remote sensing

MDPI

Article

Hyperspectral Image Classification Based on a Least Square Bias Constraint Additional Empirical Risk Minimization Nonparallel Support Vector Machine

Guangxin Liu, Liguo Wang * and Danfeng Liu

College of Information and Communication Engineering, Dalian Minzu University, Dalian 116600, China
* Correspondence: wangliguo@hrbeu.edu.cn

Abstract: Hyperspectral image classification technology is important for the application of hyperspectral technology. Support vector machines (SVMs) work well in supervised classifications of hyperspectral images; however, they still have some shortcomings, and their use of a parallel decision plane makes it difficult to conform to real hyperspectral data distribution. The improved nonparallel support vector machine based on SVMs, i.e., the bias constraint additional empirical risk minimization nonparallel support vector machine (BC-AERM-NSVM), has improved classification accuracy compared its predecessor. However, BC-AERM-NSVMs have a more complicated solution problem than SVMs, and if the dataset is too large, the training speed is significantly reduced. To solve this problem, this paper proposes a least squares algorithm, i.e., the least square bias constraint additional empirical risk minimization nonparallel support vector machine (LS-BC-AERM-NSVM). The dual problem of the LS-BC-AERM-NSVM is an unconstrained convex quadratic programming problem, so its solution speed is greatly improved. Experiments on hyperspectral image data demonstrate that the LS-BC-AERM-NSVM displays a vast improvement in terms of solution speed compared with the BC-AERM-NSVM and achieves good classification accuracy.

Keywords: hyperspectral remote sensing image; supervised classification; least square support vector machine; nonparallel support vector machine

Citation: Liu, G.; Wang, L.; Liu, D. Hyperspectral Image Classification Based on a Least Square Bias Constraint Additional Empirical Risk Minimization Nonparallel Support Vector Machine. *Remote Sens.* **2022**, *14*, 4263. https://doi.org/10.3390/rs14174263

Academic Editors: Xiangrong Zhang, Yansheng Li, Lichao Mou, Licheng Jiao and Xu Tang

Received: 15 July 2022
Accepted: 25 August 2022
Published: 29 August 2022

1. Introduction

The emergence of hyperspectral remote sensing was a milestone in the development of modern remote sensing technology. Hyperspectral remote sensing images have extremely high spectral resolution, which greatly improves the ability to distinguish objects. Therefore, hyperspectral remote sensing technology is widely used in many fields [1]. An important basic content of hyperspectral data analysis and processing is classification, that is, the creation of a unique category for each pixel. Classification is an important means for researchers to use hyperspectral image information, as it can clearly reveal the distribution of ground object information. Therefore, the accuracy of the classification directly affects the accuracy of subsequent information processing. However, hyperspectral image classification [2] still faces a series of challenges, e.g., redundant information, high spectral dimensions and limited training samples. As such, research on hyperspectral image classification technology is still necessary.

Many algorithms can be applied to hyperspectral image classification. According to whether there is prior knowledge of the category attributes of the image samples, the classification approach can be either supervised or unsupervised. The unsupervised method does not require prior knowledge of the sample, but rather, completes the task by performing cluster analyses, yielding statistics on the spectral or spatial characteristics of the image itself. Commonly used unsupervised classification algorithms include K-means [3] and ISODATA [4] clustering. In contrast, supervised classification requires training samples. Through learning via training samples, a specific classification criterion is constructed which

is then used to complete the classification task. Among supervised algorithms, KNN [5], decision trees [6,7], random forests [8], and support vector machines [9–11] are frequently used for hyperspectral data classification.

At present, supervised classification methods are still the mainstream of research, among which the support vector machine (SVM) has been widely studied and used because of its good performance. The SVM is a machine learning algorithm based on the statistical learning theory developed by Vapnik et al. [12]. An SVM can minimize the empirical error and maximize the classification interval. The central idea is to find the optimal plane by maximizing the interval between two parallel support planes in order to achieve supervised learning [13]. An SVM has strong nonlinear and high-dimensional data processing ability and solves the dimension disaster problem. Because of its solid theoretical foundation [14] and good generalization, the SVM has achieved good results in remote sensing image classification settings [15]. Melgani et al. analyzed the classification output from an SVM of hyperspectral data and proved that this approach is an effective alternative to traditional pattern recognition methods, regardless of the adopted multiclass strategy. In recent years, most research on SVM classifications of hyperspectral data has been based on the use of spectral features to further combine spatial features in order to improve the classification accuracy [16]. Chan et al. [17] discussed the classification situation combined with spatial information in an SVM classifier, giving rise to further improvements in the classification accuracy. Jin et al. [18] reported the fast, accurate and nondestructive identification of wheat seeds using SVM classification combined with spatial spectral feature extraction. Suykens et al. [19] changed the inequality constraints of an SVM to equality constraints and proposed a least squares support vector machine (LSSVM). The experimental results showed that the LSSVM required less computation time. Therefore, the LSSVM, as an alternative to the SVM, is widely used in research. At present, many scholars are using the LSSVM algorithm combined with other feature processing algorithms as a basic classifier to study the classification of hyperspectral images. Gao and Heng-Zhen et al. [20] used a LSSVM as a basic classifier and combined the band extraction algorithm to classify hyperspectral images, which improved the classification accuracy and reduced the computational requirements. Shao and Yuanyuan et al. [21] applied hyperspectral imaging to classify wheat grains and extracted spectral information about damaged and healthy grain samples. The features were processed by the principal component analysis (PCA) and continuous projection algorithm (SPA) methods; then, the LSSVM model was used for classification, and good classification accuracy was obtained. However, the parallel plane SVM classification model still has some problems regarding practical remote sensing data, and the distribution of hyperspectral data poses problems in terms of meeting the assumption that parallel planes are separable. The idea of a nonparallel support vector machine was proposed by Jayadeva et al. and implemented as a twin support vector machine (TWSVM) [22–24]. By solving two smaller quadratic programming problems, two nonparallel hyperplanes were determined so that each hyperplane was closer to one class and farther away from the other. As a classical nonparallel support vector, the TWSVM is widely used because of its faster solution speed and the characteristics of the nonparallel decision plane. However, TWSVMs do not perform satisfactorily with hyperspectral images. The original problem only minimizes the empirical risk and does not minimize the structural risk, which affects their generalization performance. Kaya et al. compared the classification of a TWSVM of hyperspectral data with that of a SVM. The TWSVM achieved classification results similar to those of the SVM, but with a shorter training time [25]. Liu et al. combined multifeature optimization and a TWSVM to classify remote sensing images, processed the features of hyperspectral images, and finally, used the TWSVM as a classifier, which improved the classification accuracy of hyperspectral images [26]. The same authors then proposed a new, nonparallel SVM algorithm, the BC-AERM-NSVM, based on the nonparallel plane SVM [27], which now referred to as the BAENSVM. This algorithm considers both empirical and structural risk minimization. The nonparallel characteristics allow it to obtain better classification accuracy than an SVM, and

structural risk minimization leads to better generalization performance than the TWSVM. It achieved better classification accuracy in four classification experiments on commonly used hyperspectral data. However, the BAENSVM solution matrix is larger than that of the SVM, so the solution speed is reduced.

This paper proposes a new support vector machine model based on BAENSVM, i.e., the least square bias constraint additional empirical risk minimization nonparallel support vector machine (LS-BC-AERM-NSVM). It solves the problem of the long computation time of the BAENSVM algorithm when a large number of hyperspectral training datasets are used. LS-BC-AERM-NSVM (hereafter referred to as LSBAENSVM) modifies the original problem of the BAENSVM model, changes the constraint condition to an equality constraint and changes the L1 regularization term of the error variable to the L2 regularization term; as such, the dual problem it solves becomes a convex quadratic programming problem with a pair of unconstrained conditions. Without constraints, the speed of the parameter solution is greatly improved. With an increase in the scale of the hyperspectral training data, the training time of the algorithm remains low.

The remainder of this paper is organized as follows:

Section 2 includes detailed information about the algorithm model itself and the preliminary preparations for the experiment, i.e.:

(1) The selection of experimental software tools and the hardware conditions of the experiment.
(2) A brief introduction of the BC-AERM-NSVM algorithm and a detailed description of the LS-BC-AERM-NSVM algorithm model.
(3) An evaluation index of the experimental results.

Section 3 presents images and analyses of the experimental results, and Section 4 provides a summary of the full text.

2. Materials and Methods

2.1. Software Description

In this research, a computer with an AMD R7 4800 H CPU and 16 GB RAM produced by AMD Corporation in the United States was used. The algorithm model was implemented in the Python 3.8 programming language produced by Python Software Foundation in the United States via in the PyCharm software. The normalization function in the sklearn package was used in the project to preprocess the hyperspectral data. All solutions to systems of linear equations in model calculations were solved using the scipy.linalg.solve() function. All convex quadratic programming problems were solved using the CVXOPT toolkit.

2.2. Data

This paper applied two public hyperspectral remote sensing datasets for experiments. Basic information about the two datasets is presented below.

2.2.1. Indian Pines Dataset

The Indian Pines dataset includes hyperspectral data about images of Indian pine trees in northwest Indiana, USA, made using an airborne visible infrared imaging spectrometer (AVIRIS) in 1992. It has a total of 220 bands, generally excluding 20 bands that cannot be reflected by water; as such, the remaining 200 bands are the research objects. The size of the piece of data is 145 × 145, i.e., a total of 21,025 pixels. There are a total of 10,249 pixels containing a total of 16 types of ground objects. Figure 1 shows a sample band from the Indian Pines dataset.

Figure 1. Sample band from the Indian Pines dataset.

2.2.2. Kennedy Space Center Dataset

The Kennedy Space Center dataset includes hyperspectral data of images of the Kennedy Space Center, Florida, made using an imaging spectrometer (AVIRIS) sensor in 1996, with a total of 224 bands. Generally, 48 bands which are affected by noise are excluded, leaving 176 bands as the research object in this study. The size of each piece of data is 512 × 614, making a total of 314,368 pixels. There are 4811 pixels containing 13 types of ground objects. Figure 2 shows a sample band from the Kennedy Space Center dataset.

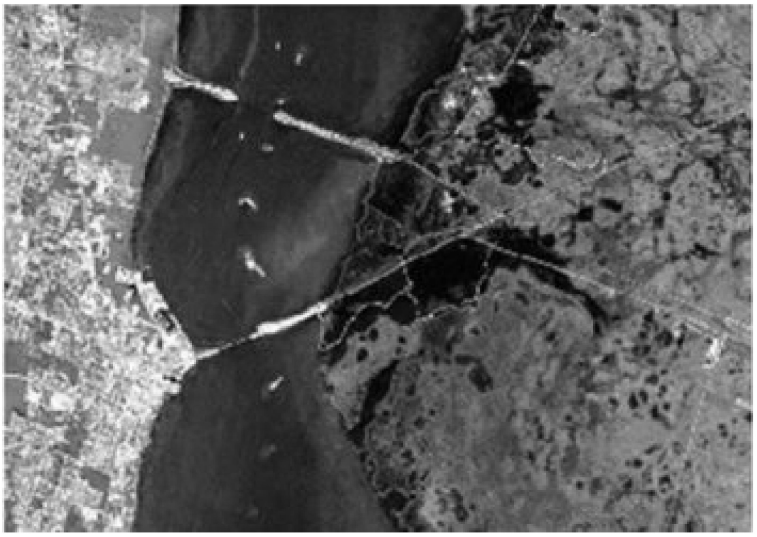


Figure 2. Sample band from the Kennedy Space Center dataset.

2.2.3. Pavia University Dataset

The Pavia University dataset includes hyperspectral data of images of Pavia City, Italy, made using an imaging spectrometer (ROSI) sensor in 2003, with a total of 115 bands, generally excluding 12 bands affected by noise. The remaining 103 bands are the research object of this study. The size of these pieces of data is 610 × 340, making a total of 2,207,400 pixels. There are a total of 42,776 pixels containing nine types of ground objects. Figure 3 shows a sample band from the Pavia University dataset.

