LEAST SQUARES SUPPORT TENSOR MACHINE

Meng Lv, Xinbin Zhao, Lujia Song, Haifa Shi, Ling Jing

Department of Applied Mathematics, College of Science, CAU, Beijing 100083, China
lvmeng0122@163.com, zhaoxinbin1986811@163.com, biggirlsong@126.com, shihaifa@163.com, jingling_student@163.com

Keywords: Tensor representation, Alternating projection, Least squares support vector machine, Support tensor machine, Least squares support tensor machine.

Abstract

Least squares support vector machine (LS-SVM), as a variant of the standard support vector machine (SVM) operates directly on patterns represented by vector and obtains an analytical solution directly from solving a set of linear equations instead of quadratic programming (QP). Tensor representation is useful to reduce the overfitting problem in vector-based learning, and tensor-based algorithm requires a smaller set of decision variables as compared to vector-based approaches. Above properties make the tensor learning specially suited for small-sample-size (S3) problems. In this paper, we generalize the vector-based learning algorithm least squares support vector machine to the tensor-based method least squares support tensor machine (LS-STM), which accepts tensors as input. Similar to LS-SVM, the classifier is obtained also by solving a system of linear equations rather than a QP. LS-STM is based on the tensor space, with tensor representation, the number of parameters estimated by LS-STM is less than the number of parameters estimated by LS-SVM, and avoids discarding a great deal of useful structural information. Experimental results on some benchmark datasets indicate that the performance of LS-STM is competitive in classification performance compared to LS-SVM.

1 Introduction

Support vector machine (SVM) is a general learning method which is based on statistical learning theory (SLT) [1, 2], and has been proven to be more powerful than existing methods in many aspects. As a least squares version of SVM, least squares support vector machine (LS-SVM) proposed by Suykens and Vandewalle [3, 4] finds the optimal hyperplane by solving a set of linear equations rather than a quadratic programming (QP), which leads to lower computational cost.

In the machine learning community, high dimensional data with many attributes are often encountered in the real applications. How to represent the data is one of the cores of machine learning. Most of the traditional learning algorithms are based on the vector space model, such as SVM and LS-SVM [5]. But in many computer vision applications, many objects are naturally represented by multidimensional arrays, i.e., tensors [6]. For example, gray image and gray image sequence can be regarded as 2nd and 3rd tensor (the examples of them can be seen in Figure 1.(a) and Figure 2.(b), respectively).

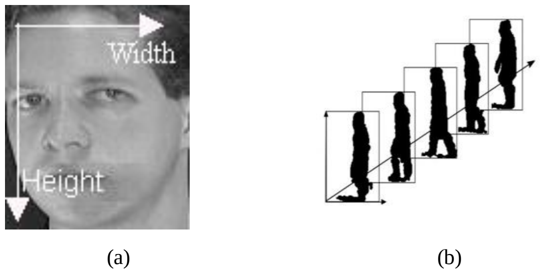

Figure 1. (a) A gray face image is a 2nd order tensor, which is also a matrix. **(b)** The 3rd order tensor representation of a gait sequence.

It is reasonable to consider that pixels close to each other are correlated to some extent. But the traditional methods for solving the problem with tensor data in machine learning were always to scan the tensor into vector, thus it is easy to destroy the data structure and generate high dimensional vectors. In recent years, the machine learning based on tensor space has attracted significant interest from the research community. Several algorithms have been extended to deal with tensors, such as support tensor machine (STM) [7-17], multi-linear principal component analysis (MPCA) [18], multi-linear discriminant analysis (MDA) [19], canonical analysis correlation of tensor (CAC) [20] and non-negative tensor factorization (NTF) [21].

With utilizing the tensor representation, the number of parameters estimated by the tensor-based learning can be greatly reduced. Therefore, the tensor-based learning algorithms are especially suitable for solving the small-sample-size (S3) problem, which the number of samples available for training is small and the number of input features used to represent the data is large. At the same time, involving high-dimensional data can also reduce the computational complexity observed in problems.

In this paper, we propose a novel method called least squares support tensor machine (LS-STM), which is a tensor version of LS-SVM or a least squares version of STM. LS-STM is based on the tensor space, which directly accepts tensor as inputs, without vectorization. Obtaining a classifier in the tensor space not only retains the data structure information, but also helps overcome the overfitting problem encountered mostly in vector-based learning. Compared with STM, LS-STM solves a system of learning equations in every iterative rather than a QP, which eventually converges to an optimal solution after a few iterations.

