electronics

MDPI

Article

DQKNet: Deep Quasiconformal Kernel Network Learning for Image Classification

Jia Zhai [1,2,*], Zikai Zhang [2], Fan Ye [2], Ziquan Wang [2] and Dan Guo [2]

[1] Communication University of China, Beijing 100024, China
[2] National Key Laboratory of Scattering and Radiation, Beijing 100854, China; zzk@nwpu.edu.cn (Z.Z.); yefan@bit.edu.cn (F.Y.); wang-zq23@mails.tsinghua.edu.cn (Z.W.); 20080717@stu.neu.edu.cn (D.G.)
* Correspondence: 173310080904003@cuc.edu.cn or zhaiji_bj@163.com

Abstract: Compared to traditional technology, image classification technology possesses a superior capability for quantitative analysis of the target and background, and holds significant applications in the domains of ground target reconnaissance, marine environment monitoring, and emergency response to sudden natural disasters, among others. Currently, the enhancement of spatial spectral resolution heightens the difficulty and reduces the efficiency of classification, posing a substantial challenge to the aforementioned applications. Hence, the classification algorithm is required to take both computing power and classification accuracy into account. Research indicates that the deep kernel mapping network can accommodate both computing power and classification accuracy. By employing the kernel mapping function as the network node function of deep learning, it effectively enhances the classification accuracy under the condition of limited computing power. Therefore, to address the issue of network structure optimization of deep mapping networks and the insufficient application of line feature learning and expression in existing network structures, considering the adaptive optimization of network structures, deep quasiconformal kernel network learning (DQKNet) is proposed for image classification. Firstly, the structural parameters and learning parameters of the deep kernel mapping network are optimized. This approach can adaptively adjust the network structure based on the distribution characteristics of the data and enhance the performance of image classification. Secondly, the computational network node optimization method of quasiconformal kernel learning is applied to this network, further elevating the performance of the deep kernel learning mapping network in image classification. The experimental results demonstrate that the improvement in the deep kernel mapping network from the perspectives of accounting children, mapping network nodes, and network structure can effectively enhance the feature extraction and classification performance of the data. On the five public datasets, the average AA, OA, and KC values of our algorithm are 91.99, 91.25, and 85.99, respectively, outperforming the currently most-advanced algorithms.

Keywords: image classification; deep quasiconformal kernel network learning; feature extraction

Citation: Zhai, J.; Zhang, Z.; Ye, F.; Wang, Z.; Guo, D. DQKNet: Deep Quasiconformal Kernel Network Learning for Image Classification. *Electronics* **2024**, *13*, 4168. https://doi.org/10.3390/electronics13214168

Academic Editor: Beiwen Li

Received: 29 August 2024
Revised: 18 September 2024
Accepted: 23 September 2024
Published: 24 October 2024

1. Introduction

The concept of deep kernel learning was first introduced by Yu et al. in 2008. Its goal is to enhance visual recognition speed and accuracy through the use of kernel regularization techniques in neural networks. By employing stochastic gradient descent, the method significantly improves both the efficiency and accuracy of visual recognition, laying the foundation for subsequent research in deep kernel learning [1]. This approach addresses several limitations of traditional kernel learning, such as poor performance when dealing with complex problems due to a reliance on limited sample-driven learning. Furthermore, kernel learning machines are often highly sensitive to the choice of the kernel function, lacking the ability to adapt to input data. The principles of deep learning have further advanced the development of kernel learning.

In 2012, Cho et al. proposed MKMS, based on a deep belief network, which described the transition from a support vector machine to a deep multilayer structure in a more complete and comprehensive way [2]. Brahma et al. introduced kernel-based deep neural networks from the perspective of popular learning and presented a relatively complete theoretical framework for deep kernel learning [3]. In 2011, researchers enhanced the Fisher discriminant ability to improve the separability of different sample categories [4]. Several algorithms were introduced, such as FKNN and DLF, to effectively improve image classification [5]. To address clustering challenges, Prenger proposed the RBF kernel-based K-means algorithm, which combines nonlinear kernels with deep architectures [6]. Meanwhile, Harikrishnan et al. applied kernel K-means to satellite images, using deep learning to achieve nonlinear clustering of a pseudo-training set of pixels representing changed and invariant fields [7]. To tackle recursion problems, Varma et al. developed a local deep kernel learning algorithm to accelerate the prediction performance of nonlinear support vector machines and optimize the spatial tree structure [8]. Finally, to address deep model selection, Strobl et al. proposed a multikernel deep learning approach, which enhances the performance of deep kernel learning through multilayer and multikernel function learning [9].

Regarding image classification technology, it holds significant application value in various domains, such as ground target reconnaissance, marine environment monitoring, and emergency response, among others. Hui Bi et al. proposed a novel target reconnaissance and classification framework based on sparse SAR images to automatically extract features and train models for ground target reconnaissance involving complex image data [10]. Zeyuan Shao et al. put forward an efficient model for detecting small objects in the marine environment to accomplish ocean monitoring [11]. Additionally, Liron Bergman et al. employed a classification-based approach to respond to sudden abnormal events [12], and Haoran Ye et al. also adopted this idea to detect the fault of the oceanographic multi-layer winch fiber rope arrangement [13]. Given the unique challenges of image classification, researchers have proposed targeted approaches to network structure optimization, resulting in various deep learning-based classification methods [14–17]. These include HSI spatial classification [18] and spectral–spatial residual networks (SSRNs) [19], which extract discriminative features from spatial context and rich spectral data. A 3D CNN has also been developed to retain deep spectral features, effectively differentiating them. In [20], a deep belief network (DBN) was introduced to improve classification, while in [21], a fully convolutional neural network (F-CNN) was proposed for the same purpose. Some studies have focused on jointly extracting spatial features for classification [20], leading to the development of a series of improved algorithms [22–24]. In [25,26], a compact and differentiated stacked autoencoder model was designed for HSI, utilizing low-dimensional feature maps. Other methods have aimed to improve HSI classification by extracting spectral–spatial features using enhanced deep learning networks [27–31]. Additionally, unlike traditional methods that directly extract spatial features from adjacent regions, new filtering techniques, such as Gabor filtering and attribute profiling, have been proposed [32]. Subsequent research introduced fusion strategies for HSI spectral–spatial classification, applying multiple CNNs in groups. However, the combination of multiple supervised CNNs is computationally expensive and time-consuming [33].

Given the limitations of existing network structures in learning and representing line features, and with a consideration of adaptive network optimization, a method for optimizing deep kernel mapping networks based on structural adaptation has been studied. This method aims to improve classification performance by adjusting the width and depth of the network. First, the structural and learning parameters of the deep kernel mapping network were optimized, and the network structure was adaptively adjusted according to the distribution characteristics of the data to verify the feasibility of enhancing image classification performance. Second, the kernel operator and network node optimization methods introduced in the subsequent three chapters were applied to this network. The deep kernel mapping network, based on multiple optimizations, was further studied to

improve its performance in image classification. Finally, the proposed deep kernel mapping network algorithm, based on three-factor multiple optimization, was comprehensively compared and analyzed against current image classification methods using an open dataset.

We summarize our contributions as follows:

(1) We present a novel deep learning model, DQKNet, for image classification. Through the combination of the deep kernel mapping network and quasiconformal kernel learning, the model optimizes the network structure and enhances the accuracy and efficiency of image classification.
(2) We optimize both the structural and learning parameters of the deep kernel mapping network, enabling the network to adapt its structure based on the data's distribution characteristics. This enhances image classification performance while allowing models to achieve more efficient learning and classification with limited computational resources.
(3) Through experiments on multiple public datasets, we demonstrate that DQKNet surpasses existing state-of-the-art algorithms in both computational efficiency and classification accuracy.

2. Materials and Methods

2.1. Framework

The infrastructure of the deep kernel mapping network is illustrated in Figure 1. The depth of the kernel mapping network, as computed by each network node, and the network connection weight calculations directly influence the network's performance. This includes the network node calculations based on the underlying kernel function of the operator. The "accounting child" is combined with the network node calculations to form a comprehensive network node computation. Therefore, to assess the applicability of the deep kernel mapping network in classification, its performance is analyzed from three key elements: the accounting child, network nodes, and the overall network structure.