Figure 3. Sample band of the Pavia University dataset.

2.2.4. Salinas Dataset

The Salinas dataset includes hyperspectral data of images of Salinas Valley, California, made using an imaging spectrometer (AVIRIS) sensor, with a total of 224 bands, generally excluding 20 bands affected by noise. The remaining 204 bands are the research object of this study. The size of these data is 512 × 217, comprising a total of 105,984 pixels. There are a total of 54,129 pixels containing 16 types of ground objects. Figure 4 shows a sample band from the Salinas dataset.

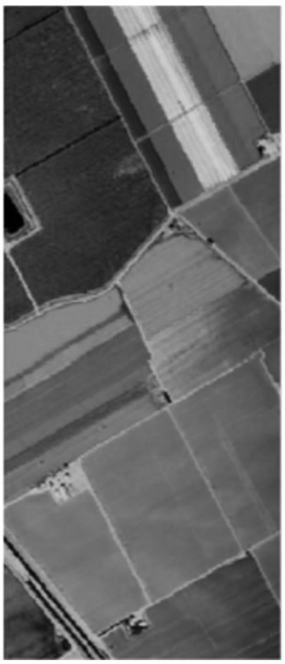


Figure 4. Sample band of the Salinas dataset.

2.3. Bias Constraint Additional Empirical Risk Minimization Nonparallel Support Vector Machine

For the binary classification problem of m datasets in the n-dimensional feature space R^n, matrix C of size $m \times n$ represents all the sample points, and the i-th sample point is

$C_i(i = 1, 2, \ldots, m)$, $C_i = (i = C_{i1}, C_{i2}, \ldots, C_{in})^T$. Assuming that the number of positive samples in all samples is m_+ and the number of negative samples is m_-, all positive samples form a matrix of size $m_+ \times n$, which is represented by A, and all negative samples form a matrix of size $m_- \times n$, represented by B. $y_i \in \{1, -1\}$ represents the category of the sample of the i-th sample point, Y is a diagonal matrix of $m \times m$ size, and y_i forms its diagonal; then, Y_{ii} represents the category information of the C_i sample.

First, consider the linear case of BAENSVM, whose two decision hyperplanes are shown in (1) and (2):

$$f(x) = \omega_+ x + b_+ = 1 \tag{1}$$

$$f(x) = \omega_- x + b_- = -1 \tag{2}$$

Here, $\omega_\pm \in R^n$ and $b \in R^n$.

The original problem of the BAENSVM is obtained by adding the least square and offset constraint terms of the positive and negative samples, respectively, on the basis of the SVM. We added an additional empirical risk minimization term to obtain two nonparallel decision planes that better fit the distribution trend of positive and negative samples to obtain (3) and (4), i.e., the two original problems that need to be optimized.

$$\begin{aligned} &\min_{\omega, b, \xi_+} \tfrac{1}{2}\left(\|\omega_+\|^2 + b_+^2\right) + \tfrac{c_1}{2}\eta_+{}^{\mathrm{T}}\eta_+ + \mathrm{c_3 e^T}\xi_+ \\ &s.t.\ \mathrm{A}\omega_+ + \mathrm{e_+ b_+} = \eta_+ \\ &\mathrm{Y}(\mathrm{C}\omega_+ + \mathrm{e}b_+) + \xi_+ \geq \mathrm{e} \\ &\xi_+ \geq 0. \end{aligned} \tag{3}$$

$$\begin{aligned} &\min_{\omega, b, \xi_-} \tfrac{1}{2}\left(\|\omega_-\|^2 + b_-^2\right) + \tfrac{c_2}{2}\eta_-{}^{\mathrm{T}}\eta_- + \mathrm{c_4 e^T}\xi_- \\ &s.t.\ \mathrm{B}\omega_- + \mathrm{e\_b\_} = \eta_- \\ &\mathrm{Y}(\mathrm{C}\omega_- + \mathrm{e}b_-) + \xi_- \geq \mathrm{e} \\ &\xi_- \geq 0. \end{aligned} \tag{4}$$

where η_+ and η_- are vectors of dimension m, ξ_+ and ξ_- are slack variables, $c_i, i = 1, 2, 3, 4$ is a penalty parameter, e_+ is a vector with all values of 1 and dimension m_+, e_- is a vector with all values of 1 and dimension m_-, and e is a vector with all values of 1 and dimension m.

Using the Lagrange multiplier method to solve the original problems, i.e., (3) and (4), the following dual problems can be obtained:

$$\begin{aligned} &\max_{\lambda, \alpha} \mathrm{e}^T\alpha - \tfrac{1}{2}\left[\lambda^T\ \alpha^T\right]\begin{bmatrix} AA^T + \frac{1}{c_1}I_+ + E_1 & -\left(AC^T + E_2\right)Y^T \\ -Y\left(CA^T + E_3\right) & Y\left(CC^T + E_4\right)Y^T \end{bmatrix}\left[\lambda^T\ \alpha^T\right]^T \\ &s.t.\ 0 \leq \alpha \leq \mathrm{c_3}e^T \end{aligned} \tag{5}$$

$$\begin{aligned} &\max_{\theta, \gamma} \mathrm{e}^T\gamma - \tfrac{1}{2}\left[\theta^T\ \gamma^T\right]\begin{bmatrix} BB^T + \frac{1}{c_2}I_- + F_1 & -\left(BC^T + F_2\right)Y^T \\ -Y\left(CB^T + F_3\right) & Y\left(CC^T + F_4\right)Y^T \end{bmatrix}\left[\theta^T\ \gamma^T\right]^T \\ &s.t.\ 0 \leq \gamma \leq \mathrm{c_4}e^T \end{aligned} \tag{6}$$

where $E_i, i = 1, 2, 3, 4$ represents a matrix of size $m_+ \times m_+, m_+ \times m, m \times m_+, m \times m$, and all the values in the matrix are 1; $F_i, i = 1, 2, 3, 4$ represents a matrix of size $m_- \times m_-, m_- \times m$, $m \times m_-, m \times m$, and the values in the matrices are all 1; I_+ represents the identity matrix of size $m_+ \times m_+$; and I_- represents the identity matrix of size $m_- \times m_-$.

The parameters corresponding to the positive and negative hyperplanes are obtained using Equations (7)–(10).

$$\omega_+ = -A^T\lambda^* + C^TY^T\alpha^* \tag{7}$$

$$\omega_- = B^T\theta^* + C^TY^T\gamma^* \tag{8}$$

$$b_+ = -e_+^T\lambda^* + e^TY^T\alpha^* \tag{9}$$

$$b_- = -e_-^T\theta^* + e^T Y^T \gamma^* \tag{10}$$

where $\lambda^*, \alpha^*, \theta^*, \gamma^*$ are the optimal solutions of the Lagrange multipliers, obtained by solving the Equations (5) and (6).

Next, we determine the category to which the new sample point belonged by calculating the distance from the new sample point to the two decision planes. Setting $b_+ = b_+ - 1$ and $b_- = b_- + 1$, the specific calculation formula is shown in (11).

$$\text{Class} = \arg\min_{i=+,-} \frac{\left|\left(x^T \cdot \omega_i\right) + b_i\right|}{\|\omega_i\|} \tag{11}$$

Nonlinear classification problems can be transformed into a linear classification problem in a certain dimensional feature space through nonlinear transformation, and a linear model can be learned in a high-dimensional feature space. $\phi(x)$ represents the mapping of x to a high-dimensional space. In the dual problems (5) and (6) of linear BAENSVM learning, the nonlinear BAENSVM is obtained using the kernel function instead of the inner product. The dual problem of the nonlinear BAENSVM is as follows:

$$\begin{aligned} &\max_{\alpha} e^T\alpha - \tfrac{1}{2}\left[\lambda^T\ \alpha^T\right]\begin{bmatrix} K\left(AA^T\right) + \frac{1}{c_1}I_+ + E_1 & -\left(K\left(AC^T\right) + E_2\right)Y^T \\ -Y\left(K\left(CA^T\right) + E_3\right) & Y\left(K\left(CC^T\right) + E_4\right)Y^T \end{bmatrix}\left[\lambda^T\ \alpha^T\right]^T \\ &s.t.\ 0 \le \alpha \le c_3 e^T \end{aligned} \tag{12}$$

$$\begin{aligned} &\max_{\alpha} e^T\gamma - \tfrac{1}{2}\left[\theta^T\ \gamma^T\right]\begin{bmatrix} K\left(BB^T\right) + \frac{1}{c_2}I_- + F_1 & -\left(K\left(BC^T\right) + F_2\right)Y^T \\ -Y\left(K\left(CB^T\right) + F_3\right) & Y\left(K\left(CC^T\right) + F_4\right)Y^T \end{bmatrix}\left[\theta^T\ \gamma^T\right]^T \\ &s.t.\ 0 \le \gamma \le c_4 e^T \end{aligned} \tag{13}$$

Similar to the linear case, i.e., $b_+ = b_+ - 1$ and $b_- = b_- + 1$, the corresponding decision function in the nonlinear case is:

$$\text{Class} = \arg\min_{i=+,-} \frac{\left|K\left(x^T \cdot \omega_i\right) + b_i\right|}{\sqrt{K\left(\omega_i^T\omega_i\right)}} \tag{14}$$

2.4. Least Square Bias Constraint Additional Empirical Risk Minimization Nonparallel Support Vector Machine

On the basis of the original problem of the BAENSVM algorithm, the LSBAENSVM replaces inequality constraints with equality constraints, modifies the L1 regularization term of the error variable to the L2 regularization term, and obtains two new original problems. The dual problem corresponding to the new original problem is an unconstrained convex quadratic programming problem, and the absence of constraints greatly improves the efficiency of the solution and reduces the computational cost.

2.4.1. Linear Case

LSBAENSVM also uses two nonparallel classification decision planes as follows:

$$f(x) = \omega_+ x + b_+ = 1 \tag{15}$$

$$f(x) = \omega_- x + b_- = -1 \tag{16}$$

The new original problem is obtained as (17) and (18):

$$\begin{aligned} &\min_{\omega,b,\xi_+} \tfrac{1}{2}\left(\|\omega_+\|^2 + b_+^2\right) + \tfrac{c_1}{2}\eta_+{}^T\eta_+ + \tfrac{c_3}{2}\xi_+^T\xi_+ \\ &s.t.\ A\omega_+ + e_+b_+ = \eta_+ \\ &Y(C\omega_+ + eb_+) + \xi_+ = e \end{aligned} \tag{17}$$

$$\begin{array}{l} \min\limits_{\omega,\mathrm{b},\xi_-} \frac{1}{2}\left(\|\omega_-\|^2 + b_-^2\right) + \frac{c_2}{2}\eta_-{}^{\mathrm{T}}\eta_- + \frac{c_4}{2}\xi_-^{\mathrm{T}}\xi_- \\ s.t.\ \mathrm{B}\omega_- + \mathrm{e\_b}_- = \eta_- \\ \mathrm{Y}(\mathrm{C}\omega_- + \mathrm{e}b_-) + \xi_- = \mathrm{e} \end{array} \tag{18}$$

Using the Lagrangian multiplier method to solve the dual problem to Formula (17), the Lagrangian function is obtained as follows:

$$\begin{array}{l} L(\omega_+, b_+, \xi_+, \alpha, \beta) = \frac{1}{2}\left(\|\omega_+\|^2 + b_+^2\right) + \frac{c_1}{2}\eta_+^T\eta_+ + \mathrm{c}_3\xi_+^T\xi_+ \\ +\lambda^T(A\omega_+ + e_+b_+ - \eta_+) \\ +\alpha^T(e - \xi_+ - Y(C\omega_+ + eb_+)) \end{array} \tag{19}$$

where $\alpha = (\alpha_1, \ldots, \alpha_m), \beta = (\beta_1, \ldots, \beta_m)$ and $\lambda = (\lambda_1, \ldots, \lambda_{m_+})$ are Lagrange multiplier vectors. Taking the partial derivative of the Lagrangian function (19), the KKT conditions are obtained as follows:

$$\nabla_{\omega_+} L = \omega_+ + A^T\lambda - C^TY^T\alpha = 0 \tag{20a}$$

$$\nabla_{b_+} L = b_+ + e_+^T\lambda - e^TY^T\alpha = 0 \tag{20b}$$

$$\nabla_{\eta_+} L = c_1\eta_+ - \lambda = 0 \tag{20c}$$

$$\nabla_{\xi_+} L = c_3e^T - \alpha^T = 0 \tag{20d}$$

Formulas (20a) and (20b) yield:

$$\begin{bmatrix} \omega_+^T & b_+^T \end{bmatrix}^T = \begin{bmatrix} -A^T & C^TY^T \\ -e_+^T & e^TY^T \end{bmatrix} \begin{bmatrix} \lambda^T & \alpha^T \end{bmatrix}^T \tag{21}$$

Putting (20a)–(20d) into the Lagrangian function (19), the following dual formula can be obtained:

$$\max_{\lambda,\alpha} \mathrm{e}^T\alpha - \frac{1}{2}\begin{bmatrix} \lambda^T & \alpha^T \end{bmatrix} \begin{bmatrix} AA^T + \frac{1}{c_1}I_+ + E_1 & -(AC^T + E_2)Y^T \\ -Y(CA^T + E_3) & Y(CC^T + E_4)Y^T + \frac{1}{c_3}J \end{bmatrix} \begin{bmatrix} \lambda^T & \alpha^T \end{bmatrix}^T \tag{22}$$

where J represents an identity matrix of size $m \times m$. The dual problem obtained at this time is an unconstrained convex quadratic programming problem. Since there are no constraints, the solution speed of the problem will be greatly increased.