To examine the effectiveness of the proposed LS-STM, we employ LS-STM as a classifier to two benchmark datasets. The final experimental results indicate that the performance of the LS-STM is competitive compared to LS-SVM.

The rest of the paper is organized as follows. An overview of LS-SVM and STM are provided in Section 2, which prepare well for introducing LS-STM. Section 3 provides the model and the algorithm of LS-STM. Section 4 demonstrates experimental results. The conclusion and future work are drawn in Section 5.

2 An Overview of LS-SVM and STM

2.1 Least Squares Support Vector Machine

Least squares support vector machine (LS-SVM) proposed by Suykens and Vandewalle [3, 4] is a variant of SVM, in which the inequality constraints in SVM are converted into equality constraints. Similar to SVM, LS-SVM aims at finding the optimal hyperplane that maximizes the margin between two classes. The training of the LS-SVM is done by solving a set of linear equations, instead of a quadratic programming problem. Hence, LS-SVM provides a valid reduction in the computational time and provides competitive testing accuracy with that of SVM.

For traditional two-class classification problem, the training set is

$$T=\{(x_1,y_1),\cdots,(x_l,y_l)\}\in(\mathbb{R}^n\times \mathrm{Y})^l, \tag{1}$$

where $x_i\in\mathbb{R}^n$ is a training sample, $y_i\in\mathrm{Y}=\{+1,-1\}$ is the class label of x_i $(i=1,\cdots,l)$ and.

The task is to find the decision function

$$f(x)=\sum_{i=1}^{l}\alpha_i y_i K(x,x_i)+b, \tag{2}$$

to derive the value of y for any input x, where α_i is a real constant and b is the bias term, $K(x,x_i)$ is a kernel function. The LS-SVM is formulated as follows:

$$\begin{aligned}\min_{w,b,\eta}\quad & \frac{1}{2}\|w\|^2+\frac{C}{2}\sum_{i=1}^{l}\eta_i^2,\\ \text{s.t.}\quad & y_i(w^{\mathrm{T}}\Phi(x_i)+b)=1-\eta_i, i=1,\cdots,l.\end{aligned} \tag{3}$$

where C is a regularization constant which controls the trade-off between the two terms, w is the weight vector, Φ is a nonlinear mapping that maps vector x into the feature space and $\eta_i(i=1,\cdots,l)$ are slack variables, which unlike SVM, can be negative.

For solving the optimization problem (3), we construct the Lagrangian function as follows:

$$\begin{aligned}L(w,b,\eta,\alpha)&=\frac{1}{2}\|w\|^2+\frac{C}{2}\sum_{i=1}^{l}\eta_i^2\\ &-\sum_{i=1}^{l}\alpha_i[y_i(w^{\mathrm{T}}\Phi(x_i)+b)]+\sum_{i=1}^{l}\alpha_i(1-\eta_i),\end{aligned} \tag{4}$$

where $\alpha_i(i=1,\cdots,l)$ are Lagrange multipliers.

The necessary conditions for the optimality are:

$$\begin{cases}\dfrac{\partial L}{\partial w}=0\Rightarrow w=\sum_{i=1}^{l}\alpha_i y_i\Phi(x_i),\\ \dfrac{\partial L}{\partial b}=0\Rightarrow \sum_{i=1}^{l}\alpha_i y_i=0,\\ \dfrac{\partial L}{\partial \eta_i}=0\Rightarrow \alpha_i=C\eta_i, i=1,\cdots,l,\\ \dfrac{\partial L}{\partial \alpha_i}=0\Rightarrow y_i(w^{\mathrm{T}}\Phi(x_i))+b)-1+\eta_i=0.\end{cases} \tag{5}$$

By eliminating w and η_i, the solution is given by:

$$\begin{bmatrix}0 & -Y^{\mathrm{T}}\\ Y & H+C^{-1}I\end{bmatrix}\begin{bmatrix}b\\ \alpha\end{bmatrix}=\begin{bmatrix}0\\ e\end{bmatrix}, \tag{6}$$

here $Y=[y_1,\cdots,y_l]^{\mathrm{T}}$, $H_{ij}=y_iy_j\Phi(x_i)^{\mathrm{T}}\Phi(x_j)$ and $e=[1,\cdots,1]^{\mathrm{T}}$.

