

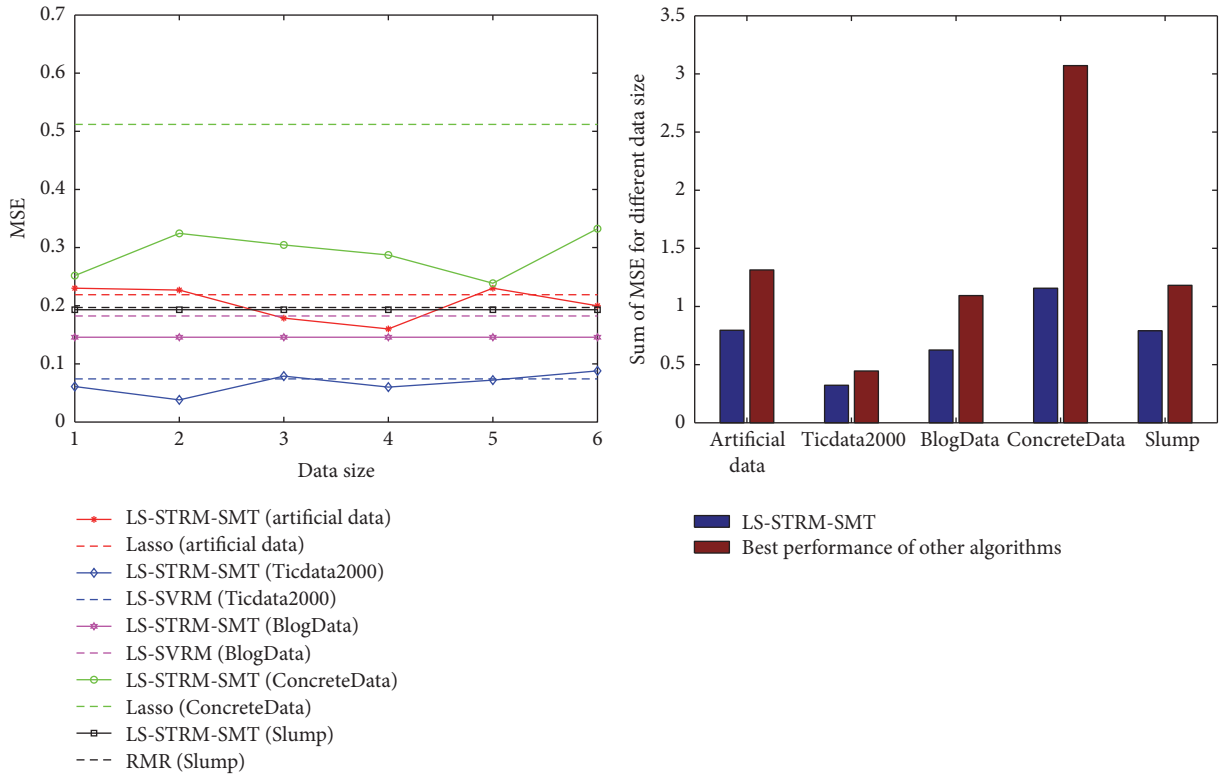
by the introducing of submatrix of the tensor which can be applied to reformulate the LS-STRM-SMT problem as a series of LS-SMRM problems that can be figure out in a way similar to MRMLSVM. What is more, LS-STRM-SMT differs from traditional approach which destroys the structural information of the tensor totally; it can reserve the structure tensor information of the tensor to some extent which may be the reason why our method has a better performance than some other proposed algorithms. Besides, a small parameter K can decrease the dimensions of the unknown quantity which can avoid overfitting to some degree. In addition, from a practical point of view, it has also been shown by the preliminary numerical experiments that the performance of our LS-STRM-SMT is better.



(a) MSE of different K for artificial data

(b) MSE of different K for Ticdata2000

Figure 3: MSE of different K for artificial data and Ticdata2000.



(a) Line graph according to Tables 2, 3, and 4

(b) Bar graph according to Tables 2, 3, and 4

Figure 4: Comparison of our algorithm and the other algorithm that performs best.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (no. 11561066).