Accordingly, ω_- and b_- corresponding to the negative class can be obtained:

$$\begin{bmatrix} \omega_-^T & b_-^T \end{bmatrix}^T = \begin{bmatrix} -B^T & C^TY^T \\ -e_-^T & e^TY^T \end{bmatrix} \begin{bmatrix} \theta^T & \gamma^T \end{bmatrix}^T \tag{23}$$

Using the same method, the dual equation corresponding to the negative sample solution can be obtained as follows:

$$\max_{\theta,\gamma} \mathrm{e}^T\gamma - \frac{1}{2}\begin{bmatrix} \theta^T & \gamma^T \end{bmatrix} \begin{bmatrix} BB^T + \frac{1}{c_2}I_- + F_1 & -(BC^T + F_2)Y^T \\ -Y(CB^T + F_3) & Y(CC^T + F_4)Y^T + \frac{1}{c_4}J \end{bmatrix} \begin{bmatrix} \theta^T & \gamma^T \end{bmatrix}^T \tag{24}$$

Next, we determine the class of a new sample by comparing its distance to the positive and negative hyperplanes. Setting $b_+ = b_+ - 1$ and $b_- = b_- + 1$, the corresponding decision function in the nonlinear case is:

$$\mathrm{Class} = \arg\min_{\mathrm{i}=+,-} \frac{\left|(x^T \cdot \omega_i) + \mathrm{b}_i\right|}{\|\omega_i\|} \tag{25}$$

2.4.2. Nonlinear Case

A linear case can be extended to a nonlinear case, such as a support vector machine, by introducing a kernel function into the linear model. The kernel function $K(x, x') = \phi(x) \cdot \phi(x')$ and the corresponding transformation $X = \phi(x)$ are introduced, where $X \in H$, H is the Hilbert space. On the basis of linear problems (22) and (24), two original problems in the nonlinear case can be obtained.

$$\begin{array}{l} \min\limits_{\omega,\mathrm{b},\xi_+} \frac{1}{2}\left(\|\omega_+\|^2 + \mathrm{b}_+^2\right) + \frac{\mathrm{c}_1}{2}\eta_+{}^{\mathrm{T}}\eta_+ + \frac{\mathrm{c}_3}{2}\xi_+{}^{\mathrm{T}}\xi_+ \\ s.t.\phi(\mathrm{A})\omega_+ + \mathrm{e}_+\mathrm{b}_+ = \eta_+ \\ \mathrm{Y}(\ \phi(\mathrm{C})\omega_+ + \ \mathrm{e}b_+) + \xi_+ = \ \mathrm{e} \end{array} \tag{26}$$

$$\begin{array}{l} \min\limits_{\omega,\mathrm{b},\xi_-} \frac{1}{2}\left(\|\omega_-\|^2 + \mathrm{b}_-^2\right) + \frac{\mathrm{c}_2}{2}\eta_-{}^{\mathrm{T}}\eta_- + \frac{\mathrm{c}_4}{2}\xi_-{}^{\mathrm{T}}\xi_- \\ s.t.\phi(B)\ \omega_- + \mathrm{e}\_\mathrm{b}_- = \eta_- \\ \mathrm{Y}(\phi(C)\omega_- + \ \mathrm{e}b_-) + \xi_- = \ \mathrm{e} \end{array} \tag{27}$$

Then, we use the Lagrange multiplier method to solve (26) and (27) and obtain the dual problem as follows:

$$\max_{\lambda,\alpha} \mathrm{e}^T\alpha - \frac{1}{2}\begin{bmatrix}\lambda^T & \alpha^T\end{bmatrix}\begin{bmatrix} K(AA^T) + \frac{1}{c_1}I_+ + E_1 & -(K(AC^T) + E_2)Y^T \\ -Y(K(CA^T) + E_3) & Y(K(CC^T) + E_4)Y^T + \frac{1}{c_3}J \end{bmatrix}\begin{bmatrix}\lambda^T & \alpha^T\end{bmatrix}^T \tag{28}$$

$$\max_{\theta,\gamma} \mathrm{e}^T\gamma - \frac{1}{2}\begin{bmatrix}\theta^T & \gamma^T\end{bmatrix}\begin{bmatrix} K(BB^T) + \frac{1}{c_2}I_- + F_1 & -(K(BC^T) + F_2)Y^T \\ -Y(K(CB^T) + F_3) & Y(K(CC^T) + F_4)Y^T + \frac{1}{c_4}J \end{bmatrix}\begin{bmatrix}\theta^T & \gamma^T\end{bmatrix}^T \tag{29}$$

By solving the dual problems of (28) and (29), two decision hyperplanes can be obtained as follows:

$$-K\left(x^TA^T\right)\lambda^* + \left(x^TC^T\right)Y^T\alpha^* + b_+ = 1 \tag{30}$$

$$-K\left(x^TB^T\right)\theta^* + \left(x^TC^T\right)Y^T\gamma^* + b_- = -1 \tag{31}$$

$$b_+ = -e_+^T\lambda^* + e^TY^T\alpha^* \tag{32}$$

$$b_- = -e_-^T\theta^* + e^TY^T\gamma^* \tag{33}$$

Setting $b_+ = b_+ - 1$ and $b_- = b_- + 1$, the corresponding decision function in the nonlinear case is:

$$\mathrm{Class} = \arg\min_{\mathrm{i}=+,-} \frac{\left|K\left(x^T\cdot\omega_i\right) + \mathrm{b}_i\right|}{\sqrt{K\left(\omega_i^T\omega_i\right)}} \tag{34}$$

where:

$$K\left(x^T\omega_+\right) = -K\left(x^TA^T\right)\lambda^* + K\left(x^TC^T\right)Y^T\alpha^* \tag{35}$$

$$K\left(x^T\omega_-\right) = -K\left(x^TB^T\right)\theta^* + K\left(x^TC^T\right)Y^T\gamma^* \tag{36}$$

$$\begin{array}{l} K\left(\omega_+^T\omega_+\right) = \lambda^{*T}K(AA^T)\lambda^* - \lambda^{*T}K(AC)^TY^T\alpha^* \\ -\alpha^{*T}YK(CA^T)\lambda^* + \alpha^{*T}YK(CC^T)Y^T\alpha^* \end{array} \tag{37}$$

$$\begin{array}{l} K\left(\omega_-^T\omega_-\right) = \theta^{*T}K(BB^T)\theta^* - \theta^{*T}K(BC)^TY^T\gamma^* \\ -\gamma^{*T}YK(CB^T)\theta^* + \gamma^{*T}YK(CC^T)Y^T\gamma^* \end{array} \tag{38}$$

2.5. Application of Algorithms in Hyperspectral Image Classification

A binary classification algorithm is proposed in this paper; however, the applied hyperspectral dataset contains multiclassification data, and the classification algorithm needs

to be extended to a multiclassification situation. Taking the linear case of the LSBAENSVM algorithm as an example, the specific implementation steps are shown in Algorithm 1.

Algorithm 1: The Classification Process of the LSBAENSVM Algorithm Model for a Hyperspectral Dataset

Step 1: Each category of the hyperspectral dataset can be combined into pairs to obtain $\frac{1}{2}(n \times (n-1))$ binary classification tasks.
Step 2: Hyperparameters c_1, c_2, c_3, c_4 of the LSBAENSVM model are set.
Step 3: Each binary classification task is trained using LSBAENSVM.
1. First, we used the parameter set in Step 2 to solve parameters $\alpha^*, \lambda^*, \theta^*, \gamma^*$ according to Formulas (22) and (24). Here, c_1 and c_3 are the two parameters of Formula (22), and c_2 and c_4 are the two parameters of Formula (24).
2. The normal vectors and offsets of the two decision hyperplanes are obtained with (21) and (23).
Finally, $\frac{1}{2}(n \times (n-1))$ classifier models are obtained.
Step 4: For the $\frac{1}{2}(n \times (n-1))$ classifier models trained in Step 3, the category of the new sample is predicted by Formula (25), all predicted categories are recorded, and the sample is classified into the category with the most votes.

2.6. Accuracy Assessment

As a comparison array used to represent accuracy evaluations, the confusion matrix is generally used to evaluate the classification accuracy of hyperspectral data. The columns in the array represent the reference data and the rows represent the category data obtained by classifying the image data. The form is:

$$H = \begin{bmatrix} h_{11} & h_{12} & \cdots & h_{1N} \\ h_{21} & h_{22} & \cdots & h_{2N} \\ \vdots & \vdots & & \vdots \\ h_{N1} & h_{N2} & \cdots & h_{NN} \end{bmatrix} \tag{39}$$

Among them, the overall classification accuracy (OA) and Kappa coefficient are important statistics derived from the confusion matrix that are generally used as evaluation criteria for the classification results. OA represents the ratio of the sum of all correctly classified pixels (i.e., the sum of all values on the diagonal axis in the confusion matrix) to the sum of all pixels. However, OA only considers the number of correctly classified pixels on the diagonal axis in the confusion matrix. The Kappa coefficient comprehensively considers various factors in the confusion matrix and can determine the accuracy of the overall classification; the larger the value of the kappa coefficient, the higher the accuracy of the corresponding classification algorithm. Therefore, the OA and Kappa coefficients are usually used to jointly evaluate the classification accuracy of hyperspectral images.

3. Results

This section shows a comparison of the classification accuracies and solution times of the SVM, TWSVM, BAENSVM, LSSVM, LSTWSVM and LSBAENSVM methods when applied the four hyperspectral datasets, i.e., the Indian Pines, Kennedy Space Center, Pavia University, and Salinas sets. All classification algorithms are applied in a nonlinear fashion with a Gaussian kernel function.

3.1. Indian Pines Dataset

The Indian Pines dataset has a large difference in the number of samples of each category, and the number of some pixels is small. For this dataset, 10%, 20%, 30%, and 40% of each category are selected as training data. The rest of the data are used as the test set. In this way, we can test the computational times of each algorithm using datasets of different sizes and compare the classification accuracies.

The division of the training set of Indian Pines is shown in Table 1.

Table 1. The number of samples in each category of the Indian Pines dataset and the division of training samples.