Like the support vector machine, by setting

$$\Phi(x_i)^{\mathrm{T}}\Phi(x_j)=K(x_i,x_j). \tag{7}$$

can be avoided the explicit treatment of variables in the feature space.

2.2 Support Tensor Machine

Because of the shortcomings of support vector machine, support tensor machine has been developing these years [8, 11, 12]. It extends support vector machine to accept general tensor as input. For two-class classification problem, the training set is

$$\mathrm{T}=\{(X_1,y_1),\cdots,(X_l,y_l)\}\in(\mathbb{R}^{n_1}\otimes\mathbb{R}^{n_2}\times\mathrm{Y})^l. \tag{8}$$

where $X_i\in\mathrm{R}^{n_1}\otimes\mathrm{R}^{n_2}$ is a training sample and $y_i\in\{+1,-1\}$ is the class label of X_i. STM aims at finding a multilinear decision function

$$f(X)=u^{\mathrm{T}}Xv+b \tag{9}$$

such that the two classes can be separated with maximum margin, where $u\in\mathrm{R}^{n_1}$, $v\in\mathrm{R}^{n_2}$ and $b\in\mathrm{R}$ are unknown and can be solved through the primal problem defined below:

$$\begin{aligned}\min_{u,v,b,\xi}\quad & \frac{1}{2}\|uv^{\mathrm{T}}\|_{\mathrm{F}}^2+C\sum_{i=1}^{l}\eta_i,\\ \text{s.t.}\quad & y_i(u^{\mathrm{T}}X_iv+b)\ge 1-\eta_i,\\ & \eta_i\ge 0,\qquad i=1,\cdots,l.\end{aligned} \tag{10}$$

where $C>0$ is a regularization parameter and η_i is a slack variable. It should be emphasized that the weights tensor uv^{T} is a rank-one matrix.

The full algorithm of STM is stated below:

Algorithm 1: Support tensor machine (STM).

1.Initialization: Let $u=(1,\cdots,1)^{\mathrm{T}}$, $\beta_1=\|u\|^2$ and $x_i=X_i^{\mathrm{T}}u$,

2. Computing v: v can be computed by solving the following problem:

$$\begin{aligned}\min_{v,b,\eta}\ & \frac{1}{2}\beta_1 v^{\mathrm{T}}v + C\sum_{i=1}^{l}\eta_i,\\ \text{s.t.}\ & y_i(v^{\mathrm{T}}x_i+b)\ge 1-\eta_i,\\ & \eta_i\ge 0, i=1,\cdots,l.\end{aligned} \tag{11}$$

3. Computing u : By step 2 , let $\beta_2=\|v\|^2$ and $\tilde{x}_i = X_i v$, u can be computed by solving the following problem:

$$\begin{aligned}\min_{u,b,\eta}\ & \frac{1}{2}\beta_2 u^{\mathrm{T}}u + C\sum_{i=1}^{l}\eta_i,\\ \text{s.t.}\ & y_i(u^{\mathrm{T}}\tilde{x}_i+b)\ge 1-\eta_i,\\ & \eta_i\ge 0, i=1,\cdots,l.\end{aligned} \tag{12}$$

4. Iteratively computing u and v: By step 2 and step 3, we can iteratively compute u and v until they tend to converge.

Note: The problems (11) and (12) are the standard SVM. Thus, any computational method for SVM can also be used here.

It is easy to see that u and v are dependent on each other, and can not be solved independently. Alternating iterative algorithm is an effective method to solve the problem. The details of the algorithm of STM and the convergence proof of it can be seen [11, 12].

3 Least Squares Support Tensor Machine

As an extension of STM, we develop a least squares version of STM or a tensor version of LS-SVM, which accepts tensors as inputs. By using a least squares error term in the objective function, the inequality constraints in STM are converted into equality constraints, therefore, we obtain the solution that only needs to solve a system of linear equations rather than a QP. Based on the characteristic of optimization problem, we use alternating iterative algorithm, which requires the determination of a significantly smaller set of decision variables as compared to vector-based approaches. The above properties make LS-STM specially suited for small sample-size (S3) problems. The training set of LS-STM is the same as (8), the training sample X_i is a second order tensor (matrix). LS-STM aims at constructing a classifier with the decision function of the form (9).