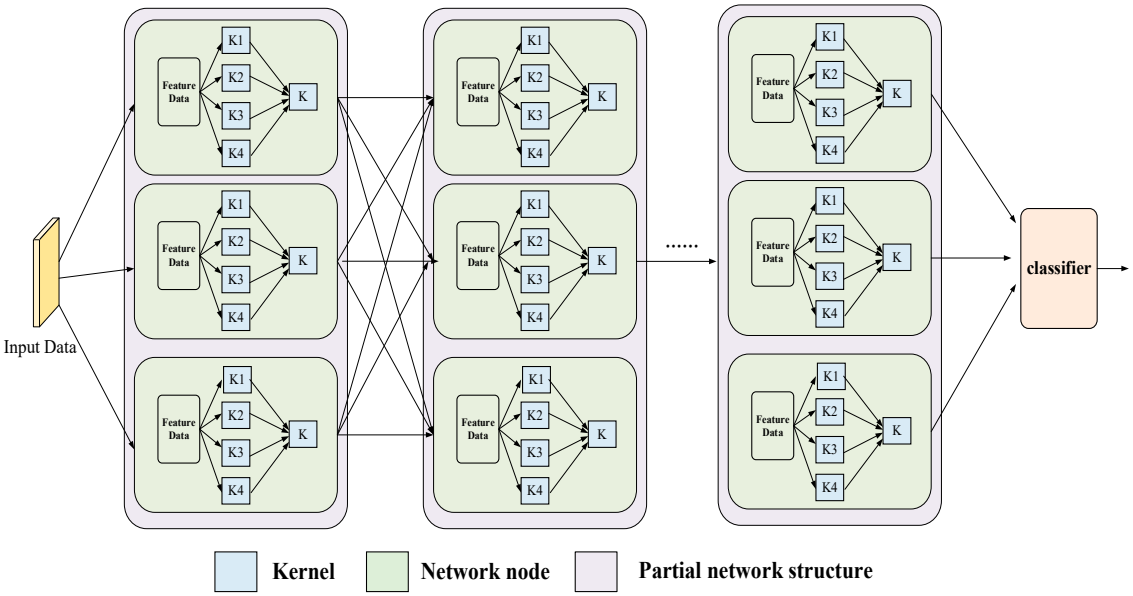


Figure 1. Architecture of the deep kernel mapping network and three elements.

The impact on classification performance is analyzed from three perspectives: accounting substructures, network nodes, and network structure. Addressing the suboptimization issue of deep mapping networks, combined with the challenge of strong coupling between features and high correlation in large bands, classification performance is enhanced through improvements in the inner construction of the kernel function operator. From the perspective of network node subcombination optimization, the network optimization node serves as the foundation for overall network optimization, solving the node optimization problem

in deep mapping networks. This approach addresses the challenges of strong nonlinearity and inaccurate representation from a single accounting substructure. Additionally, to tackle the network structure optimization issue and the limitations of existing network structures in feature learning and representation, adaptive optimization of the network structure is employed to effectively enhance feature extraction and classification performance. Although deep learning networks combined with kernel learning algorithms have shown promise, they do not yet enable adaptive adjustment of the network structure. Consequently, the network struggles to adapt to heterogeneous data. For the same dataset, classification accuracy varies based on the depth and width of the network structure. For datasets of different scales, a fixed depth and width structure cannot achieve optimal classification accuracy. To address this, the algorithm's structure was flexibly adjusted according to the dataset, allowing the deep kernel mapping structure to vary in depth and width. The deep kernel mapping network's performance is influenced by the calculations of each network node and the network connection weights. Each network node's calculations are based on a basic kernel function operator, with the kernel operator performing combinational calculations as part of the network nodes.

For data analysis, data $D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}$, including vector $x_i \in R^n$ and label vector $y_i \in \{-1, 1\}, i = 1, 2, \ldots, m$. Based on the kernel function $k(x_i, x_j)$ and the nonlinear mapping characteristics of the input data, $K_{D,m}$ is the combination kernel, and $\theta_m^{d,w}$ is the network node calculation for the basic combination of kernel operators. The decision function of the algorithm can be written as follows:

$$f(x) = sign(\sum_{k=1}^{M} \theta_k \sum_{i=1}^{m} \alpha_i y_i k(x_i, x) + b) \tag{1}$$

where θ_k is the combination coefficient of the structurally variable network algorithm. The structurally variable network algorithm is uniformly written as follows:

$$f(x) = sign(\sum_{i=1}^{m} \alpha_i y_i k(x_i, x; \theta_i) + b) \tag{2}$$

where θ, α, and b are learned by a structurally variable network algorithm. By extending the combinatorial kernel cascade to layer, the combinatorial kernel function of is defined as follows:

$$K^{(L)}(x, y) = \phi^{(L)}(\phi^{(L-1)}(\ldots \phi^{(1)}(x))) \cdot \phi^{(L)}(\phi^{(L-1)}(\ldots \phi^{(1)}(y))) \tag{3}$$

where x and y are the vectors input to the algorithm. $\phi^{(L)}$ is the kernel function. In addition to cascading multilayer combinatorial kernels, the structurally variable network algorithm also extends multiple deep structures into multiple channels. The kernel function is defined as follows:

$$K^{(l)}(x, y) = \left\{ \theta_{1,1}^{(l)} K_{1,1}^{(l)} \left(\theta_{1,1}^{(l-1)} K_{1,1}^{(l-1)} + \cdots \right) + \cdots \theta_{h,m}^{(l)} K_{h,m}^{(l)}(\ldots) \right\} \tag{4}$$

where $K_{h,m}^{(l)}$ is the mth basic kernel in the hth set in the Lth row and $\theta_{h,m}^{(l)}$ refers to the weights. The characteristic output of each channel is input into the combinatorial kernel according to the summation average rule. The combinatorial kernel is expressed as follows:

$$K_f = \frac{1}{W} \sum_{m=1}^{W} K_{D,m} \tag{5}$$

where K_f is the combination kernel of the last layer, except for the depth and width. The $K_{D,W}$ of the structurally variable network algorithm in each layer and channel is defined as follows:

$$\begin{aligned} K_{d,w} &= \sum_{m=1}^{M} \theta_m^{d,w} k_m^{d,w},\ d = 1,2,\ldots,D,\ w = 1,2,\ldots,W \\ s.t. &\sum_{m=1}^{M} \theta_m^{d,w} = 1, \\ &\theta_m^{d,w} > 0 \quad m = 1,2,\ldots,M \end{aligned} \tag{6}$$

where $k_m^{d,w}$ is the kernel function of each layer with the $\theta_m^{d,w}$.

Therefore, the optimization of DQKNet involves two perspectives, including the kernel mapping network node $k_m^{d,w}$ and the optimal network parameters $\theta_m^{d,w}$.

2.2. Learning $k_m^{d,w}$ for the Optimal Kernel Mapping Network Node

The main purpose of this part is to optimize the optimal network computing node $k_m^{d,w}$. Data will obtain different geometric structures in the nonlinear mapping space. Similarly, different kernel operators will result in different class discrimination abilities, which directly affects the effect of classification. Due to the restriction of the kernel function learning mechanism, different class discrimination abilities can be obtained in the nonlinear mapping space accordingly. At present, no kernel function can adapt to all data. The kernel function should change its structure depending on the input data; that is, the kernel function should be data-dependent, which is also the basic idea for constructing a data-dependent kernel function. The quasiconformal mapping kernel function is introduced as the objective kernel function for kernel optimization, and the Mahalanobis kernel function is extended by using mapping theory. Therefore, the network node $k_m^{d,w}$ is described mathematically by the $k_q(x,y)$ quasiconformal kernel as follows:

$$k_q(x,y) = f(x)f(y)k(x,y) \tag{7}$$

where x, y represents the sample vector and a is the basic accounting sub. In practical applications, polynomial and Gaussian kernels are generally used as the basic accounting sub. To improve the classification performance, the extended Mahalanobis distance calculation sub-A is adopted, assuming that a is a positive function, and it is expressed as follows:

$$f(x) = b_0 + \sum_{n=1}^{N_{XV}} b_n e(x, \widetilde{x}_{\mathbf{n}}) \tag{8}$$

where $e(x, \widetilde{x}_{\mathbf{n}}) = e^{-\delta \|x - \widetilde{x}_{\mathbf{n}}\|^2}$, δ is the weight adjustment parameter, $\widetilde{x}_n$ is called the expansion vector, N_{XV} is the number of expansion vectors, and b_n ($n = 0,1,2,\ldots,N_{XV_s}$) is the expansion coefficient associated with $\widetilde{x}_{\mathbf{n}}$ ($n = 0,1,2,\ldots,N_{XV_s}$). In practical work, the method of selecting the expansion vector directly affects the performance of the algorithm. In conventional calculation problems, researchers use randomly selected samples as the expansion vector. Accordingly, this paper uses different calculation methods to realize the feature calculation for different spectra.