Class	Samples	10%	20%	30%	40%
Alfalfa	54	5	11	16	22
Corn-notill	1434	143	287	430	574
Corn-mintill	834	83	167	250	334
Corn	234	23	47	70	94
Grass-pasture	497	50	99	149	199
Grass-trees	747	75	149	224	299
Grass-pasture-mowed	26	3	5	8	11
Hay-windrowed	489	49	98	147	196
Oats	20	2	4	6	8
Soybean-notill	968	97	194	290	388
Soybean-mintill	2468	247	494	740	988
Soybean-clean	614	61	123	184	246
Wheat	212	21	42	64	85
Woods	1294	129	259	388	518
Buildings-Grass-Trees-Drives	300	30	60	90	120
Stone-Steel-Towers	95	10	19	29	38
Total	10,286	1029	2057	3086	4120

Figure 5 shows our solution time comparison of the SVM, TWSVM, BAENSVM, LSSVM, LSTWSVM and LSBAENSVM algorithms under the conditions of the usage of 10%, 20%, 30% and 40% of the Indian Pines training dataset. The indicated times are the average values of ten runs.

	10% train rate	20% train rate	30% train rate	40% train rate
SVM	5.14	10.92	22.71	41.52
TWSVM	5.12	10.63	17.63	27.86
BAENSVM	16.56	42.56	106.77	224.33
LSSVM	0.06	0.21	0.51	1.29
LSTWSVM	0.06	0.21	0.52	1.12
LSBAENSVM	0.61	1.54	2.92	4.76

Figure 5. Time-consumption comparison of SVM, TWSVM, BAENSVM, LSSVM, LSTWSVM and LSBAENSVM at different scales using the Indian Pines training dataset.

Figure 5 shows that when 10% of the data are selected as the training dataset and the data size is 1029, the training time of the BAENSVM is approximately three times that of the SVM. This is because the BAENSVM needs to solve two larger-scale matrices than the SVM solving matrix, significantly increasing the computational requirements. At this time, the solution time of the LSBAENSVM proposed in this paper is only 0.61 s, indicating a vast reduction in computational load thanks to the fact that the convex quadratic programming problem it solves has no constraints. In general, the least square forms of SVM, TWSVM and BAENSVM can greatly reduce the complexities of the calculations. The training times of LSSVM, LSTWSVM and LSBAENSVM are only 0.06, 0.06, and 0.61 s, respectively, which shows that introducing the least square form into the algorithm is an effective way to improve the calculation speed. Because LSBAENSVM has a larger calculation data matrix, the calculation complexity is still larger than those of LSSVM and LSTWSVM, but the training time remains modest compared with that of the BAENSVM, and its calculation speed is 27 times faster than BAENSVM using data on this scale. With a continuous increase in the number of training datasets, the solution time of each algorithm increases to varying degrees. Observe the broken line graph of the solution times of the six algorithms under 10%, 20%, 30%, and 40% of the training dataset. With an increase in the scale of the training data, the calculation consumption of the six algorithms increases by different degrees. Because the matrix of the two solving problems of the TWSVM is smaller, the increase in its computational complexity is smaller than that of the SVM. At the same time, the matrix of the two solving problems of the BAENSVM is larger than that of the SVM, so the increase in computational complexity is larger than that of the SVM. At this time, the computational complexities of the LSSVM, LSTWSVM, and LSBAENSVM methods increase very slightly. At 20%, 30%, and 40% of the training dataset, the speed of LSBAENSVM is 27 times, 36 times and 47 times faster than that of BAENSVM, respectively, indicating that the speed advantage of the former is more obvious with an increase in the scale of training data.

Table 2 shows the experimental accuracies of the SVM, TWSVM, BAENSVM, LSSVM, LSTWSVM, and LSBAENSVM methods under training datasets of 10%, 20%, 30%, and 40% scales of the Indian Pines dataset. The best experimental results are shown in bold.

Table 2. Classification results of Indian Pines hyperspectral images.

Train Rate	Accuracy	SVM	TWSVM	BAENSVM	LSSVM	LSTWSVM	LSBAENSVM
10%	OA	82.42	82.79	82.84	84.08	83.49	**84.18**
	Kappa	81.16	81.50	81.70	82.88	82.69	**82.95**
20%	OA	87.10	87.02	87.61	88.63	88.21	**88.78**
	Kappa	86.06	86.07	86.70	87.44	87.38	**87.78**
30%	OA	89.41	89.43	89.79	90.24	90.09	**90.41**
	Kappa	88.48	88.68	88.96	89.43	89.41	**89.50**
40%	OA	90.15	90.02	90.57	91.15	90.65	**91.26**
	Kappa	89.28	89.39	89.76	90.49	89.89	**90.57**

Bold in the table indicates the optimal accuracy.

Figure 6 sequentially shows the restoration graphs under the classification of SVM, TWSVM, BAENSVM, LSSVM, LSTWSVM, and LSBAENSVM, corresponding to the classification results presented in Table 2.

Table 2 shows the experimental accuracies of the six classification models. Figure 6 is a restoration diagram of the specific classification results of the experimental results in Table 2. Figure 6 shows that LSBAENSVM has fewer classification errors than the other algorithms in the four sets of experiments. From Table 2, to specifically analyze the experimental results, it can be seen that LSBAENSVM achieved the best classification results on the Indian Pines dataset. Compared with the BAENSVM model, when each class uses 10%, 20%, 30%, and 40% of the training data, the OA of LSBAENSVM is 0.63%, 0.45%, 0.39%, and 0.29% higher than that of BAENSVM, respectively, while the Kappa coefficients are 1.34%, 1.17%, 0.62%, and 0.69% higher than those of BAENSVM, respectively. The classification accuracy of the BAENSVM model is better than those of SVM and TWSVM.

For example, using 10%, 20%, 30%, and 40% of the training data of, the OA of BAENSVM is 0.42%, 0.51%, 0.38%, and 0.42% higher than that of SVM, respectively, and the Kappa coefficients are 0.54%, 0.64%, 0.48%, and 0.48% higher than those of the SVM. This is because the BAENSVM algorithm is a nonparallel support vector machine constructed on the basis of an SVM. Under certain conditions, it can degenerate into an SVM, although the classification result must be better than that of an SVM. However, TWSVM only minimizes the empirical risk, which affects the classification accuracy, so the classification result is poor. While the LSSVM also achieves good classification results, LSBAENSVM has better classification accuracy. The OA is 0.1%, 0.15%, 0.17%, and 0.11% higher than that of SVM. Compared with the relationship between the BAENSVM and SVM, the LSBAENSVM is theoretically a nonparallel support vector machine based on an LSSVM but with better classification accuracy.

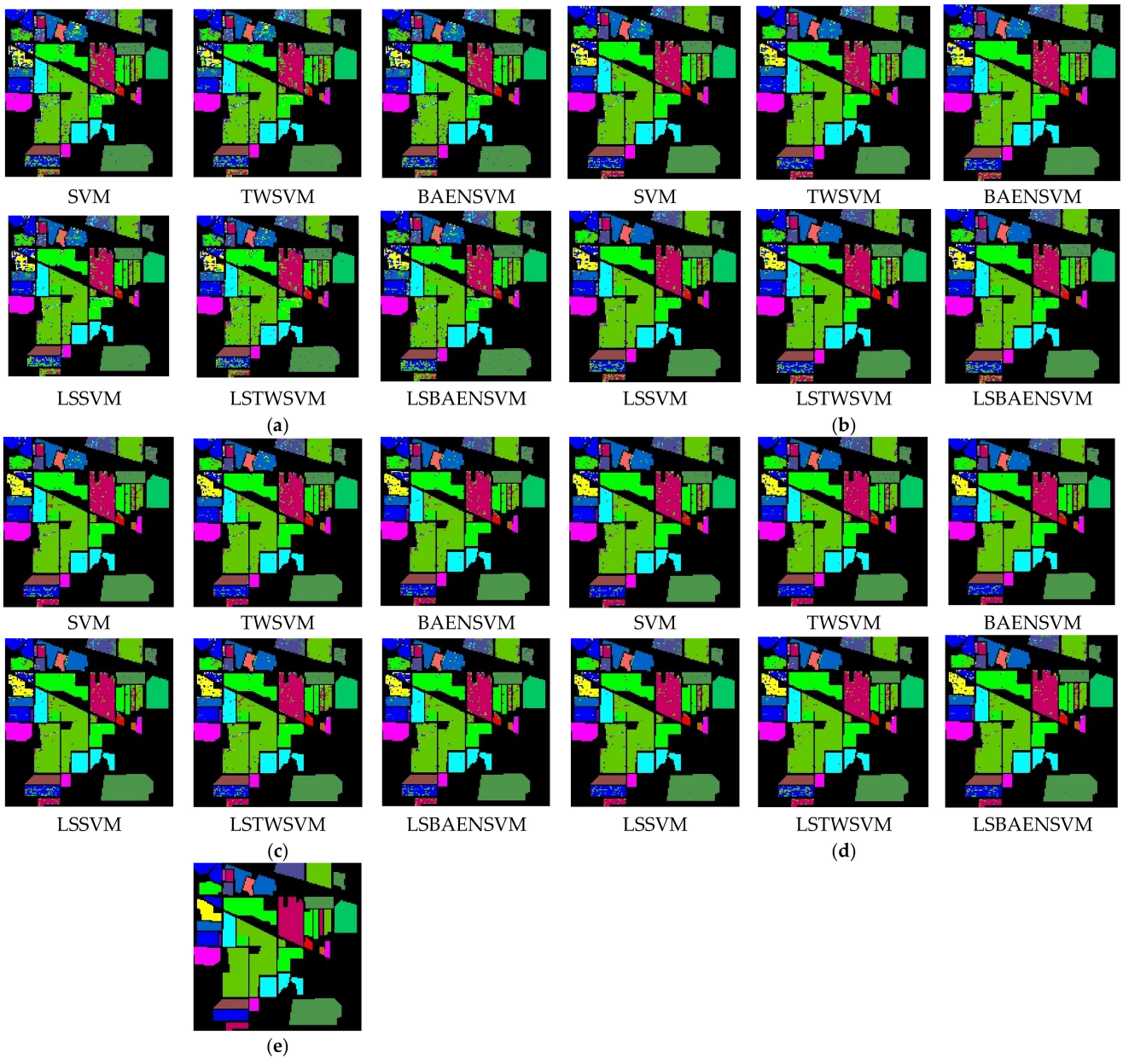


Figure 6. Indian Pines hyperspectral image classification result image. (**a**) 10% Indian Pines dataset, (**b**) 20% Indian Pines dataset, (**c**) 30% Indian Pines dataset, (**d**) 40% Indian Pines dataset, (**e**) ground truth.

3.2. Kennedy Space Center Dataset

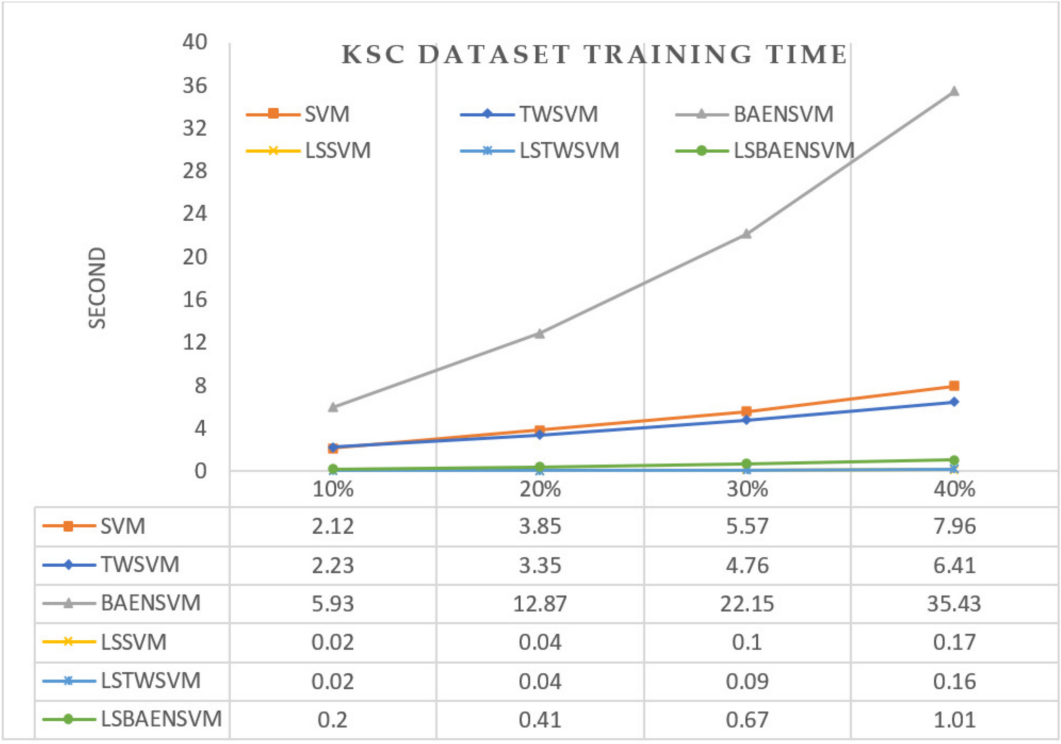
The Kennedy Space Center dataset has a small number of samples. For this dataset, 10%, 20%, 30%, and 40% of each category is selected as training data. The rest of the data are used as the test set. In this way, we can test the time requirement of each algorithm using datasets of different sizes and compare the classification accuracy.