The optimization problem of LS-STM is given by:

$$\begin{aligned}\min_{u,v,b,\eta}\ & \frac{1}{2}\|uv^{\mathrm{T}}\|_{\mathrm{F}}^2 + \frac{C}{2}\sum_{i=1}^{l}\eta_i^2,\\ \text{s.t.}\ & y_i(u^{\mathrm{T}}X_i v+b)=1-\eta_i, i=1,\cdots,l.\end{aligned} \tag{13}$$

where C is a trade-off parameter between the margin maximization and empirical error minimization, η_i is a slack variable.

For solving LS-STM (13), we consider its Lagrangian function as:

$$\begin{aligned}L(u,v,b,\eta,\alpha) &= \frac{1}{2}\|uv^{\mathrm{T}}\|_{\mathrm{F}}^2 + \frac{C}{2}\sum_{i=1}^{l}\eta_i^2\\ &\quad -\sum_{i=1}^{l}\alpha_i[y_i(u^{\mathrm{T}}X_i v+b)-1+\eta_i],\end{aligned} \tag{14}$$

Due to

$$\begin{aligned}\frac{1}{2}\|uv^{\mathrm{T}}\|_{\mathrm{F}}^2 &= \frac{1}{2}\mathrm{Tr}(uv^{\mathrm{T}}vu^{\mathrm{T}})\\ &= \frac{1}{2}(v^{\mathrm{T}}v)\mathrm{Tr}(uu^{\mathrm{T}})\\ &= \frac{1}{2}(v^{\mathrm{T}}v)(u^{\mathrm{T}}u),\end{aligned} \tag{15}$$

The Lagrangian function (14) can be rewritten as follows:

$$\begin{aligned}L(u,v,b,\eta,\alpha) &= \frac{1}{2}(u^{\mathrm{T}}u)(v^{\mathrm{T}}v) + \frac{C}{2}\sum_{i=1}^{l}\eta_i^2\\ &\quad -\sum_{i=1}^{l}\alpha_i[y_i(u^{\mathrm{T}}X_i v+b)-1+\eta_i],\end{aligned} \tag{16}$$

Consider KKT conditions, we get:

$$u = \frac{\sum_{i=1}^{l}\alpha_i y_i X_i v}{v^{\mathrm{T}}v}, \tag{17}$$

$$v = \frac{\sum_{i=1}^{l}\alpha_i y_i X_i u}{u^{\mathrm{T}}u}, \tag{18}$$

$$\sum_{i=1}^{l}\alpha_i y_i = 0, \tag{19}$$

$$\eta_i = \alpha_i / C, i=1,\cdots,l. \tag{20}$$

From equations (17) and (18), it is obviously that u and v rely on each other, and cannot be solved with traditional methods. Like STM, we use alternating iterative algorithm.

Algorithm 2: Least Squares Support tensor machine (LS-STM).

1. Initialization: Let $u=(1,\cdots,1)^{\mathrm{T}}$, $\beta_1=\|u\|^2$ and $x_i = X_i^{\mathrm{T}}u$.

2. Computing v : v can be computed by solving the optimization problem.

$$\begin{aligned}\min_{v,b,\eta}\ & \frac{1}{2}\beta_1 v^{\mathrm{T}}v + \frac{C}{2}\sum_{i=1}^{l}\eta_i^2,\\ \text{s.t.}\ & y_i(v^{\mathrm{T}}x_i+b)=1-\eta_i, i=1,\cdots,l.\end{aligned} \tag{21}$$

3. Computing u : Let $\beta_2=\|v\|^2$ and $\tilde{x}_i = X_i v$, u can be computed by solving the optimization problem .

$$\begin{aligned}\min_{u,b,\eta}\ & \frac{1}{2}\beta_2 u^{\mathrm{T}}u + \frac{C}{2}\sum_{i=1}^{l}\eta_i^2,\\ \text{s.t.}\ & y_i(u^{\mathrm{T}}\tilde{x}_i+b)=1-\eta_i, i=1,\cdots,l.\end{aligned} \tag{22}$$

4. Iteratively computing u and v : From step 2 and step 3, we cannot stop iteratively compute u and v until the below conditions are satisfy

$$\|u_k - u_{k-1}\| \le \varepsilon,\ \|v_k - v_{k-1}\| \le \varepsilon. \tag{23}$$

Note: The problems (21) and (22) are both LS-SVM. Thus, the method for solving LS-SVM in Part II can be used here. The convergence proof of Algorithm 2 is similar to Algorithm 1.