Step 1: Expansion vector solving objective function construction

To solve for the expansion vector of the Markovian operator of a quasiconformal mapping, the optimal objective of the optimal mapping space is constructed, the optimal nonlinear mapping space is achieved by adjusting the optimal expansion coefficient, and optimal between-class discrimination is achieved in this space. Therefore, to solve for the appropriate expansion coefficient, it is difficult to calculate the feature space within the feature space of nonlinear mapped images to effectively construct an effective feature space to achieve data classification. Its mathematical description is as follows. For the input

training sample $\{x_i\}_{i=1}^n$, the $n \times n$ kernel matrix $K = [k_{ij}]_{n\times n}$ of r is constructed, and its basic matrix element a is follows:

$$k_{ij} = k(x_i, x_j) \tag{9}$$

where the dimension of the sample space is d. The matrix K constructed from the above elements, which is a positive definite symmetric matrix, can be decomposed by SVD into the following:

$$K_{n\times n} = P_{n\times r}\Lambda_{r\times r}P_{r\times n}^T \tag{10}$$

According to positive definite matrix decomposition analysis, the diagonal matrix Λ of its data is composed of eigenvalue decomposition values of the kernel matrix K, and the corresponding eigenvectors form matrix P. In this case, the mapping from the input space to the feature space can be expressed as follows:

$$\begin{aligned} &\Phi_r^e : \chi \to R^r \\ &x \to \Lambda^{-1/2:}P^T(k(x, x_1), k(x, x_2), \ldots, k(x, x_n))^T \end{aligned} \tag{11}$$

In the equation, the characteristics of space are obtained by mapping space $\Phi_r^e(\chi) \subset R^r$ according to experience, and the space vector is obtained by inflation adjustment to determine the distribution of the spectrum sample; thus, by performing a quasiconformal mapping Markov calculation to realize the optimal classification, from the expansion of the vector optimization algorithm, optimal classification results can be obtained. The sample $\{\Phi(x_i)\}_{i=1}^n$ of data in the nonlinear mapping space has the same geometric structure as the sample $\{\Phi_r^e(x_i)\}_{i=1}^n$ obtained by the Mahalanobis accounting submapping of the quasiconformal mapping in the empirical feature space. In other words, the classification of data can be realized by classification in space. Let Y be the matrix of $n \times r$ with column vectors $\Phi_r^e(x_i)$, as follows:

$$Y = KP\Lambda^{-1/2} \tag{12}$$

Therefore, the feature inner product matrix of $\{\Phi_r^e(x_i)\}$ in the empirical feature space after mapping the data is as follows:

$$YY^T = KP\Lambda^{-1/2}\Lambda^{-1/2}P^TK = K \tag{13}$$

Therefore, in the feature space $\{\Phi(x_i)\}_{i=1}^n$ of the feature space and in the spectral feature space of data, the distance n of nonlinear mapping vectors $\{\Phi(x_i)\}_{i=1}^n$ can be obtained by the following formula:

$$\left\|\Phi(x_i) - \Phi(x_j)\right\|^2 = k(x_i, x_i) + k(x_j, x_j) - 2k(x_i, x_j) \tag{14}$$

The above formula shows that the training data samples have different discriminative powers in the feature space with the mapping of the quasiconformal mapping Markovian operator, and the same-class discriminative ability of features can be obtained through the mapping network of the quasiconformal mapping Markovian operator. Therefore, in practical applications, the use of a deep kernel mapping network to realize data has the largest ground object class discrimination power, so the optimal objective function can be constructed for the optimization problem of a quasiconformal mapping Markovian calculation in the empirical feature space. In the subprocess of Mahalanobis accounting for data using quasiconformal mapping, for a given expansion vector, a constrained optimization equation is established to achieve accurate ground object classification, and the optimal expansion coefficient is solved by solving the optimization equation to achieve the optimal effect of ground object classification in the feature space. Therefore, by evaluating the discrimination of classes in the empirical feature space, in this paper, the maximum margin criterion (MMC) and Fisher criterion (FC) of feature categories in the empirical feature space are used to construct the optimal objective function. By using this principle, the class discrimination of samples in the empirical feature space can be measured. By using this

objective function, the objective function that solves the optimal expansion coefficient can be established, and the optimal equation can be constructed through this objective function.

In the empirical feature space, to measure the category discrimination in the empirical feature space, linear metric learning can be used to measure the space. A new metric function can be obtained through linear transformation, and the metric function obtained with the mapping is as follows:

$$\widetilde{d}(x,y) = \sqrt{(A^T x - A^T y)^T (A^T x - A^T y)} \tag{15}$$

If $M = AA^T$, M is a semisymmetric matrix. The formula is as follows:

$$\begin{aligned} \widetilde{d}(x,y) &= \sqrt{(A^T x - A^T y)^T (A^T x - A^T y)} \\ &= \sqrt{(x-y)^T AA^T (x-y)} \\ &= \sqrt{(x-y)^T M (x-y)} \end{aligned} \tag{16}$$

M is extended to half of the above equation for the matrix, which is a more general Markov matrix. The internal structure of the traditional Markov matrix is achieved using only the data. The aim is to describe the distribution characteristics of the data. Using the Markov distance matrix can be sufficient to determine the relationship between the characteristic and the label, and the Markov matrix can be used to distinguish between different characteristics of the contribution. The constrained optimization problem is defined as follows:

$$\begin{aligned} &\min_{\mathbf{M}} \mathcal{L}_X(M) + \lambda r(M) \\ &\text{s.t. } c_X(M) \end{aligned} \tag{17}$$

where $\mathcal{L}_X(M)$ and $c_X(M)$ are the loss function and the constraint on the training set.

In the network learning, for a given data sample set $X : \{x_i\}_{i=1}^{M} \subset \mathbb{R}^N$, the constrained set of category samples can be constructed, including a similar set W and dissimilar set B, where W is the constrained set constructed by sample pairs of the same category and set B is constructed in the opposite manner. Therefore, the following method of constructing a convex optimization problem can be used to optimize the Mahalanobis distance:

$$\begin{aligned} &\min_{\mathbf{M} \geq 0} \sum_{(\boldsymbol{x}_i, \boldsymbol{x}_j) \in W} d_M^2(x_i, x_j) \\ &\text{s.t.} \sum_{(\boldsymbol{x}_i, \boldsymbol{x}_j) \in B} d_M(x_i, x_j) \geq 1 \end{aligned} \tag{18}$$

where $d_M^2(x_i, x_j) = (x_i - x_j)^T M (x_i - x_j)$ and M is a positive semidefinite matrix. The constraint is added mainly to remove trivial solutions for $M = 0$. In the concrete solution, the iterative updating method can be used to solve the problem. In each iteration, the mature Newton downhill method is used for the gradient descent process to obtain an updated Markov matrix. In the calculation process, the implementation of the algorithm is simple. For the optimization problem, the extent of the optimization calculation changes with the optimization function, and convergence is relatively slow for the Mahalanobis operator, with a complex optimization process as follows:

$$d_M(x_i, x_j) = \sqrt{(A^T x_i - A^T x_j)^{\mathrm{T}} (A^T x_i - A^T x_j)} \tag{19}$$

On the constrained set W, under the action of projection matrix A, the sum of squared distances between all pairs of points is as follows:

$$d_W = \sum_{(\boldsymbol{x}_i, \boldsymbol{x}_j) \in W} (A^T x_i - A^T x_j)^{\mathrm{T}} (A^T x_i - A^T x_j) \tag{20}$$

The sum of squares of distances between all pairs of points in constrained set B can be calculated as follows.

$$\begin{aligned} d_B &= \sum_{(x_i, x_j) \in B} (A^T x_i - A^T x_j)^{\mathrm{T}} (A^T x_i - A^T x_j) \\ &= tr(A^T S_B A) \end{aligned} \tag{21}$$

The optimal objective function is constructed to achieve a constrained solution for the similar set W and dissimilar set B. The method of solving for the optimal A^* is as follows:

$$A^* = \underset{\mathbf{A}^{\mathrm{T}}\mathbf{A}=\mathbf{I}}{\operatorname{argmax}} \frac{tr(A^T S_B A)}{tr(A^T S_W A)} \tag{22}$$

Therefore, computing the Mahalanobis distance matrix $\mathbf{M}^* = \mathbf{A}^*(\mathbf{A}^*)^{\mathrm{T}}$ is a post-learning process. In the learning domain of the algorithm, the learning criterion maximizes the accuracy of the test data. Given the function $f : x \to R$, the threshold version of f is defined as follows:

$$er(f) = \frac{1}{n} |\{n+1 \le i \le 2n : y_t f(x_i) \le 0\}| \tag{23}$$

where the kernel-based classifier is the threshold of the formal kernel expansion operation, and the bounded norm is as follows:

$$||w||^2 = \sum_{i,j=1}^{2n} \alpha_i \alpha_j k(x_i, x_j) = \boldsymbol{\alpha}^T K \boldsymbol{\alpha} \le \frac{1}{\gamma^2} \tag{24}$$

For any $\gamma > 0$ with a probability of at least $1 - \delta$ in the data (x_i, y_i), the $er(f)$ of each function $f \in F_k$ does not exceed $\frac{1}{n} \sum_{i=1}^{n} \max(1 - y_i f(x_i)) + \frac{1}{\sqrt{n}} (4 + \sqrt{2 \log(\frac{1}{\delta})} + \sqrt{\frac{\varsigma(k)}{n\gamma^2}})$, where $\varsigma(\boldsymbol{K}) = E \underset{k \in \boldsymbol{K}}{\max} \sigma^T K \sigma$ is the expected value for σ. Assuming that λ_j is the largest eigenvalue of K_j, then Equations (25) and (26) are as follows:

$$\varsigma(\boldsymbol{K}) = cE \underset{k \in \boldsymbol{K}}{\max} \sigma^T \frac{K}{trace(K)} \tag{25}$$

$$\varsigma(\boldsymbol{K}) \le c \min(m, n \max_J \frac{\lambda_J}{trace(K_J)}) \tag{26}$$

In conclusion, for the classification of ground objects, the Fisher criterion and maximum interval criterion are implemented in the empirical feature space mapping, and the expansion vector is solved by solving the optimization objective function.

Step 2 Optimization function solution

To solve the problems of objective functions based on the Fisher criterion and the maximum interval criterion, the following equation-solving methods are used to solve the two objective functions.