The division of the Kennedy Space Center training set is shown in Table 3.

Table 3. The number of samples in each category of the Kennedy Space Center dataset and the division of training samples.

Class	Samples	10%	20%	30%	40%
Scrub	761	77	153	229	305
Willow swamp	243	25	49	73	98
CP hammock	256	26	52	77	103
Slash pine	252	26	51	76	102
Oak/Broadleaf	161	17	33	49	65
Hardwood	229	23	46	69	92
Swamp	105	11	21	32	42
Graminoid marsh	431	44	87	130	173
Spartina marsh	520	52	104	156	208
Cattail marsh	404	41	81	122	162
Salt marsh	419	42	84	126	168
Mud flats	503	51	101	151	202
Water	527	53	106	159	211
Total	4811	488	968	1449	1931

Figure 7 shows a solution time comparison of the SVM, TWSVM, BAENSVM, LSSVM, LSTWSVM and LSBAENSVM algorithms when using 10%, 20%, 30%, and 40% of the Kennedy Space Center training dataset. The indicated times are the average values of ten runs.



	10%	20%	30%	40%
SVM	2.12	3.85	5.57	7.96
TWSVM	2.23	3.35	4.76	6.41
BAENSVM	5.93	12.87	22.15	35.43
LSSVM	0.02	0.04	0.1	0.17
LSTWSVM	0.02	0.04	0.09	0.16
LSBAENSVM	0.2	0.41	0.67	1.01

Figure 7. Time-consumption comparison of SVM, TWSVM, BAENSVM, LSSVM, LSTWSVM, and LSBAENSVM using different scales of the Kennedy Space Center training dataset.

The six broken lines in Figure 7 show the solving speeds of the six models. It is obvious that the broken lines corresponding to the SVM, TWSVM, and BAENSVM models are above those corresponding to the LSSVM, LSTWSVM, and LSBAENSVM models. This is because the least square form applied in the former models greatly reduces the solving speed. Due to the increase in the data size, the training time of baensvm is increased by 2.8, 3.3, 4.0, and 4.45 times, respectively, compared to that of SVM. Additionally, it can be seen that the growth rate of the BAENSVM training times is more pronounced with an increase of data size. Compared with LSBAENSVM, the solution speed of LSBAENSVM is 29, 31, 33 and 35 times faster than that of BAENSVM; this is because the introduction of least squares converts the dual problem into an unconstrained convex quadratic programming problem, which greatly improves the solution speed.

Table 4 shows the experimental accuracies of the SVM, TWSVM, BAENSVM, LSSVM, LSTWSVM, and LSBAENSVM methods under training dataset scales of 10%, 20%, 30%, and 40% of the Kennedy Space Center datasets. The best experimental results are shown in bold.

Table 4. Classification results of Kennedy Space Center hyperspectral images.

Train Number	Accuracy	SVM	TWSVM	BAENSVM	LSSVM	LSTWSVM	LSBAENSVM
10%	OA	91.84	91.19	92.15	92.63	91.54	**92.86**
	Kappa	90.93	90.73	91.36	92.23	91.03	**92.40**
20%	OA	93.28	92.59	93.45	93.82	92.64	**94.03**
	Kappa	92.75	92.23	92.95	93.31	92.29	**93.46**
30%	OA	94.08	93.70	94.30	94.58	93.97	**94.76**
	Kappa	93.50	93.51	93.90	94.02	93.45	**94.24**
40%	OA	94.59	94.31	94.82	94.91	94.28	**95.17**
	Kappa	94.02	94.03	94.49	94.42	93.96	**94.74**

Bold in the table indicates the optimal accuracy.

Figure 8 sequentially shows the restoration graphs under the classification of SVM, TWSVM, BAENSVM, LSSVM, LSTWSVM, and LSBAENSVM, corresponding to the detailed classification results in Table 4.

Table 4 shows the experimental accuracies of six classification models. Figure 8 is a restoration diagram of the specific classification results of the experimental results in Table 4. Figure 8 shows that the LSBAENSVM has fewer classification errors than the other algorithms in the four sets of experiments. From Table 4, it can be seen that LSBAENSVM achieves the best classification results using the Kennedy Space Cente dataset. Compared with the BAENSVM model, when each class takes 10%, 20%, 30%, and 40% of the training data, the OA of LSBAENSVM is 0.71%, 0.58%, 0.46%, and 0.35% higher than that of BAENSVM, respectively, while the Kappa coefficients are 1.04%, 0.51%, 0.34%, and 0.25% higher than those of BAENSVM. The classification accuracy of the BAENSVM model is better than that of the SVM. For example, in the training data of 10%, 20%, 30% and 40%, the OA of BAENSVM is 0.31%, 0.17%, 0.22% and 0.23% higher than those of SVM, respectively, and the Kappa coefficients are 0.43%, 0.20%, 0.40% and 0.47% higher than those of the SVM. This is because the BAENSVM algorithm is a nonparallel support vector machine based on an SVM. As such, the classification accuracy of TWSVM is 0.96%, 0.86%, 0.60%, and 0.51% lower than that of BAENSVM. This is because baensvm minimizes the structural risk compared with TWSVM, and thus, has stronger generalization ability. While the LSSVM also achieves good classification results, LSBAENSVM has better classification accuracy than LSSVM. The OA is 0.23%, 0.21%, 0.18% and 0.26% higher than those of the SVM. Compared with the relationship between the BAENSVM and SVM, the LSBAENSVM is theoretically a nonparallel support vector machine based on an LSSVM but with better classification accuracy. As long as control penalty parameters c_1 and c_3 are small enough, the classification accuracy of lsbaensvm is similar to that of LSSVM.

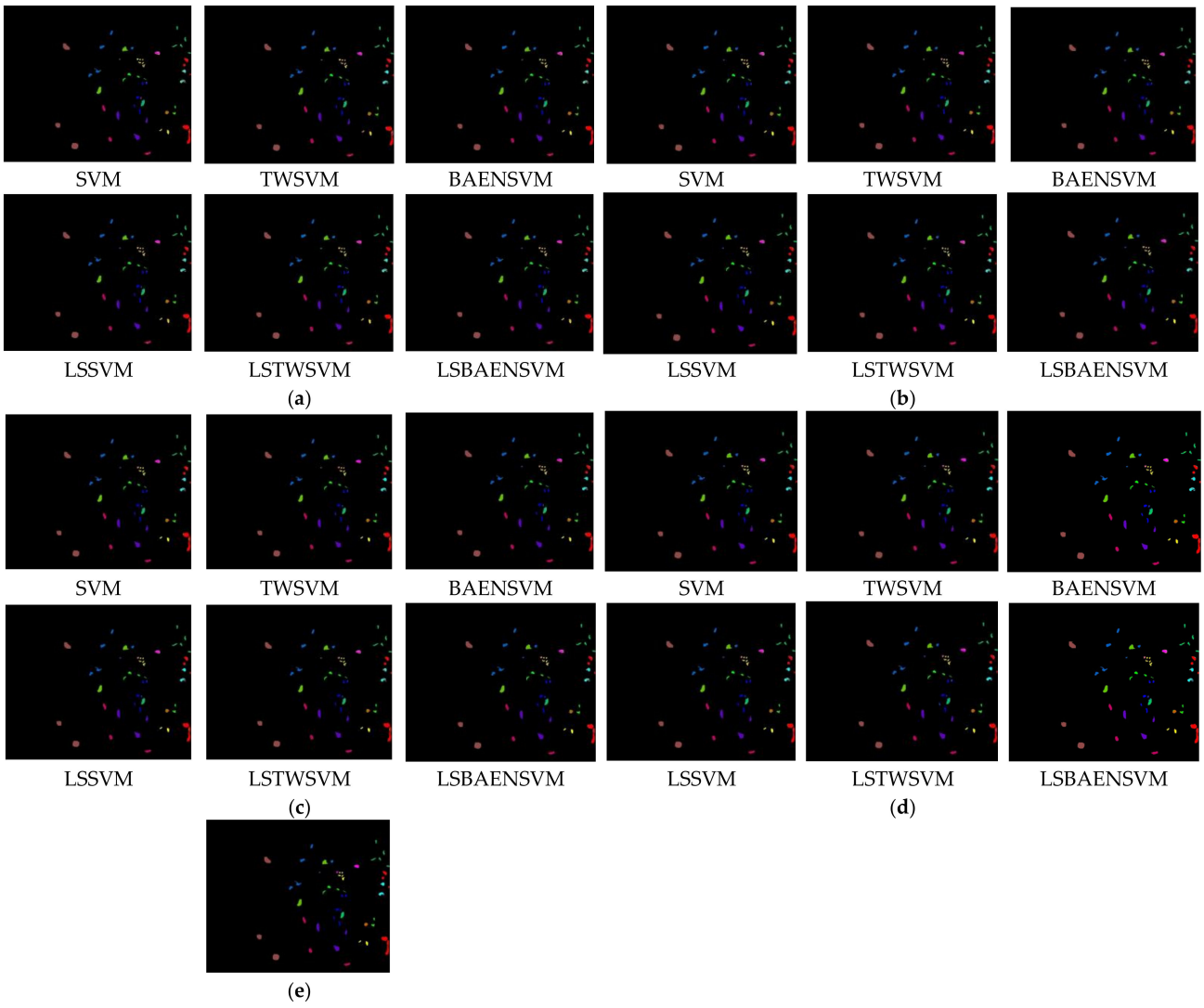


Figure 8. Kennedy Space Center hyperspectral image classification result image. (**a**) 10% KSC dataset, (**b**) 20% KSC dataset, (**c**) 30% KSC dataset, (**d**) 40% KSC dataset, (**e**) ground truth.

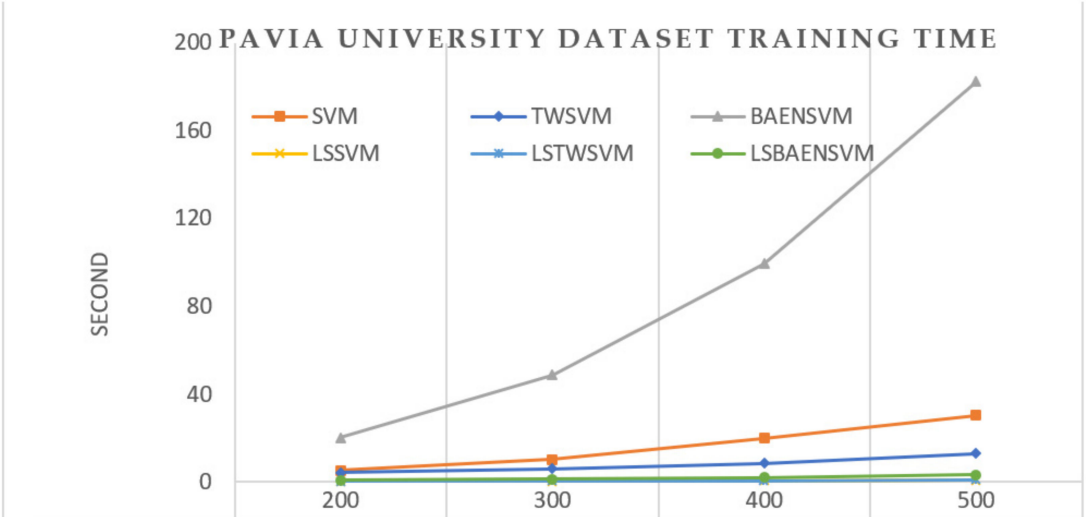
3.3. Pavia University Dataset

The Pavia University dataset contains a large amount of data, and each category selects the same number of samples for training. For each category, 200, 300, 400, and 500 data points were selected as training samples to conduct four sets of experiments in order to test the classification results under different scales of training data. The specific classifications are shown in Table 5.