References

[1] J. Yang, D. Zhang, and A. F. Frangi, "Two-dimensional PCA: a new approach to apperence -based face representation and recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 1, pp. 131–137, 2004.

[2] A. Kong, D. Zhang, and M. Kamel, "A survey of palmprint recognition," *Pattern Recognition*, vol. 42, no. 7, pp. 1408–1418, 2009.

[3] J. J. Koo, A. C. Evans, and W. J. Gross, "3-D brain MRI tissue classification on FPGAs," *IEEE Transactions on Image Processing*, vol. 18, no. 12, pp. 2735–2746, 2009.

[4] D. Le Bihan, J.-F. Mangin, C. Poupon et al., "Diffusion tensor imaging: concepts and applications," *Journal of Magnetic Resonance Imaging*, vol. 13, no. 4, pp. 534–546, 2001.

[5] R. Abraham, J. E. Marsden, and R. T. Manifolds, *Manifolds, tensor analysis, and applications*, vol. 2 of *Global Analysis Pure and Applied: Series B*, Addison-Wesley Publishing Co., Reading, Mass., 1983.

[6] W. Guo, I. Kotsia, and I. Patras, "Tensor learning for regression," *IEEE Transactions on Image Processing*, vol. 21, no. 2, pp. 816–827, 2012.

[7] Z. Hua, L. Lexin, and Z. Hongtu, "Tensor Regression with Applications in Neuroimaging Data Analysis," *Journal of the American Statistical Association*, vol. 108, no. 502, pp. 540–552, 2013.

[8] J. Wright, A. Yang Y, and A. Ganesh, "Robust Face Recognition via Sparse Representation," *IEEE Transactions on Pattern Analysis Machine Intelligence*, vol. 31, no. 2, pp. 210–227, 2008.

[9] G. Lu, D. Zhang, and K. Wang, "Palmprint recognition using eigenpalms features," *Pattern Recognition Letters*, vol. 24, no. 9-10, pp. 1463–1467, 2003.

[10] D. Cai, X. He, and J. Wen R, "Support tensor machines for text categorization," *International Journal of Academic Research in Business & Social Sciences*, vol. 2, no. 12, pp. 2222–6990, 2006.

[11] Z. Hao, L. He, B. Chen, and X. Yang, "A linear support higher-order tensor machine for classification," *IEEE Transactions on Image Processing*, vol. 22, no. 7, pp. 2911–2920, 2013.

[12] C. Hou, F. Nie, C. Zhang, D. Yi, and Y. Wu, "Multiple rank multilinear SVM for matrix data classification," *Pattern Recognition*, vol. 47, no. 1, pp. 454–469, 2014.

[13] M. Signoretto, Q. Tran Dinh, L. De Lathauwer, and J. A. Suykens, "Learning with tensors: a framework based on convex optimization and spectral regularization," *Machine Learning*, vol. 94, no. 3, pp. 303–351, 2014.

[14] K. Goebel and W. A. Kirk, "A fixed point theorem for asymptotically nonexpansive mappings," *Proceedings of the American Mathematical Society*, vol. 35, pp. 171–174, 1972.

[15] H. Pirsiavash, D. Ramanan, and C. Fowlkes, "Bilinear classifiers for visual recognition," in *Proceedings of the 23rd Annual Conference on Neural Information Processing Systems, NIPS 2009*, pp. 1482–1490, December 2009.

[16] A. J. Smola and B. Lkopf, "A tutorial on support vector regression," *Statistics and Computing*, vol. 14, no. 3, pp. 199–222, 2004.

[17] J. A. K. Suykens, L. Lukas, P. Van et al., *Least Squares Support Vector Machine Classifiers: a Large Scale Algorithm*, 2000.

[18] H. Zhou and L. Li, "Regularized matrix regression," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 76, no. 2, pp. 463–483, 2014.

[19] R. Tibshirani, "Regression shrinkage and selection via the lasso: a retrospective," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 73, no. 3, pp. 273–282, 2011.

[20] S. Boyd, *Alternating Direction Method of Multipliers*, 2011.