(1) Optimization with the Fisher criterion

The gradient method is used to solve the optimization equation based on the Fisher criterion, and the partial differential equations for $J_1(\alpha) = \alpha^T E^T B_0 E \alpha$ and $J_2(\alpha) = \alpha^T E^T W_0 E \alpha$ are obtained

(2) Optimization of the objective function based on the Fisher criterion

The gradient method is used to solve the optimization equation based on the Fisher criterion. Let $J_1(\alpha) = \alpha^T E^T B_0 E \alpha$ and $J_2(\alpha) = \alpha^T E^T W_0 E \alpha$. The partial differential equation for $J_1(\alpha)$ and $J_2(\alpha)$ to α is determined as follows:

$$\frac{\partial J_1(\alpha)}{\alpha} = 2 E^T B_0 E \alpha \tag{27}$$

$$\frac{\partial J_2(\alpha)}{\alpha} = 2E^T W_0 E\alpha \tag{28}$$

Then, the partial differential of $J_{Fisher}(\alpha)$ is as follows:

$$\frac{\partial J_{Fisher}(\alpha)}{\partial \alpha} = \frac{2}{J_2^2}(J_2 E^T B_0 E - J_1 E^T W_0 E)\alpha \tag{29}$$

To optimize and maximize J_{Fisher}, according to the gradient descent method, $\frac{\partial J_{Fisher}(\alpha)}{\partial \alpha} = 0$. Then, Equation (30) is as follows:

$$J_1 E^T W_0 E\alpha = J_2 E^T B_0 E\alpha \tag{30}$$

Therefore, if $\left(E^T W_0 E\right)^{-1}$ exists, then Equation (31) is as follows:

$$J_{Fisher}\alpha = (E^T W_0 E)^{-1}(E^T B_0 E)\alpha \tag{31}$$

Then, J_{Fisher} is the maximum eigenvalue of $(E^T W_0 E)^{-1}(E^T B_0 E)$, and the obtained eigenvector is the optimal expansion coefficient α of the objective function based on the Fisher criterion.

In the data classification problem, $(E^T W_0 E)^{-1}(E^T B_0 E)$ is asymmetric, and $E^T W E$ is a singular matrix. In this case, the cyclic iterative calculation method is adopted to solve α. Then, Equation (32) is as follows:

$$\alpha^{(n+1)} = \alpha^{(n)} + \varepsilon(\frac{1}{J_2}E^T B_0 E - \frac{J_{Fisher}}{J_2}E^T W_0 E)\alpha^{(n)} \tag{32}$$

where ε is the learning rate, so Equation (33) is as follows:

$$\varepsilon(n) = \varepsilon_0(1 - \frac{n}{N}) \tag{33}$$

In the calculation process, ε_0 is the initial value of the learning rate, and n and N are the current iteration and the total iteration number, respectively.

Therefore, in terms of the learning rate of the initialization parameter ĉ and the number of iterations N, these parameters directly determine the solution time of the algorithm, and these parameters determine the total solution time of the optimization function. Therefore, the optimal expansion coefficient vector can only be obtained when n and N are properly chosen. Because the cyclic iteration method is used to solve the probl2m, the optimal solution is often not unique and depends on the selection of initialization parameters.

(3) Optimizing with the maximum interval criterion

Maximum interval optimization equations are deduced with the Lagrange method, converting equation for the optimal characteristic equation with the λ parameter. The method solves the $2\widetilde{S}_B - \widetilde{S}_T$ matrix eigenvalue and eigenvector to obtain the solution for the optimal α^*. This parameter enables the maximum interval of categories to be obtained. Thus, Equations (34) and (35) are as follows:

$$P^T \widetilde{S}_B P = \Lambda \tag{34}$$

$$P^T \widetilde{S}_T P = I \tag{35}$$

Assuming that θ and ϕ are the eigenvalues and eigenvectors of matrix $\widetilde{S}_T$, $P = \phi\theta^{-1/2}\psi$ can be calculated, where ψ is calculated by matrix $\theta^{-1/2}\phi^T \widetilde{S}_B \phi\theta^{-1/2}$. Among them, matrix P is calculated by using $2\widetilde{S}_B - \widetilde{S}_T$. $P = \phi\theta^{-1/2}\psi$ is calculated using SVD decomposition technology, and the $n \times m$ matrix A is as follows:

$$A = U\Lambda^{1/2}V \tag{36}$$

where $\Lambda^{1/2}$ is a diagonal matrix and U and V are the U and V column vectors of $n \times \min(n, m)$ and $m \times \min(n, m)$, respectively, which are feature vectors of AA^T and A^TA.

2.3. Learning $\theta_m^{d,w}$ for the Optimal Network Structure

To optimize the learning parameters $\theta_m^{d,w}$ of the deep kernel mapping network, according to the estimation of T_{span} derived from the support vector span space, the structurally variable network algorithm minimizes the upper bound T_{span} of error to optimize the algorithm parameters. The specific formula is as follows:

$$\begin{gathered} \min_{\theta} T_{span}, s.t.\ \theta_i \geq 0,\ \sum_{i=1}^{M} \theta_i = 1 \\ T_{span} = \sum_{p=1}^{n} \psi\left(\alpha_p^0 S_p^2 - 1\right) \end{gathered} \tag{37}$$

where α_p^0 is the coefficient of the SVM; n is the number of support vectors; S_p is the distance between point $\phi(x_p)$ and set Λ_p; and x_p is the support vector. Specifically, Λ_p is defined as follows:

$$\Lambda_p = \left\{ \sum_{i \neq p, \alpha_i^0 > 0} \lambda_i \phi(x_i), \sum_{i \neq p} \lambda_i = 1 \right\} \tag{38}$$

The structurally variable network algorithm uses a constructor $\psi(x)$ to obtain a smooth error approximation. The constructor is expressed as follows:

$$\psi(x) = (1 + \exp(-Ax + B))^{-1} \tag{39}$$

where A and B are constants. During the implementation of this algorithm, the values are $A = 5$, $B = 0$ S_p^2, which can be expressed as follows:

$$S_p^2 = \frac{1}{(\widetilde{K}_{sv}^{-1})_{pp}} \tag{40}$$

where sv is a set of support vectors; $sv = \left\{ x_p \middle| \alpha_p^0 > 0, p = 1, 2, \ldots, l \right)$; and K_{sv} is the dot product matrix between support vectors, as follows:

$$\widetilde{K}_{sv} = \begin{pmatrix} K_{sv} & 1 \\ 1^T & 0 \end{pmatrix} \tag{41}$$

The values given by the span of the above formula are not continuous. The algorithm of the variable structure network uses regularization terms to replace constraints while calculating S_p^2 to smooth the S_p^2 values. The formula is as follows:

$$S_p^2 = \min_{\lambda, \sum_i \lambda_i = 1} \left\| \phi(x_p) - \sum_{i \neq p}^{n} \lambda_i \phi(x_i) \right\|^2 + \eta \sum_{i \neq p}^{n} \frac{1}{\alpha_i^0} \lambda_i^2 \tag{42}$$

The matrix expression is abbreviated as follows:

$$S_p^2 = 1/(\widetilde{K}_{sv}^{-1} + Q)_{pp}^{-1} - Q_{pp} \tag{43}$$

where Q is a diagonal matrix and the matrix elements are $Q_{ii} = \eta / \alpha_i^0$, $Q_{n+1,n+1} = 0$; η is a constant. In the structurally variable network algorithm, $\eta = 0.1$.

The SVM coefficient α is fixed to solve for the combination coefficient θ, and the combination coefficient θ is fixed to solve for the SVM coefficient α. The two steps are iterated alternately. When the difference between the S_p^2 values of i and $i - 1$ is less than e^{-4}

or the algorithm is iterated 100 times, the algorithm parameters stop updating. The obtained model is the optimal model for testing. The parameter update formula is as follows:

$$\theta^{i-1} \leftarrow \theta^{i} + lr \cdot \frac{\partial T_{span}}{\partial \theta} \tag{44}$$

where $\partial T_{span}/\partial\theta$ calculates the gradient update direction and the partial derivative is approximately expressed as follows:

$$\frac{\partial T_{span}}{\partial \theta} = -\sum_{p=1}^{n} A\psi(x_p)^2 \cdot \exp(Ax_p + B) \cdot \frac{\partial S_p^2}{\partial \theta} \tag{45}$$

According to the specific definition of S_p^2, the partial derivative can be calculated as follows:

$$\frac{\partial S_p^2}{\partial \theta_p} = (1/B_{pp}^{-1})^2 (B^{-1}(\frac{\partial \widetilde{K}_{sv}}{\partial \theta_p} + GF)B^{-1})_{pp} - (GF)_{pp} \tag{46}$$

where the matrix $B = \widetilde{K}_{sv} + Q$ through the diagonal matrix G, where the matrix elements are $G_{ii} = -\eta/(\alpha_{sv_i}^0)^2, G_{n+1,n+1} = 0$; $F = diag(Y_{sv}\overline{A}K_{sv}Y_{sv}\alpha_{sv}^0; 0)$, which is calculated by $Y_{sv} = diag((y_{sv_1}, y_{sv_2}, \ldots, y_{sv_n})^T)$. $\overline{A}$ is the inverse matrix, where $\widetilde{K}_{sv}$ cuts off the last row and column, and $\partial\alpha_{sv}^0/\partial\theta_p = -Y_{sv}\overline{A}K_{sv}Y_{sv}\alpha_{sv}^0$. Through the above matrix calculation, the optimal equation is solved.