Table 5. Ground truth classes for the Pavia University data and their respective numbers of samples.

Class	Samples	Train Dataset 1	Train Dataset 2	Train Dataset 3	Train Dataset 4
Asphalt	6631	200	300	400	500
Meadows	18,649	200	300	400	500
Gravel	2099	200	300	400	500
Trees	3064	200	300	400	500
Painted metal sheets	1345	200	300	400	500
Bare Soil	5029	200	300	400	500
Bitumen	1330	200	300	400	500
Self-Blocking Bricks	3682	200	300	400	500
Shadows	947	200	300	400	500
Total	42,776	1800	2700	3600	4500

Figure 9 shows a solution time comparison of the SVM, TWSVM, BAENSVM, LSSVM, LSTWSVM, and LSBAENSVM algorithms under the four Pavia University experiments. The indicated times are the averages of ten runs.



	200	300	400	500
SVM	5.29	10.17	19.77	30.27
TWSVM	4.06	5.96	8.37	12.75
BAENSVM	20.15	48.75	99.2	182.32
LSSVM	0.11	0.23	0.44	0.72
LSTWSVM	0.11	0.23	0.41	0.75
LSBAENSVM	0.64	1.21	2.07	3.3

Figure 9. Time-consumption comparison of SVM, TWSVM, BAENSVM, LSSVM, LSTWSVM, and LSBAENSVM using different scales of Pavia University training datasets.

As seen in Figure 9, when the number of training samples in each category is equal, the line graph of the TWSVM time-consumption calculation is completely under that of the support vector machine. This shows that TWSVM is less computationally expensive because it solves two matrix problems which are of smaller scales than those of the SVM. The BAENSVM time-consumption calculation line chart is far above other those of the classification algorithms. This is because this method needs to solve two convex quadratic programming problems, and the solution matrix size of each optimization problem is larger than that of an SVM. Therefore, the computational complexity of the BAENSVM algorithm is much greater than those of the other algorithms. In this experiment, when the training datasets of each category are 200, 300, 400 and 500 data points, the solution times of BAENSVM are 3.8, 4.8, 5.0 and 6.0 times those of the SVM, respectively. Therefore, with an increase in data size, the solution time of the BAENSVM increases gradually compared to the SVM. Observing the line graph, the calculation timelines corresponding to the three algorithms, i.e., LSSVM, LSTWSVM and LSBAENSVM, are far lower than those of SVM, TWSVM, and BAENSVM, indicating that the least squares form can significantly reduce the complexity of the calculations. It can be seen from the figure that when the training datasets of each category are 200, 300, 400, and 500 data points, the solution times of BAENSVM are 31.5, 40.3, 47.9, and 55.2 times those of the LSBAENSVM, respectively. As the size of the training set increases, the computational speed advantage of LSBAENSVM over BAENSVM increases.

Table 6 shows the experimental accuracies of the SVM, TWSVM, BAENSVM, LSSVM, LSTWSVM, and LSBAENSVM methods when using the Pavia University dataset. The best experimental results are shown in bold.

Table 6. Classification results of Pavia University hyperspectral images.

Train Number	Accuracy	SVM	TWSVM	BAENSVM	LSSVM	LSTWSVM	LSBAENSVM
200	OA	91.35	91.68	91.87	91.82	91.46	**92.50**
	Kappa	88.50	89.35	89.34	89.40	88.88	**90.20**
300	OA	91.75	91.95	92.36	92.20	92.14	**92.81**
	Kappa	88.94	88.58	89.87	89.88	89.64	**90.55**
400	OA	92.57	91.89	92.73	92.60	92.71	**93.12**
	Kappa	89.93	89.29	90.27	90.20	90.41	**90.78**
500	OA	92.83	92.32	93.13	92.89	93.07	**93.42**
	Kappa	90.18	89.74	90.72	90.49	90.74	**91.07**

Bold in the table indicates the optimal accuracy.

Figure 10 sequentially shows the restoration graphs under the classification of SVM, TWSVM, BAENSVM, LSSVM, LSTWSVM, and LSBAENSVM, corresponding to the classification results presented in Table 6.

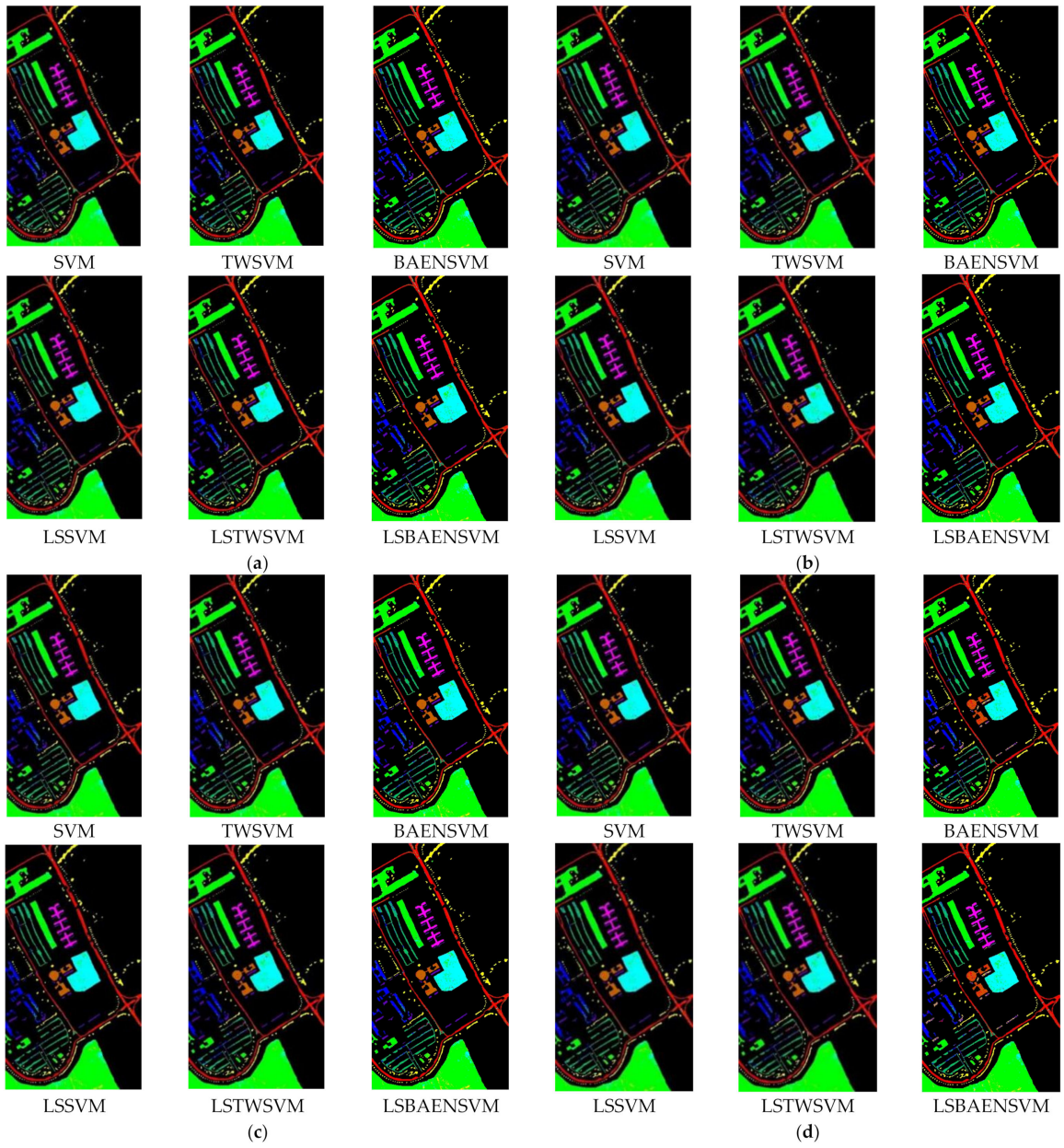


Figure 10. *Cont.*

(**e**)

Figure 10. Pavia University hyperspectral image classification result image. (**a**) Training Dataset 1, (**b**) training Dataset 2, (**c**) training Dataset 3, (**d**) training Dataset 4, (**e**) ground truth.

Table 6 shows the experimental accuracies of the six classification models. Figure 10 shows that LSBAENSVM has fewer classification errors than the other algorithms in the four sets of experiments. From Table 6, it can be seen that LSBAENSVM achieves the best classification results on the Pavia University dataset. Compared with the BAENSVM model, the Oas of LSBAENSVM are 0.63%, 0.45%, 0.39%, and 0.29% higher than those of BAENSVM when 200, 300, 400, and 500 training data are taken for each class, respectively, and the Kappa coefficients are 0.86%, 0.68%, 0.51%, and 0.35% higher than those of BAENSVM. Compared with LSSVM, the Oas of LSBAENSVM are 0.68%, 0.61%, 0.52%, and 0.56% higher than those of LSSVM, respectively, while the Kappa coefficients are 0.80%, 0.67%, 0.58%, and 0.65% higher than those of BAENSVM. Similar to the relationship between the LSBAENSVM and BAENSVM, the LSBAENSVM is essentially equivalent to the nonparallel support vector machine algorithm obtained by adding the least squares of positive and negative samples to the LSSVM algorithm model. Therefore, theoretically, LSBAENSVM can obtain better accuracy than LSSVM.