4 Experimental Results

In this section, we will present experimental results and compare the results of LS-STM with the vector-based classification method LS-SVM and tensor-based classification method STM. These experiments are conducted on the benchmark datasets including both that in a vector representation, for example, the Lung Cancer dataset (56attributes/3classes/32instances), and that in a second order tensor (matrix) representation: Letter Reco-gnition dataset (16attributes/26classes/20000instances).

For each classification problem, ten independent runs are performed and their classification accuracies on the sets are averaged. The sample points we used in the experiment are selected randomly. The parameter C is obtained by cross validation, and the range of the regularization constant C is from 2^{-6} to 2^{6} with each step by multiplying 2. We initialize both u and v are vectors of ones of appropriate dimensions. At the same time, the kernel function between LS-SVM and LS-STM both choose the linear kernel, i.e., $K(x,x_i) = x^{\mathrm{T}} x_i$ with respect to $\Phi(x) = x$.

4.1 Lung Cancer [22]

This dataset has 32 instances of 3 classes, each class has 9, 13, 10 instances, respectively, and each instance has 56 attributes. It is a small-sample-size (S3) dataset. The model we proposed is to dispose two classification problem, so the data we used is processed as follows: we set the points of second class to be the positive class points, and then we set the other two classes into negative class.

In fact, on the Lung Cancer dataset, we must transform a vector data into a matrix. The process of transfer a vector with n attributes to a matrix with the size of $n_1 \times n_2$ where $n = n_1 \times n_2$ is described below. For each sample point we let the vector of the previous n_2 elements to be the first column of the new matrix. Then let n_2+1 to $2n_2$ elements as the second column of the new matrix. By parity of reasoning, we can obtain a second order tensor with the scale of $n_1 \times n_2$.

Here, we use two kinds of matrixization: 7×8 and 8×7, and the average classification accuracy as shown in Table 1.

Table 1. The comparison between matricization 7×8 and 8×7.

Number of samples	Accuracy (%)	
	Matricization (7×8)	Matricization (8×7)
1	46.0	55.3
2	35.7	60.7
3	45.8	59.2
4	50.0	62.5
5	55.5	62.7

From Table 1, it is obviously that the different matricization methods resulted in different test accuracy. When the training set have only two samples (including two positive points and one negative points), the gap of the classification accuracy of LS-STM between on 8×7 matrix pattern outperforms that on 7×8 matrix pattern is the largest, reaching 25%. The main reason of this phenomenon is that the matrix of different size will have different internal structure after matrix transformation. To cite a simple example, we get a vector $A = [123456]$, then transfer it into the matrix of size 2×3 and size 3×2. We express them with A_1 and A_2 respectively. Through the matricization method we proposed above, we can obtain $A_1 = [135;246]$ and $A_2 = [14;25;36]$. Obviously, matrix A_1 and A_2 have different inner structures. It is quite reasonable that different inner structures lead to different results.

In order to directly see the superiority of LS-STM on 8×7 matrix pattern, we also give the comparison between 8×7 matrix pattern and 7×8 matrix pattern in Figure 2. In addition, the horizontal ordinate is the number of samples, the vertical ordinate is accuracy. The results were tested with accuracy calculate tests, but they were unsatisfactory, just like most existing results, such as RDA: 62.5%, KNN: 53.1%, Opt. Disc. Plane: 59.4% [22].

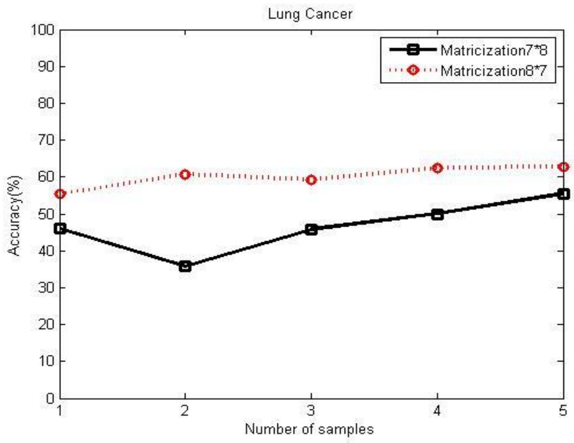

Figure 2. The comparison between matricization 7×8 and 8×7.

The methods based on tensor space are suitable for small-sample-size (S3) problems. Least squares support tensor machine has great advantages in this aspect than some vector-based methods, such as least squares support vector machine. Here, the results of LS-STM and STM are both based on 8×7 matrix pattern. The comparison result between LS-SVM、LS-STM and STM are shown in Table 2 and Figure 3.