2.4. Algorithm Steps

Combined with the analysis and optimization ideas of different influencing factors, the kernel operator, network node, and network structure optimization methods are adopted to construct a classification method framework based on deep kernel mapping network multiple optimization. Regarding the kernel operator, the Mahalanobis distance kernel operator is more beneficial to classification. In terms of network nodes, using different multikernel combination structures to analyze and learn network nodes can effectively improve the effect of classification. In terms of network structure, a hybrid deep kernel learning network can effectively improve the accuracy of classification. The overall flow of the algorithm is shown in Algorithm 1.

Step 1. Basic kernel operator optimization of the deep mapping network

The Mahalanobis distance calculation subparameter and basic calculation subparameter are selected, and the Mahalanobis distance matrix is constructed to determine the contribution degree of different features to the classification problem. Combined with different imaging conditions, the coupling correlation relationship of features is adjusted by the Mahalanobis distance matrix to improve the classification ability of features. On this basis, an operator based on quasiconformal mapping is proposed, which can improve the performance of the Markovner model through the theory of quasiconformal mapping.

Step 2. Network node optimization of the deep mapping network

The network node optimization method based on the nonlinear combination of accounting subsets is adopted. The data dependence function is used to realize the nonlinear combination optimization of different accounting subsets, and the objective function is solved. The multifunction optimization strategy is constructed by using multiple mapping characteristics, and the optimal parameters of nonlinear combination optimization are solved.

Step 3. Optimizing the network structure of the deep mapping network

Different kernel structure parameters are used to select the structure. After selecting the appropriate structure, the optimization results of Step 2 and Step 3 are used to optimize the structure parameters and learning parameters of the deep kernel mapping network.

Step 4. Deep kernel mapping network classification

A support vector machine is used as a classifier to solve the basic data classification problem, and its classification decision function is expressed as follows:

$$f(x) = sign(\sum_{i=1}^{n} \alpha_i y_i (x_i \cdot x) + b) \tag{47}$$

where α_i is the dual coefficient and b is the deviation of the decision function $f(x)$. The optimization problem of the support vector machine is as follows:

$$\begin{aligned} &\min_{(\alpha,b,\xi)} \tfrac{1}{2}\|\omega\|^2 + C\sum_{i=1}^{m} \xi_i \\ &s.t.\ \mathrm{y}_i(\alpha_i k(x_i, x) + b) \geq 1 - \xi_i, \\ &\quad \xi_i \geq 0, C > 0, i = 1, 2, \ldots, m \end{aligned} \tag{48}$$

where ξ_i is the relaxation variable and C is the regularization coefficient.

The optimization of this part mainly analyzes the influence of different network depths and widths on the algorithm. The depth and width of the algorithm are limited to the range $D \in [4, 5, 6], \mathrm{W} \in [1, 2, 3]$. After training, the combination coefficients of the variable-depth and width-structure of the SVM classification model and algorithm with classification ability are obtained. The training data are expressed as $I = \{(x_i, y_i)|i = 1, 2, \ldots .m\}$.

Algorithm 1: Model for Structurally Variable Networks (I, θ^0, D, W, lr, C)

Input: $I = \{(x_i, y_i)|i = 1, 2, \ldots .m\}$, θ^0 $\frac{1}{W}$, D, W, lr, C.
Output: parameter matrix.
1 initialize k, calculate K;
2 for $i \leftarrow 1$ to N do
3. Calculate the optimal vector using K_f;
4 for $w \leftarrow 1$ to W do
5 for $d \leftarrow 1$ to D do
6 Calculate $\widetilde{K}_{sv}$, B, $\frac{\partial S_p^2}{\partial \theta}$; $\frac{\partial T_{span}}{\partial \theta}$
7 update; $\theta^{i-1} \leftarrow \theta^i + lr \cdot \frac{\partial T_{span}}{\partial \theta}$
8 end
9 end
10 end

Due to the direct cascade between different layers of the structurally variable network algorithm, that is, the output of the combinatorial kernel function of the upper layer is the input of the base kernel function of the next layer and each channel is independent of the others, it is easy to adjust the combination mode of the depth and width of the algorithm. For the same dataset, the classification accuracies of structurally variable network algorithms with variable-depth and variable-width combined architectures are different. For datasets of different sizes, if the combination structure of depth and width is fixed, the classification accuracy cannot be the highest in all datasets.

3. Results and Discussion

Experimental Settings

In this paper, five publicly collected datasets are used to verify the algorithm performance. At present, the Indian Pines data, University of Pavia dataset, Salinas dataset, Houston 2013 dataset, and Houston 2018 dataset are mainly used. These datasets are widely used to verify the performance of the algorithm. These datasets are the data collected under different environmental conditions, which are fair to the algorithm verification. These datasets can reflect the effect of the actual application of the algorithm and verify the performance of the algorithm in the actual application. Since the dataset used is the measured data under natural conditions, and the target and imaging conditions are also in

line with the actual application scenarios, this experiment can reflect the real application conditions and realize the transformation of engineering applications in the future.

(1) Indian Pines dataset

Indian Pines dataset: This dataset is an open dataset collected and constructed by researchers in the Indian Pines region of the United States using AVIRIS sensor in 1992, including agriculture, forest, and other ground objects. The spatial resolution of this spectral dataset is 145 × 145 relative resolution and 20 m absolute spatial resolution. The data include 16 types of ground objects with 20~2468 pixels, as shown in Table 1. As one of the most common datasets used in the current field, this dataset can fairly evaluate the advantages and disadvantages of various algorithms. Therefore, as the basis for comprehensive comparison and analysis of algorithms in this paper, it can provide important data support for the evaluation and analysis of future algorithms.

Table 1. Description of the Indian Pines data.

Category Number	Category	Total Number of Samples	Training Sample	Test Sample
1	C_1—Non-tilled corn	1434	50	1384
2	C_2—Less-tilled corn	834	50	784
3	C_3—Corn	234	50	184
4	C_4—Grass/pasture	497	50	447
5	C_5—Grass/trees	747	50	697
6	C_6—Dry grass heap	489	50	439
7	C_7—Less-tilled corn	968	50	918
8	C_8—Non-tillled corn	2468	50	2418
9	C_9—Pure corn	614	50	564
10	C_{10}—Wheat	212	50	162
11	C_{11}—Forest	1294	50	1244
12	C_{12}—Buildings–grass–trees	380	50	330
13	C_{13}—Clover	46	23	23
14	C_{14}—Builtback meadow	28	14	14
15	C_{15}—Oats	20	10	10
16	C_{16}—Stone–steel tower	93	46	47

(2) University of Pavia dataset

University of Pavia Dataset: The data were collected using the Reflectance Optical System Imaging Spectrometer (ROSIS-3) at the University of Pavia, Italy, in July 2002. The data were collected by the sensor in the spectral dimension of between 0.43 μm and 0.86 μm. In this paper, 103 bands are used for research and comparative application. As shown in Table 2, the description of ground object categories in the University of Pavia dataset shows the number of ground object categories and ground object targets. This dataset is widely used in the verification experiments of classification algorithms, and different experimental results are obtained in different algorithms. Most of the experimental results have relatively high classification accuracy. The public dataset constructed by this dataset includes agriculture, forest, and other ground objects. The high spatial resolution and spectral resolution of this dataset can objectively evaluate the accuracy of the algorithm in practical applications.

Table 2. Description of the University of Pavia data.

Category Number	Category	Total Number of Samples	Training Sample	Test Sample
1	C_1—Asphalt pavement	6852	50	6802
2	C_2—Grass	18,686	50	18,636
3	C_3—Gravel roof	2207	50	2157
4	C_4—Trees	3436	50	3386
5	C_5—Metal roof	1378	50	1328
6	C_6—Bare land	5104	50	5054
7	C_7—Asphalt roof	1356	50	1306
8	C_8—Brick road	3878	50	3828
9	C_9—Shadow	1026	50	976

(3) Salinas dataset

Salinas dataset: collected by AVIRIS in the Salinas Valley in Southern California. The images have 224 bands and 51,217 pixels, with a spatial resolution of 3.7 m. For detailed information, refer to Table 3.

Table 3. Description of the Salinas data.

Category Number	Feature Category	Training	Test
1	Brocoli-green-weeds-1	50	1959
2	Brocoli-green-weeds-2	50	3676
3	Fallow	50	1926
4	Fallow-rough-plow	50	1344
5	Fallow-smooth	50	2628
6	Stubble	50	3909
7	Celery	50	3529
8	Grapes-untrained	50	11,221
9	Soil-vineyard-develop	50	6153
10	Corn-senesced-green-weeds	50	3228
11	Lettuce-romaine-4wk	50	1018
12	Lettuce-romaine-5wk	50	1877
13	Lettuce-romaine-6wk	50	866
14	Lettuce-romaine-7wk	50	1020
15	Vineyard-untrained	50	7218
16	Vineyard-vertical-trellis	50	1757

According to the reference classification map shown in the figure, there are 16 different categories marked with different colors. The image of Salinas, collected by the AVIRIS sensor in Salinas, California, consists of 512 × 217 pixels with a spatial resolution of 3.7 m/pixel.