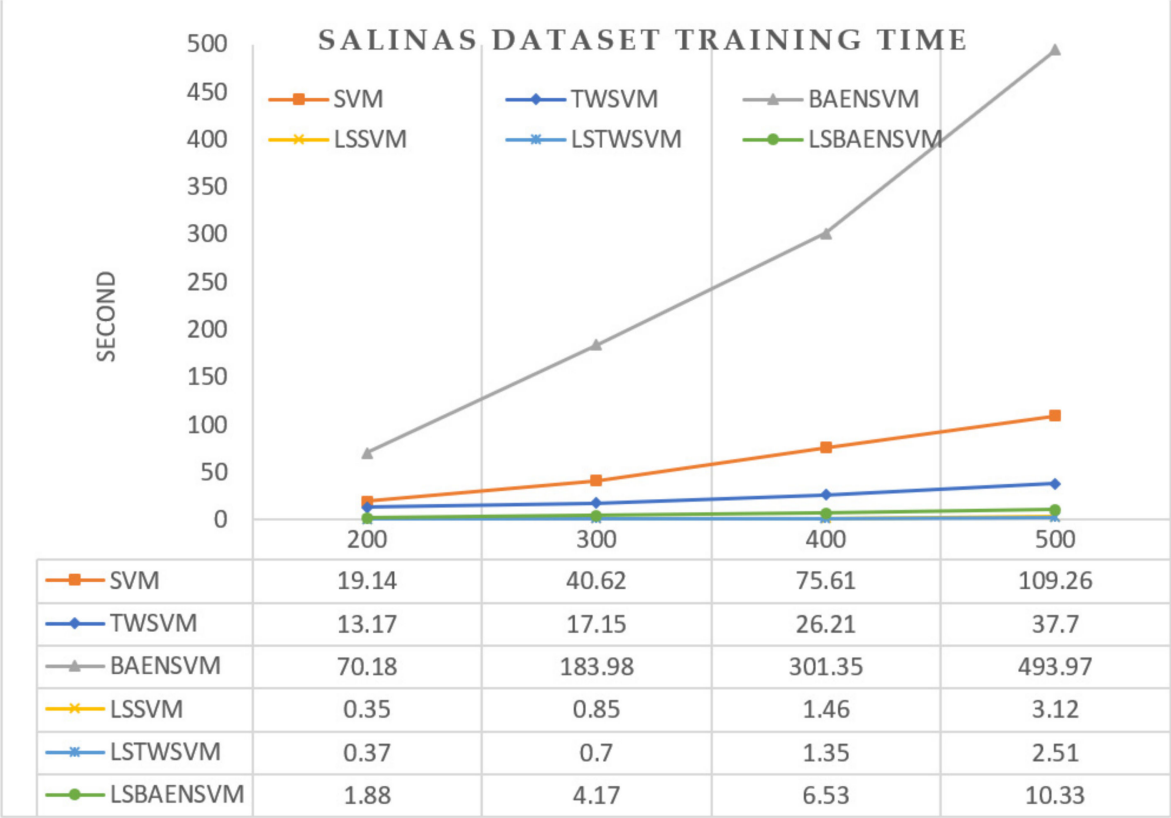
3.4. Salinas Dataset

The Salinas dataset contains a large amount of data, and each category selects the same number of samples for training. For each category, 200, 300, 400, and 500 data points are selected as training samples to conduct four sets of experiments in order to test the classification results under different scales of training data. The specific classifications are shown in Table 7.

Table 7. Ground truth classes for the Salinas scene and their respective numbers of samples.

Class	Samples	Train Dataset 1	Train Dataset 2	Train Dataset 3	Train Dataset 4
ineyard_green_weeds_1	2009	200	300	400	500
ineyard_green_weeds_2	3726	200	300	400	500
Fallow	1976	200	300	400	500
Fallow_rough_plow	1394	200	300	400	500
Fallow_smooth	2678	200	300	400	500
Stubble	3959	200	300	400	500
Celery	3579	200	300	400	500
Grapes_untrained	11,271	200	300	400	500
Soil_vinyard_develop	6203	200	300	400	500
Corn_senesced_green_weeds	3278	200	300	400	500
Lettuce_romaine_4wk	1068	200	300	400	500
Lettuce_romaine_5wk	1927	200	300	400	500
Lettuce_romaine_6wk	916	200	300	400	500
Lettuce_romaine_7wk	1070	200	300	400	500
Vinyard_untrained	7268	200	300	400	500
Vinyard_vertical_trellis	1807	200	300	400	500
Total	54,129	1800	2700	3600	4500

Figure 11 shows a solution time comparison of the SVM, TWSVM, BAENSVM, LSSVM, LSTWSVM, and LSBAENSVM algorithms under the four groups of Salinas experiments. The selected times are the average of ten runs.



	200	300	400	500
SVM	19.14	40.62	75.61	109.26
TWSVM	13.17	17.15	26.21	37.7
BAENSVM	70.18	183.98	301.35	493.97
LSSVM	0.35	0.85	1.46	3.12
LSTWSVM	0.37	0.7	1.35	2.51
LSBAENSVM	1.88	4.17	6.53	10.33

Figure 11. Time-consumption comparison of SVM, TWSVM, BAENSVM, LSSVM, LSTWSVM, and LSBAENSVM in different scales of the Salinas training dataset.

Figure 11 shows a solution time consumption of SVM, TWSVM, BAENSVM, LSSVM, LSTWSVM, and LSBAENSVM based on the four groups of experiments. When the value of each category is the same as that of Pavia dataset, the classification category is increased to 16 categories, which makes finding the solution more difficult. When the training datasets of each category comprise 200, 300, 400 and 500 data points, the solution time of BAENSVM is 70, 184, 301, and 494, respectively. Under these circumstances, model training becomes more difficult. SVM and TWSVM have faster solution speeds than BAENSVM because of their small data size, but they also require more time. The introduction of the least square idea greatly improves the speed of the algorithm. It is obvious that the time requirements of LSSVM, LSTWSVM, and LSBAENSVM are far from those of SVM, TWSVM, and BAENSVM. The solution time of LSBAENSVM is 37, 44, 46, and 48 times faster than that of BAENSVM, which further verifies the efficacy of the algorithm with the least square approach.

Table 8 shows the experimental accuracy of the SVM, TWSVM, BAENSVM, LSSVM, LSTWSVM, and LSBAENSVM methods when using the Salinas dataset. The best experimental results are shown in bold.

Table 8. Classification results of Salinas hyperspectral images.

Train Number	Accuracy	SVM	TWSVM	BAENSVM	LSSVM	LSTWSVM	LSBAENSVM
200	OA	91.13	91.08	91.37	91.34	91.01	**91.69**
	Kappa	90.19	90.12	90.39	90.35	89.97	**90.75**
300	OA	92.01	91.98	92.40	92.22	92.13	**92.57**
	Kappa	91.14	91.06	91.51	91.30	91.29	**91.69**
400	OA	92.43	92.30	92.55	92.58	92.25	**92.87**
	Kappa	91.52	91.37	91.63	91.71	91.32	**92.01**
500	OA	92.50	92.44	92.64	92.72	92.21	**92.92**
	Kappa	91.57	91.52	91.67	91.77	91.25	**92.03**

Bold in the table indicates the optimal accuracy.

Figure 12 sequentially shows the restoration graphs under the classification of SVM, TWSVM, BAENSVM, LSSVM, LSTWSVM, and LSBAENSVM, corresponding to the classification results presented in Table 8.

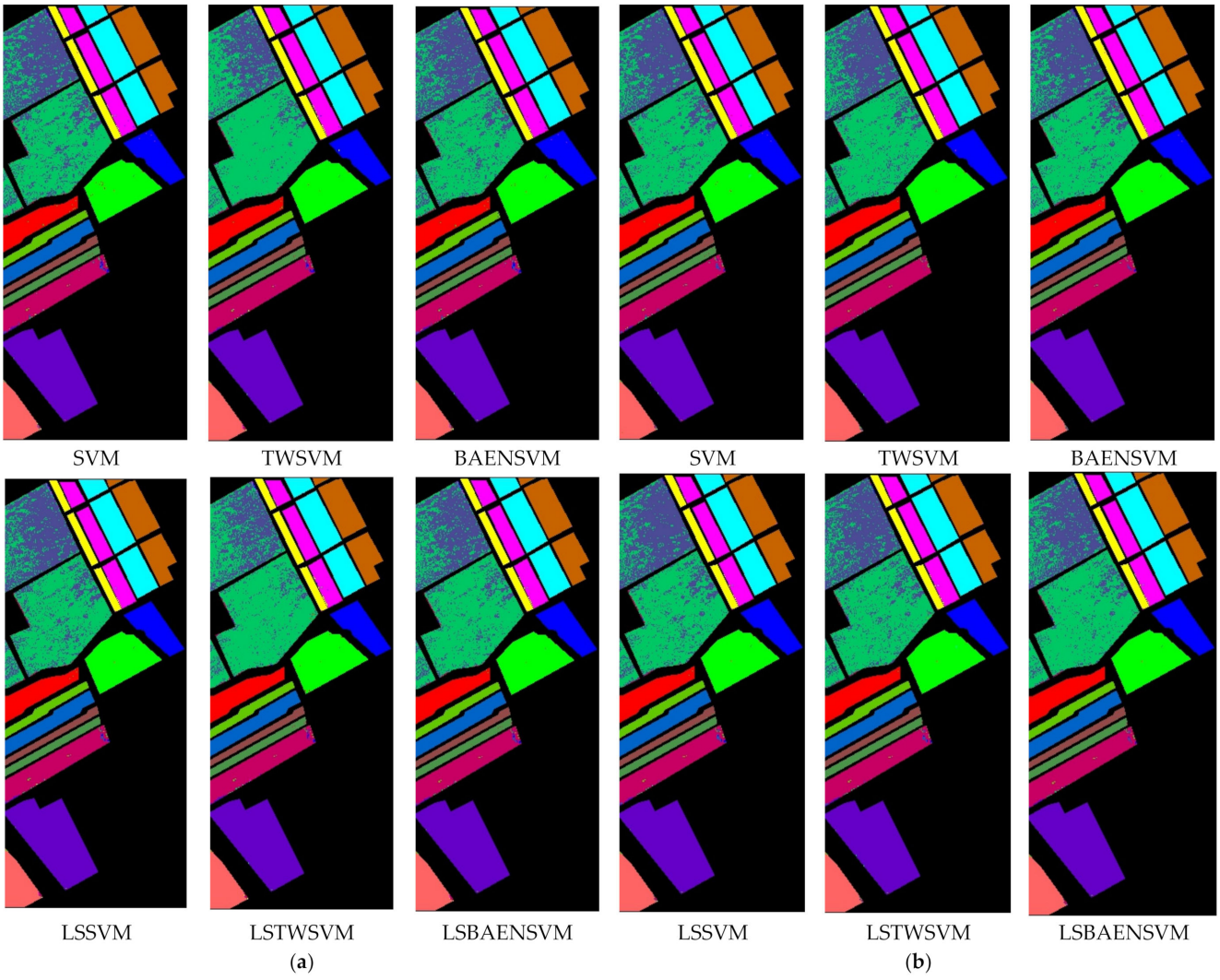


Figure 12. *Cont.*

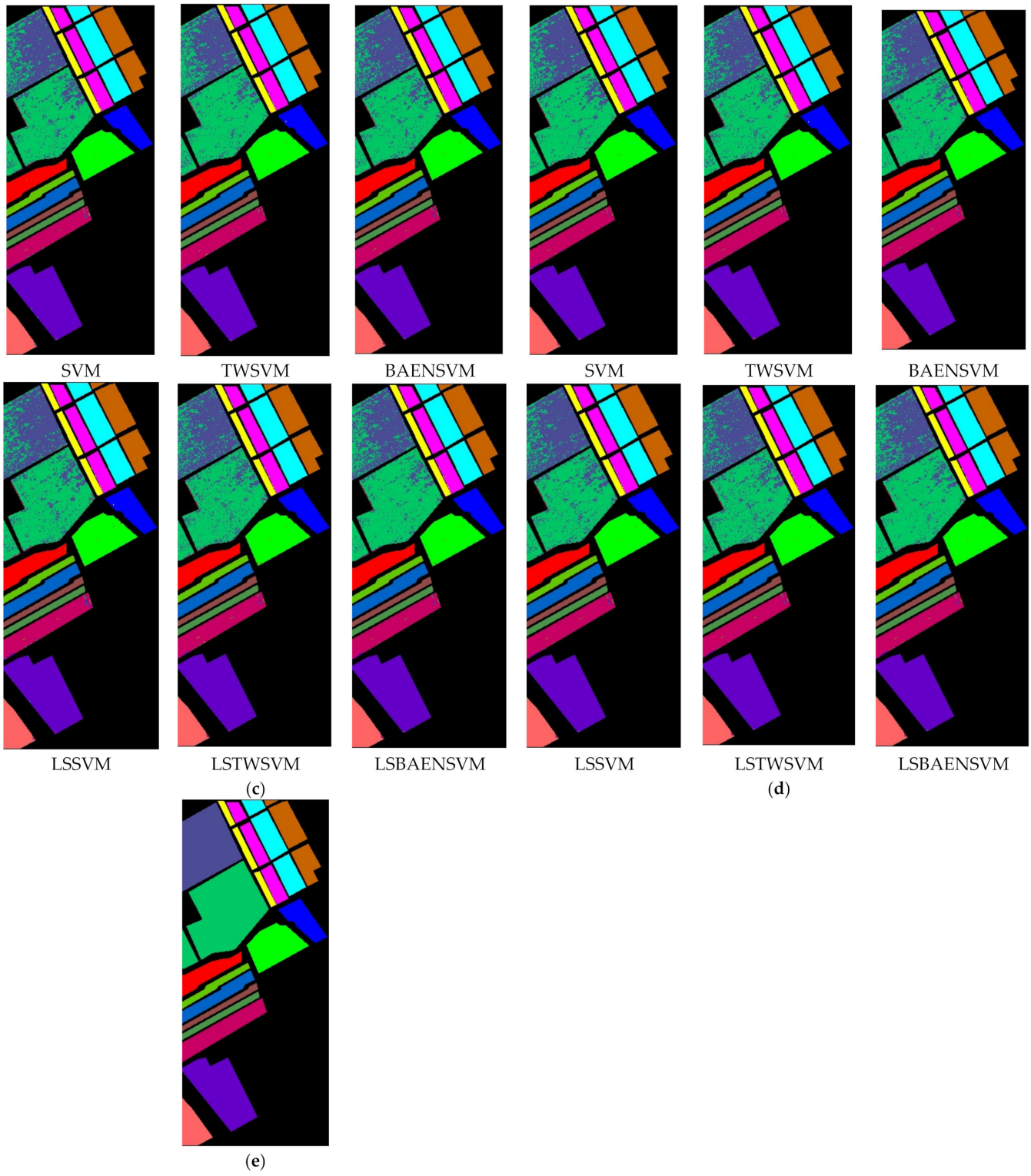


Figure 12. Salinas hyperspectral image classification result images. (**a**) Training Dataset 1, (**b**) training Dataset 2, (**c**) training Dataset 3, (**d**) training Dataset 4, (**e**) ground truth.