Table 2. The comparison between LS-SVM, LS-STM and STM.

Number of samples	Accuracy (%)		
	LS-SVM	LS-STM	STM
1	46.0	55.3	61.7
2	47.1	60.7	57.5
3	41.9	59.2	53.5
4	28.3	62.5	52.5
5	45.5	62.7	55.5
6	34.5	62.5	59.0

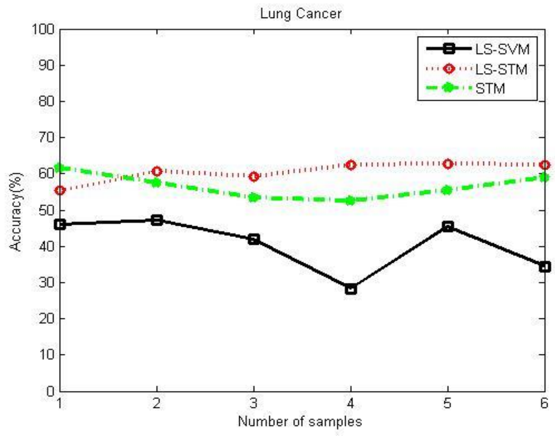

Figure 3. The comparison between LS-SVM, LS-STM and STM.

From both Table 2 and Figure 3, the classification accuracy of LS-STM is better than that of LS-SVM and STM. All accuracy values of LS-SVM are below 50%, and the maximum difference of the accuracy between LS-SVM and LS-STM is 34.17%. However, the maximum difference of the accuracy between STM and LS-STM is only 10%. For the experimental data –Lung Cancer used in this article, there are 56 variables when we use LS-SVM to get a classifier, but only 15 variables by using LS-STM. Thus, we obtain that LS-STM is more suitable for small-sample-size (S3) problems than LS-SVM.

4.2 Letter Recognition [22]

The Letter Recognition dataset has been widely used in the literature. It contains a large number of black-and-white rectangular pixels, each of them displays as one of the 26 capital letters in the English alphabet. It has 20000 instances, and each instance has 16 attributes. In this experiment, we deal with the classification problem of capital letter "A" (789 instances) and "B" (766 instances).

It should be noted that in this experiment the scale of the matricization is 4×4. The comparison results between LS-SVM and LS-STM is shown in Figure 4 below. Here, the number of the training samples is 10 (A: 10 ; B: 10).

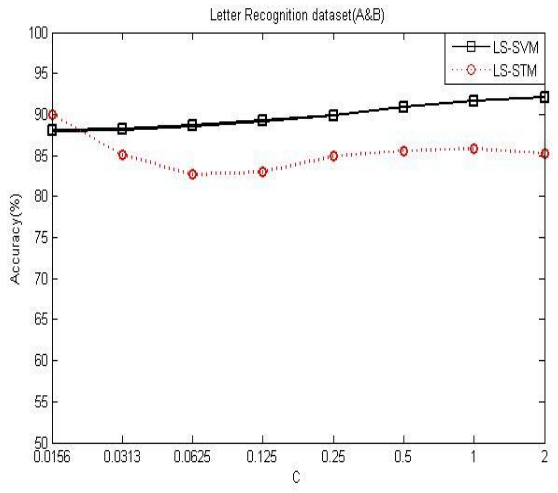

Figure 4. The comparison between LS-SVM and LS-STM on dataset Letter recognition(A&B).

From Figure 4, we can see that the classification accuracies of LS-SVM are all better than that of LS-STM, except the regularization constant C is 2^{-5}. It is mainly because of the fact that the superiority of LS-STM will be reflected obviously when solving the small-sample-size problem. However, the dataset letter recognition (A&B) used in this paper has 1555 sample points with 16 attributes. So it's not a small-sample-size problem. As a result, it is not necessarily that the performance of LS-STM is better than that of LS-SVM, which means LS-STM may show poor performance when solving the problem without high dimension and small sample. Additionally, similarly to the vector pattern dataset, for the image pattern image dataset, whether LS-STM can really increase performance or not depends on different matricization even for the same dataset. From different performance exhibitions for different matricizing patterns on the same datasets, we have that the matricization is the key step to use the methods based on tensor space.

The above two numerical experiments show that the tensor-based methods have more advantages than vector-based methods for small-sample-size (S3) problems. How to use the matricization to improve the classification accuracy is the important issue.