(4) Houston 2013 dataset

Houston 2013: This dataset has been made public in the 2013 IEEE Data Fusion Competition. The images were acquired through the National Aeronautical Laser Mapping Center (NCALM). A total of 15,029 ground truth samples are available, divided into 15 LULC categories. In this paper, the sample set is divided into 2832 training samples and 12,197 test samples. The distribution of samples is shown in Table 4.

Table 4. Description of the Houston 2013 data.

Category Number	Category Information	The Total Number of Samples	The Training Sample	Testing Sample
1	Healthy grass	1251	50	1201
2	Stressed grass	1254	50	1204
3	Synthetic grass	697	50	647
4	Trees	1244	50	1194
5	Soil	1242	50	1192
6	Water	325	50	275
7	Residential	1268	50	1218
8	Commercial	1244	50	1194
9	Road	1252	50	1202
10	Highway	1227	50	1177
11	Railway	1235	50	1185
12	Parking lot 1	1233	50	1183
13	Parking lot 2	469	50	419
14	Tennis court	428	50	378
15	Running track	660	50	610

(5) Houston 2018 dataset

Houston 2018: The dataset was taken from the 2018 IEEE Data Fusion Competition. There are 20 LULC categories distributed in 2,018,910 ground truth samples. From these samples, 2000 samples (50 samples in each class) were randomly selected for training, and the remaining 2,016,910 samples were tested. Compared with the test samples, the number of training samples is very small (0.1%), so the effect of overlap between training and test samples is negligible. The category information and spatial distribution are shown in Table 5.

Table 5. Description of the Houston 2018 data.

Category Number	Category Information	Training Sample	Testing Sample
1	Healthy grass	50	39,096
2	Stressed grass	50	129,908
3	Artificial turf	50	2636
4	Evergreen trees	50	54,222
5	Deciduous trees	50	20,072
6	Bare earth	50	17,964
7	Water	50	964
8	Residential buildings	50	158,895
9	Nonresidential buildings	50	894,669
10	Roads	50	183,183
11	Sidewalks	50	135,935
12	Crosswalks	50	5959
13	Major thoroughfares	50	185,338
14	Highways	50	39,338
15	Railways	50	27,648
16	Paved parking lots	50	45,832
17	Unpaved parking lots	50	487
18	Cars	50	26,189
19	Trains	50	21,739
20	Stadium seats	50	27,196

This group of experiments mainly evaluate the effects of basic accounting sub-network nodes and network structure on the classification performance of images, as shown in Table 6. The kernel base operator adopts a Euclidean polynomial kernel operator and Euclidean Gaussian kernel operator. Network nodes adopt an independent multikernel combination of kernel computing structures. The network structure adopts a deep mapping network structure that is 4–5 layers deep. The network's structure is shown in the table.

Table 6. Configurations of the 3 elements of the deep kernel mapping network.

Optimization Elements	Basic Kernel Operator	Network Node	Network Structure
Elements setting	European Poly nucleo-1 European RBF kernel-1	Combination of kernel operators of one class	4-layer network
	European Poly nucleo-2 European RBF kernel-2	Combination of multi-class kernel operators	5-layer network

As for the performance evaluation index of the algorithm, this paper mainly uses the image classification effect as the basis for evaluating the performance of the algorithm, and mainly uses the following three types of indicators as the evaluation index: the kappa coefficient (KC), overall accuracy (OA), and average accuracy (AA).

The mathematical expression of average classification accuracy (AA) is as follows:

$$AA = \sum_{i=1}^{m} \left(\mathbf{X}_{ii} / \sum_{j=1}^{m} \mathbf{X}_{ij} \right) \tag{49}$$

where m is the number of categories and $\mathbf{X}_{ii}$ is the correct classification number of i class pixels, that is, the value on the main diagonal of the confusion matrix. $\mathbf{X}_{ij}$ is the actual number of correct or incorrect categories.

The calculation formula of total classification accuracy (OA) is as follows. The calculation formula includes the following:

$$\mathrm{OA} = \frac{1}{n}\sum_{i=1}^{C}\mathbf{X}_{ii} \tag{50}$$

where X is the confusion matrix composed of the classification results of test samples; C is the total number of categories in the sample. The calculation formula of the kappa coefficient was calculated using the following indicators:

$$\mathrm{Kappa} = \left(n\left(\sum_{i=1}^{C}\mathbf{X}_{ii}\right) - \sum_{i=1}^{C}\left(\sum_{j=1}^{C}\mathbf{X}_{ij}\sum_{j=1}^{C}\mathbf{X}_{ji}\right)\right) / \left(n^2 - \sum_{i=1}^{C}\left(\sum_{j=1}^{C}\mathbf{X}_{ij}\sum_{j=1}^{C}\mathbf{X}_{ji}\right)\right) \tag{51}$$

In terms of classification, in addition to classification accuracy, there is also computational efficiency, which involves comparing the performance of classification. At present, in the research field, the main concern is classification accuracy. Generally speaking, the computational burden of classification is not large, and part of the computational burden can be solved by using equipment with large computing power. Therefore, this paper mainly evaluates the classification accuracy of the algorithm.

This part of the experiment is divided into two parts. First, the feasibility of the network structure optimization method of the deep kernel mapping network is verified. This part is validated with a public dataset. The main purpose is to verify the effect of network structure optimization on the performance of the deep kernel mapping network. Second, it verifies the application in classification. This part mainly verifies the feasibility analysis of the network structure optimization algorithm in classification. Finally, combined with the results of network optimization, the comprehensive evaluation of the data classification performance of the multi-optimization network is realized.

(1) Feasibility verification of network structure optimization

In this paper, five public datasets were used to verify the algorithm performance. Currently, the Indian Pines dataset, University of Pavia dataset, Salinas dataset, Houston 2013 dataset, and Houston 2018 dataset are used. These datasets are widely used to verify algorithm performance and to analyze the algorithm performance of different network structures for different types of data. In terms of parameter setting, the learning rate is e^{-5}, the maximum number of iterations is 100, the penalty coefficient range of the SVM classifier is $C = [10^{-1}, 10, 10^2]$, and the deep margin of the algorithm is $D \in [4, 5, 6]$, $\mathrm{W} \in [1, 2, 3]$. The structure of the algorithm can be adjusted flexibly according to the dataset, which is the main characteristic of the structurally variable network algorithm. The depth and width range of the architecture are limited to $D \in [4, 5, 6]$, $\mathrm{W} \in [1, 2, 3]$. Thus, nine different combinations of depth and width can be obtained, including 4×1, 4×2, 4×3, 5×1, 5×2, 5×3, 6×1, 6×2, and 6×3. Using the different datasets, the combination of depth and width is determined by algorithm learning. The depth D and width W of the architecture are taken as parameters to be learned and participate in algorithm training. The depth and width parameters of the classification results are optimized by the iterative training of different combination structures using a grid search algorithm.

In Tables 7 and 8, nine combination structures with different depths and widths are used for different data. The classification accuracy of the algorithm is shown in the table. The numbers in brackets represent the ranking of test set classification accuracy with the specified depth and width combination structure. The smaller the number is, the better the algorithm performance. The average classification accuracy of different widths is sorted, and this problem is explained in the last part of the table. For different data, the highest

classification accuracy of the deep and wide combination architecture is different, which verifies the feasibility of the network structure data adjustment algorithm structure.

Table 7. Classification accuracy with different network structures on the Indian Pines data (%).

Feature Category	Depth of the Structure	Width of the Structure		
		1	2	3
C_1—Non-tilled corn	4	82.94 (7)	81.06 (8)	86.69 (5)
	5	85.86 (6)	86.69 (5)	87.45 (3)
	6	87.09 (4)	87.65 (2)	**88.96 (1)**
C_2—Plow corn less	4	83.44 (8)	85.38 (7)	87.69 (5)
	5	85.38 (7)	88.47 (3)	86.48 (6)
	6	89.31 (2)	**89.59 (1)**	88.30 (4)
C_3—corn	4	82.05 (8)	85.28 (7)	85.28 (7)
	5	85.79 (6)	86.19 (4)	86.23 (3)
	6	86.04 (5)	**87.96 (1)**	87.45 (2)
C_4—Grass/pasture	4	84.02 (9)	85.71 (8)	86.54 (6)
	5	86.91 (5)	**91.27 (1)**	86.22 (7)
	6	88.37 (4)	89.76 (3)	90.45 (2)
C_5—Grass/trees	4	84.53 (8)	88.73 (2)	**88.91 (1)**
	5	85.11 (7)	85.27 (6)	84.50 (9)
	6	86.31 (5)	86.35 (4)	87.79 (3)
C_6—Hay pile	4	84.57 (8)	86.38 (6)	90.36 (2)
	5	86.21 (7)	90.09 (3)	88.17 (4)
	6	87.52 (5)	**91.62 (1)**	90.09 (3)
C_7—Plantless soybeans	4	80.02 (8)	81.92 (7)	85.21 (4)
	5	83.65 (6)	84.27 (5)	86.44 (3)
	6	86.44 (3)	86.49 (2)	**86.80 (1)**
C_8—Non-tilled soybeans	4	85.34 (8)	89.44 (4)	88.41 (5)
	5	87.18 (7)	89.44 (4)	90.76 (3)
	6	87.26 (6)	91.35 (2)	**92.12 (1)**

Table 8. Classification accuracy with different network structures on the Indian Pines data (%) (continued).