Table 8 shows the experimental accuracies of the six classification models. Figure 12 is a restoration diagram of the specific classification results of the experimental results in Table 8. Figure 12 shows that the LSBAENSVM has fewer classification errors than the other algorithms in the four sets of experiments. From Table 8, it can be seen that LSBAENSVM achieves the best classification results on the Salinas dataset. Compared with

the BAENSVM model, the OAs of LSBAENSVM are 0.56%, 0.56%, 0.44%, and 0.42% higher than those of BAENSVM when 200, 300, 400, and 500 training data are taken for each class, respectively, and the Kappa coefficients are 0.56%, 0.55%, 0.49%, and 0.46% higher than those of BAENSVM. Compared with LSSVM, the OAs of LSBAENSVM are 0.35%, 0.35%, 0.29%, and 0.20% higher than those of LSSVM, and the Kappa coefficients are 0.40%, 0.39%, 0.30%, and 0.26% higher than those of BAENSVM. Because LSBAENSVM is equivalent to the improvement of a nonparallel support vector machine based on LSSVM, it invariably obtains classification accuracies which are superior to those of LSSVM.

4. Conclusions

As a classic classification algorithm, the support vector machine achieves good results in the classification of hyperspectral images. Aiming to overcome the problem whereby the parallel discriminant plane of an SVM cannot meet the real distribution trend of hyperspectral data, the BAENSVM features two nonparallel decision planes based on a support vector machine. This modification provides better results in terms of classification accuracy. However, the BAENSVM also has certain problems: larger-scale solving matrices require increased training times, and large-scale data may be difficult to solve. The algorithm model proposed in this paper introduces the idea of least squares on the basis of the BAENSVM, giving rise to the LSBAENSVM model. Compared with the BAENSVM, it has offers huge improvements in terms of training speed, giving rise to the possibility for larger-scale data training. Additionally, it exhibits improved experimental accuracy.

The algorithm in this paper has many parameters that need to be manually adjusted, which can be difficult during training. Here are some tips to reduce the complexity of parameter adjustment. First, set $c_1 = c_2$ and $c_3 = c_4$, and use the grid search method to optimize the two parameters at a certain interval. After finding the parameters which yield a high degree of accuracy, keep $c_1 = c_2$ unchanged and readjust c_3 and c_4 to obtain better values. Finally, find the optimal values for c_1 and c_2 in the same way. Parameters c_3 and c_4 have the same function as C in LSSVM. As such, if $c_3 = c_4$ is fixed and they take a small value, and $c_3 = c_4$ is adjusted at the same time, a pair of parallel classification decision planes, just like with the LSSVM, will be obtained. c_1 and c_2 represent the degree of offset on the parallel decision hyperplane. Adjusting their size yields a decision hyperplane that is more consistent with the data distribution. Such a parameter selection method ensures that the classification accuracy of lsbaensvm is not inferior to that of LSSVM, even if the optimal lsbaensvm parameter is not applied.

The algorithm model proposed in this paper only improves upon the BAENSVM model. Similar to the latter model, there is a gap in accuracy compared to current popular deep learning classification methods. The main reason for this is that the classification of the model only uses spectral information and does not make in-depth use of hyperspectral data features. The main purpose of this paper is to improve the training time problem of the BAENSVM model. In future, based on the LSBAENSVM classification model, we will deep-mine the hyperspectral data information, combining spatial and spectral information to further improve the classification accuracy. At present, many classification methods exist which combine spatial and spectral information. For example, by extracting spatial texture information to combine spatial spectral features, a Gabor filter can be applied to process spatial information, to extract texture information features at multiple scales and directions, and finally, to combine spectral feature vectors to form fusion features which can then be classified using the lsbaensvm model. Alternatively, image LBP features can be used to characterize spatial texture information and fuse it with spectral features. In such a setting, fusion features and the lsbaensvm algorithm can be used for classifications.

Author Contributions: Conceptualization, L.W.; Methodology and formal analysis, G.L.; Writing and Valuable advice, D.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China (grant number 62071084, 62001434), and by the Leading Talents Project of the State Ethnic Affairs Commission.

Data Availability Statement: All dataset can be obtained at http://www.ehu.eus/ccwintco/index.php?title=Hyperspectral_Remote_Sensing_Scenes (accessed on 20 May 2011).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Sowmya, V.; Soman, K.P.; Hassaballah, M. Hyperspectral image: Fundamentals and advances. In *Recent Advances in Computer Vision*; Springer: Berlin/Heidelberg, Germany, 2019; pp. 401–424.
2. Lv, W.; Wang, X. Overview of Hyperspectral Image Classification. *J. Sens.* **2020**, *2020*, 4817234. [CrossRef]
3. Ranjan, S.; Nayak, D.R.; Kumar, K.S.; Dash, R.; Majhi, B. Hyperspectral Image Classification: A k-Means Clustering Based Approach. In Proceedings of the 2017 4th International Conference on Advanced Computing and Communication Systems (ICACCS), Coimbatore, India, 6–7 January 2017.
4. El Rahman, S.A. Hyperspectral imaging classification using ISODATA algorithm: Big data challenge. In Proceedings of the 2015 Fifth International Conference on e-Learning (econf), Manama, Bahrain, 18–20 October 2015; pp. 247–250.
5. Song, W.; Li, S.; Kang, X.; Huang, K. Hyperspectral image classification based on KNN sparse representation. In Proceedings of the 2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), Beijing, China, 10–15 July 2016; pp. 2411–2414. [CrossRef]
6. Goel, P.K.; Prasher, S.O.; Patel, R.M.; Landry, J.A.; Bonnell, R.B.; Viau, A.A. Classification of hyperspectral data by decision trees and artificial neural networks to identify weed stress and nitrogen status of corn. *Comput. Electron. Agric.* **2003**, *39*, 67–93.
7. Pradhan, B. A comparative study on the predictive ability of the decision tree, support vector machine and neuro-fuzzy models in landslide susceptibility mapping using GIS. *Comput. Geosci.* **2013**, *51*, 350–365.
8. Belgiu, M.; Drăguţ, L. Random forest in remote sensing: A review of applications and future directions. *ISPRS J. Photogramm. Remote Sens.* **2016**, *114*, 24–31. [CrossRef]
9. Tan, K.; Du, P. Hyperspectral Remote Sensing Image Classification Based on Support Vector Machine. *J. Infrared Millim. Waves* **2008**, *27*, 123–128. [CrossRef]
10. Wang, Y. Remote Sensing Image Automatic Classification with Support Vector Machine. *Comput. Simul.* **2013**, *30*, 378–385.
11. Pal, M.; Mather, P.M. Support vector machines for classification in remote sensing. *Int. J. Remote Sens.* **2005**, *26*, 1007–1011. [CrossRef]
12. Cortes, C.; Vapnik, V. Support-vector networks. *Mach. Learn.* **1995**, *20*, 273–297. [CrossRef]
13. Zhang, D.; Zhou, Z.H.; Chen, S. Semi-supervised dimensionality reduction. In Proceedings of the 2007 SIAM International Conference on Data Mining, Society for Industrial and Applied Mathematics, Minneapolis, MN, USA, 26–28 April 2007; pp. 629–634.
14. Sain, S.R. *The Nature of Statistical Learning Theory*; Taylor & Francis: Abingdon, UK, 1996; p. 409.
15. Harikiran, J.J. Hyperspectral image classification using support vector machines. *IAES Int. J. Artif. Intell.* **2020**, *9*, 684. [CrossRef]
16. Melgani, F.; Bruzzone, L. Classification of hyperspectral remote sensing images with support vector machines. *IEEE Trans. Geosci. Remote Sens.* **2004**, *42*, 1778–1790. [CrossRef]
17. Chan, R.H.; Kan, K.K.; Nikolova, M.; Plemmons, R.J. A two-stage method for spectral–spatial classification of hyperspectral images. *J. Math. Imaging Vis.* **2020**, *62*, 790–807. [CrossRef]
18. Jin, S.; Zhang, W.; Yang, P.; Zheng, Y.; An, J.; Zhang, Z.; Qu, P.; Pan, X. Spatial-spectral feature extraction of hyperspectral images for wheat seed identification. *Comput. Electr. Eng.* **2022**, *101*, 108077. [CrossRef]
19. Suykens, J.A.; Vandewalle, J. Least squares support vector machine classifiers. *Neural Processing Lett.* **1999**, *9*, 293–300. [CrossRef]
20. Gao, H.-Z.; Wan, J.-W.; Zhu, Z.-Z.; Wang, L.-B.; Nian, Y.-J. Classification technique for hyperspectral image based on subspace of bands feature extraction and LS-SVM. *Spectrosc. Spectr. Anal.* **2011**, *31*, 1314–1317.
21. Shao, Y.; Gao, C.; Xuan, G.; Gao, X.; Chen, Y.; Hu, Z. Determination of damaged wheat kernels with hyperspectral imaging analysis. *Int. J. Agric. Biol. Eng.* **2020**, *13*, 194–198. [CrossRef]
22. Mangasarian, O.L.; Wild, E.W. Multisurface proximal support vector machine classification via generalized eigenvalues. *IEEE Trans. Pattern Anal. Mach. Intell.* **2005**, *28*, 69–74. [CrossRef] [PubMed]
23. Khemchandani, R.; Chandra, S. Twin support vector machines for pattern classification. *IEEE Trans. Pattern Anal. Mach. Intell.* **2007**, *29*, 905–910.
24. Schölkopf, B.; Smola, A.J.; Bach, F. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*; MIT Press: Cambridge, MA, USA, 2002.
25. Kaya, G.T.; Torun, Y.; Küçük, C. Recursive feature selection based on non-parallel SVMs and its application to hyperspectral image classification. In Proceedings of the 2014 IEEE Geoscience and Remote Sensing Symposium, Quebec City, QC, Canada, 13–18 July 2014; pp. 3558–3561.
26. Liu, Z.; Zhu, L. A novel remote sensing image classification algorithm based on multi-feature optimization and TWSVM. In Proceedings of the Ninth International Conference on Digital Image Processing (ICDIP 2017), International Society for Optics and Photonics, Hong Kong, China, 19–22 May 2017.
27. Liu, G.; Wang, L.; Liu, D.; Fei, L.; Yang, J. Hyperspectral Image Classification Based on Non-Parallel Support Vector Machine. *Remote Sens.* **2022**, *14*, 2447. [CrossRef]