5 Conclusion

This paper reviews the learning algorithm--least squares support vector machine (LS-SVM) which based on vector space, and then we propose tensor-based method--least squares support tensor machine (LS-STM). For solving the small-sample-size (S3) problems, the tensor representation always performs better than the vector representation. Our numerical experiments of the two datasets provide the most powerful support. It is worth noting that the performance of LS-STM deeply depends on the different matricization even for the same datasets. With the development of machine learning, tensor learning has wide applicable range, it is worth further researching.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (NO. 10971223, NO.11071252) and Chinese Universities Scientific Fund (2011JS039, 2012YJ130, 2013YJ010).

References

[1] Vapnik, V., The nature of statistical learning theory. Springer-Verlag, New York,1995.

[2] Vapnik, V., Statistical learning theory. Wiley, J., New York,1998.

[3] Suykens, J.A.K., Vandewalle, J., “Least squares support vector machine classifiers,”Neural Process Lett 9:293-300,1999.

[4] Suykens, J.A.K., Gestel, T.V.J., Brabanter, D.B., Moor, D., Vandewalle, J., “Least squares support vector machines,” World Scientific, Singapore(ISBN 981-238-151-1),2002.

[5] Gerard, S., Wong, A., and Yang, C.S., “A vector space model for information retrieval,” Communications of the ACM, 18(11): 613-620,1975.

[6] Lathauwer, L.D., “Signal processing based on multilinear algebra,” Doctor Thesis, Katholieke Universiteit Leuven, Leuven, Belgium, 1997.

[7] Kotsia, I., Guo, W.W., Patras, I., “Higher rank support tensor machines for visual recognition,” Patter Recognition. 45 (2012) pp.4192-4203.

[8] Tao, D., Li, X., Wu, X., Hu, W., Maybank, S.J., “Supervised tensor learning, Knowledge and Information Systems.,” 13 (2007) pp.1-42.

[9] Kotsia, I., Patras, I., “Support tucker machines,” in Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Colorado, USA, 2011,pp.633-640.

[10] Guo, W.W., Kotsia, I., Patras, I., “Tensor learning for regression,” IEEE Transations on Image Processing, 21 (2012) pp.816-827.

[11] Cai, D., He, X.F., Han, J.W., “Learning with tensor representation,” Department of Computer Science Technical Report No.2716, University of Illinois at Urbana-Champaign(UIUCDCS-R-2006-2716), April 2006.

[12] Cai, D., He, X.F., Wen, J.R., Han, J., Ma, W.Y., “Support tensor machines for text categorization,” Department of Computer Science Technical Report No.2714, University of Illinois at Urbana-Champaign (UIUCDCS-R-2006-2714), April 2006.

[13] Kotsia, I., Patras, I., “Relative margin support tensor machines for gait and action recognition,” in International Conference on Image and Video Retrieval, Xi’an, China, 2010.

[14] Gao, C., Wu, X.J., “Kernel support tensor regression,” Procedia Engineering, 29 (2012) pp.3986-3990.

[15] Daniusis, P., Vaitkus, P., “Kernel regression on matrix patterns,” Lithuanian Mathematical Journal, Spec. edition 48-49 (2008) (1)pp.91–195.

[16] Zhang, X.S., Gao, X.B., Wang, Y., “Twin support tensor machines for MCs detection,” Journal of Electronics (China), 26 (2009) pp.318-325.

[17] Khemchandani, R., Karpatne, A., Chandra, S., “Proximal support tensor machines,” International Journal of Machine Learning and Cybernetics, online, 2012.

[18] Lu, H., Plataniotis, K.N., Venetsanopoulos, A.N., Mpca: multilinear principal component analysis of tensor objects, IEEE Transations on Neural Networks, 19 (2008) pp.18–39.

[19] Yan, S., Xu, D., Yang, Q., Zhang, L., Tang, X., Zhang, H.J., “Multilinear discriminant analysis for face recognition,” IEEE Transations on Image Processing, 16 (2007) pp.212–220.

[20] Kim, T.K., Cipolla, R.. “Canonical correlation analysis of video volume tensors for action categorization and detection,” IEEE Transations on Pattern Analysis and Machine Intelligence, 31 (2009) 1415–1428.

[21] Zafeiriou, S., “Discriminant nonnegative tensor factorization algorithms,” IEEE Transations on Neural Networks, 20 (2009) pp.217–235.

[22] http://archive.ics.uci.edu/ml/datasets.