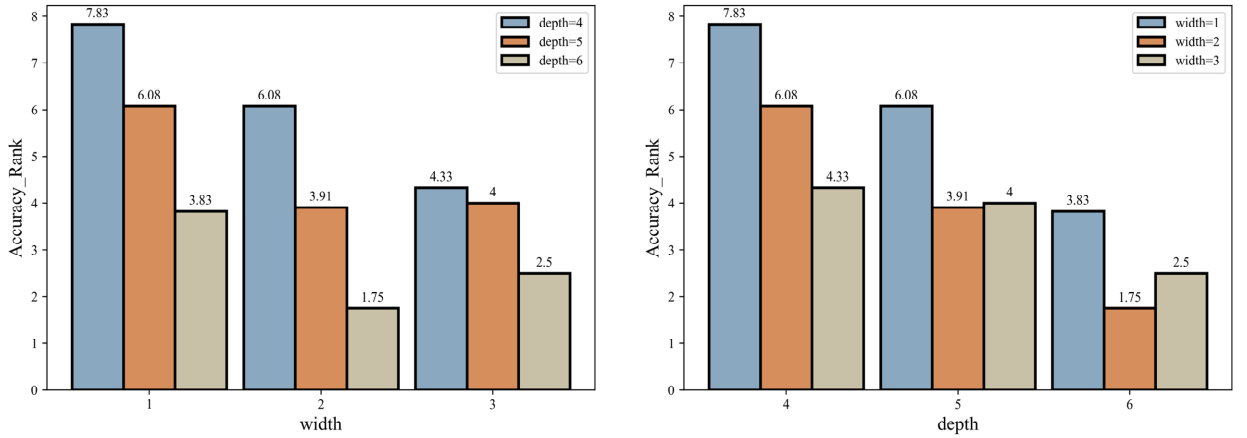
Feature Category	Depth of the Structure	Width of the Structure		
		1	2	3
C_9—Pure soybean	4	84.02 (8)	84.43 (7)	88.21 (4)
	5	86.23 (6)	89.14 (2)	**89.70 (1)**
	6	88.87 (3)	**89.70 (1)**	88.20 (5)
C_{10}—Wheat	4	80.07 (8)	86.42 (4)	83.48 (6)
	5	83.41 (7)	85.59 (5)	88.30 (2)
	6	87.56 (3)	**88.88 (1)**	85.59 (5)
C_{11}—Forest	4	83.32 (9)	84.25 (8)	90.20 (2)
	5	85.39 (7)	88.57 (5)	88.79 (4)
	6	89.56 (3)	**90.32 (1)**	87.48 (6)
C_{12}—Buildings–grass–trees	4	84.37 (7)	83.26 (8)	87.42 (5)
	5	86.99 (6)	88.39 (4)	88.74 (3)
	6	88.74 (3)	89.53 (2)	**89.96 (1)**
The average ranking		6.08	4.00	**3.72**

The classification accuracy ranking of structures with different depth and width combinations is shown in Table 9 and Figure 2. When the width is 1, 2, or 3, the classification accuracy of the algorithm increases with increasing depth. Similarly, when the depth is

4, the classification accuracy of the algorithm increases with increasing width. When the depth is 5 or 6, the classification accuracy of the algorithm increases first and then decreases with increasing width. Therefore, for different datasets, the depth and width combination structure of the adaptive adjustment algorithm can better extract features and improve the classification performance. The experimental results are fine, but not all the deepest and widest structures have the best classification results. Therefore, it is feasible to use the variable structure network based on data to improve the resolution of samples. The results of the experiment on the University of Pavia dataset, Salinas dataset, Houston 2013 dataset, and Houston 2018 dataset are shown in Tables A1–A11 and Figures A1–A4 (in Appendix A). The experimental structure includes the classification accuracy and ranking of different network structures.

Table 9. Rank of classification accuracies with different network structures on the Indian Pines data.

(Depth, Width)	1	2	3
4	7.83	6.08	4.33
5	6.08	3.91	4.00
6	3.83	**1.75**	2.50



(a) **Classification Accuracy With Different Depth On The Indian Pines Data** (b) **Classification Accuracy With Different Width On The Indian Pines Data**

Figure 2. Rank of classification accuracies with different widths and depths on the Indian Pines data.

In summary, the network structure adaptive search method can effectively improve the classification performance of data. In the process of using the structure variable network algorithm, determining the depth and width of the combined structure for specific datasets can effectively improve the accuracy of the algorithm and is more in line with the requirements of the application. The algorithm has a clear mathematical explanation and a clear and reasonable process and is easy to implement. Therefore, adaptive optimization of the network structure is feasible.

(2) Feasibility verification of the multi-optimization network

The experiments were conducted on the Indian Pines, University of Pavia, Salinas, Houston 2013, and Houston 2018 datasets to verify the comprehensive performance of different optimization methods. The application performance of operator optimization, operator optimization and node optimization double optimization, and operator, node, and structure optimization were evaluated. Overall accuracy (OA) was used to evaluate the network evaluation standard. The kernel base operator adopts the Euclidean distance calculation suboperator and Mahalanobis distance kernel operator, the network node adopts a linear combined computing node and nonlinear combined computing node, and the network structure adopts a deep mapping network structure of four, five, and six layers, as shown in Table 10.

Table 10. Configurations of the 3 elements of the deep kernel mapping network.

To Optimize the Elements	Basic Accounting Sub	Network Node	Network Structure
Elements set	Edmond–Karp nuclear Markov nuclear	Accounting for sublinear combinations Accounting for subnonlinear combinations	4-layer network 5-layer network 6-layer network

Basic Edmond–Karp accounting (EK): Optimal results obtained by using POLY1, POLY2, RBF1, and RBF2 as basic accounting subsets.

Basic Markov calculators (MK): Optimal results obtained by using the kernel function of Markov distance, including M_POLY1, M_POLY2, M_RBF1, and M_RBF2.

Quasiconformal mapping Markov calculators (QMK): Quasiconformal mapping is used to obtain the optimized calculator, including QM_POLY1, QM_POLY2, QM_RBF1, and QM_RBF2.

QMK and linear calculation 1 (LC1): Calculation method combining quasiconformal mapping Markovner calculations and the linear combined network node calculation method based on Bregman divergence.

QMK and linear calculation 2 (LC2): Method including node calculation of a linear combined network based on the maximum threshold function by using the quasiconformal mapping Markov calculation.

QMK and nonlinear calculation (NonLC): Calculation method adopting kernel operators and nonlinear nodes.

QMK, NonLC, and network structure calculation: Results obtained by the methods of accounting suboptimization, nonlinear network node calculation, and adaptive network structure calculation.

All the above algorithms are simulated with the same computing resources, and all the training samples and test samples are allocated under the same conditions.

As shown in Tables A12–A16 (in Appendix A), the accuracy evaluation on the same dataset shows that the performance of classification increases progressively in the order of operator optimization, operator optimization, network node optimization, and triple optimization. Multiple optimization can improve the performance of the algorithm.

(3) Comparative validation experiment to determine the multi-optimization network performance

In this experiment, the performance analysis of the algorithm based on multiple optimization of the kernel operator, mapping network nodes, and network structure is compared with the performance analysis of the existing classification algorithm. At present, the Indian Pines, University of Pavia, Salinas, Houston 2013, and Houston 2018 datasets are mainly used to evaluate classification performance by using classification accuracy. In this experimental evaluation, the principal component analysis method is adopted to achieve feature dimension reduction in the original data. In the experiment, the characteristic dimensions of the first four datasets are from 10 to 50, and the total number of spectral bands of the fifth dataset is 48; thus, the highest dimension of the fifth dataset is set to 45.

The focus of this part of the experiment is to compare the advancement of existing methods and evaluate the advantage in precision. Some experiments are carried out to compare the performance of the algorithm. EasyMKL [24], SimpleMKL [25], SM1MKL [34], L2MKL [35], RBF-SKL [28], POLY-SKL [28], Euclidean-MKL1 [29], Euclidean-MKL2 [30], SK-CV-RBF-SKL [31], SK-POLY-SKL [32], Mahal-RBF-SKL [32], Mahal-Poly-SKL [32], NMF-MKL [33], KNMF-MKL [33], QMKL [36], DMKL [37], SparseDKL [38], and HessianMKL [39], i.e., 18 multikernel learning and deep multikernel learning classification algorithms, are implemented. The basic ideas of these algorithms are as follows:

EasyMKL [24]: A multikernel network algorithm using direct multikernel function addition calculation;

SimpleMKL [25]: A multikernel network algorithm with different weights;

SM1MKL [34]: A multikernel combination calculation method based on a soft margin is adopted;

L2MKL [35]: A multikernel combined network computing method using the L2 norm;

RBF-SKL [28]: A method in which the RBF Euclidian kernel is used as the kernel function of the support vector machine;

POLY-SKL [28]: A method in which the polynomial Euclidean kernel used as the kernel function of the support vector machine;

Euclidean-MKL1 [29]: A calculation method of nodes with the same weight combination;

Euclidean-MKL2 [30]: A node calculation method with different weights;

SK-CV-RBF-SKL [31]: A method in which the RBF kernel is used as the kernel function of the support vector machine;

SK-POLY-SKL [31]: A method in which the polynomial kernel is used as the kernel function of the support vector machine;

Mahal-RBF-SKL [32]: A method in which the RBF kernel based on Mahalanobis distance is used as the kernel function of SVM;

Mahal-Poly-SKL [32]: A method in which the polynomial kernel based on the Mahalanobis distance is used as the kernel function of the support vector machine;

NMF-MKL [33]: Non-negative matrix factorization (NMF) MKL;

KNMF-MKL [33]: A kernel-based non-negative matrix combining the multikernel and KNMF methods;

QMKL [36]: A method using a generic mapping kernel function multikernel mapping learning network;

DMKL [37]: A deep kernel learning network based on generic mapping;

SparseDKL [38]: A deep kernel learning matrix network based on the sparse matrix principle;

HessianMKL [39]: A method using the SimpleMKL node calculation.

The experimental results are shown in Tables 11 and A17–A21 and in Figures 3 and A5–A8 (Tables A17–A21 and Figures A5–A8 are in Appendix A). The kappa coefficient (KC), overall accuracy (OA) and average classification accuracy (AA) were used to evaluate the performance of the algorithm. The multi-optimization method proposed in this paper is a combination of network structure optimization, network node optimization, and operator optimization.

Table 11. Performance comparison of the different algorithms on the Indian Pines dataset.

Serial Number	Methods	AA (%)	OA (%)	KC (%)
1	EasyMKL [24]	89.47	88.83	84.86
2	SimpleMKL [25]	90.45	89.26	85.56
3	SM1MKL [34]	91.79	90.42	86.57
4	L2MKL [35]	92.74	91.35	88.43
5	RBF-SKL [28]	87.23	86.43	82.34
6	POLY-SKL [28]	88.47	87.62	83.56
7	Euclidean-MKL1 [29]	91.64	90.34	88.27
8	Euclidean-MKL2 [30]	92.45	91.47	87.57
9	SK-CV-RBF-SKL [31]	92.24	91.45	87.35
10	SK-POLY-SKL [31]	91.46	90.37	88.34
11	Mahal-RBF-SKL [32]	91.94	90.67	88.32
12	Mahal-Poly-SKL [32]	92.39	91.84	87.03
13	NMF-MKL [33]	91.89	90.92	86.97
14	KNMF-MKL [33]	92.68	91.26	87.47
15	QMKL [36]	94.83	93.92	89.94
16	DMKL [37]	95.68	96.37	91.34
17	SparseDKL [38]	95.54	94.68	90.27
18	HessianMKL [39]	95.83	94.90	90.78
19	In this paper, methods	96.76	95.94	91.82

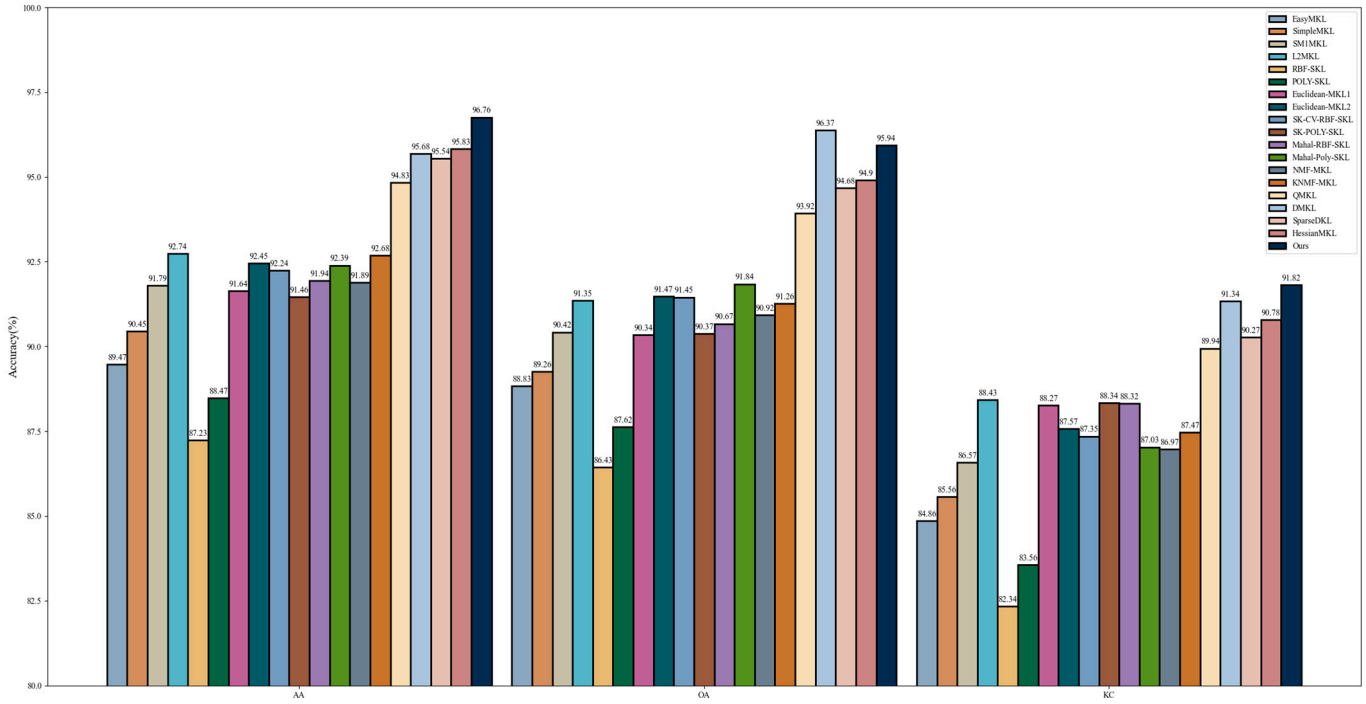


Figure 3. Performance comparison of the different algorithms on the Indian Pines dataset.

In summary, the experimental results show that the deep network is improved from three perspectives, namely the kernel operator, mapping network node, and network structure, and this method effectively improves classification results. The experimental results show that in the application of classification, most of the proposed algorithms are improved from a single operator and related methods, including optimization of the kernel operator and network node calculation. To solve the basic kernel operator problem of a deep mapping network, combined with the application problem of a strong coupling degree between features and high band correlation, the kernel operator optimization method is proposed from the perspective of the internal construction of the kernel operator, and a Mahalanobis distance metric matrix is constructed. This method maps samples to feature space with stronger classification ability, which can enhance the feature relationship of similar samples, and optimize the features used for feature classification. Considering the network node optimization problem of a deep mapping network, the strong interspectral nonlinearity of the kernel operator structure expression is not sufficiently accurate. From the perspective of network node accounting subcombination optimization, this paper effectively improves the adaptability of the learning model to complex data and can more accurately describe the data from the input space to nonlinear mapping relationships so that data belonging to different classes have better discrimination in the nonlinear mapping space.

Considering the network node optimization problem of a deep mapping network, the strong interspectral nonlinearity and single accounting substructure expressions are not sufficiently accurate. In this paper, from the perspective of network node accounting subcombination optimization, the adaptive ability of the learning model to complex data can be improved effectively, and the data from the input space to the nonlinear mapping relationship can be described more accurately so that the data belonging to different classes can have better discrimination in the nonlinear mapping space. Considering the insufficient application of the existing network structure to line feature learning, the proposed deep kernel mapping network optimization based on structure adaptation effectively improves the ability to accurately describe spectral relations and improves the classification performance to obtain features that can more accurately describe the target characteristics.

4. Conclusions

In this paper, a structural adaptive deep kernel mapping network optimization method is proposed from the perspective of the adaptive optimization of the network structure. This method allows the network to adapt its structure based on the data's distribution characteristics, thereby improving image classification performance. On this basis, the classification method based on a multi-optimized deep kernel mapping network is proposed. By combining deep kernel mapping networks with quasiconformal kernel learning, this method optimizes the network structure and enhances both the accuracy and efficiency of image classification. The feasibility of network optimization is verified by the open simulation datasets and other datasets, and then the optimization method proposed in this paper is comprehensively verified and analyzed. The results show that the deep kernel mapping network, which is optimized from three aspects, namely the kernel operator, mapping network node, and network structure, can effectively improve the feature extraction and classification performance of data. However, the adaptive network structure and deep kernel mapping in DQKNet may increase model complexity, potentially making the training process more difficult and requiring additional parameters and computational resources. In the future, we aim to address these challenges by refining the method.

Author Contributions: Conceptualization, J.Z.; Data curation, Z.W.; Funding acquisition, J.Z.; Methodology, J.Z.; Formal analysis, Z.Z.; Validation, F.Y.; Writing—original draft, J.Z.; Writing—review and editing, D.G. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The raw data supporting the conclusions of this article will be made available by the authors on request.

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

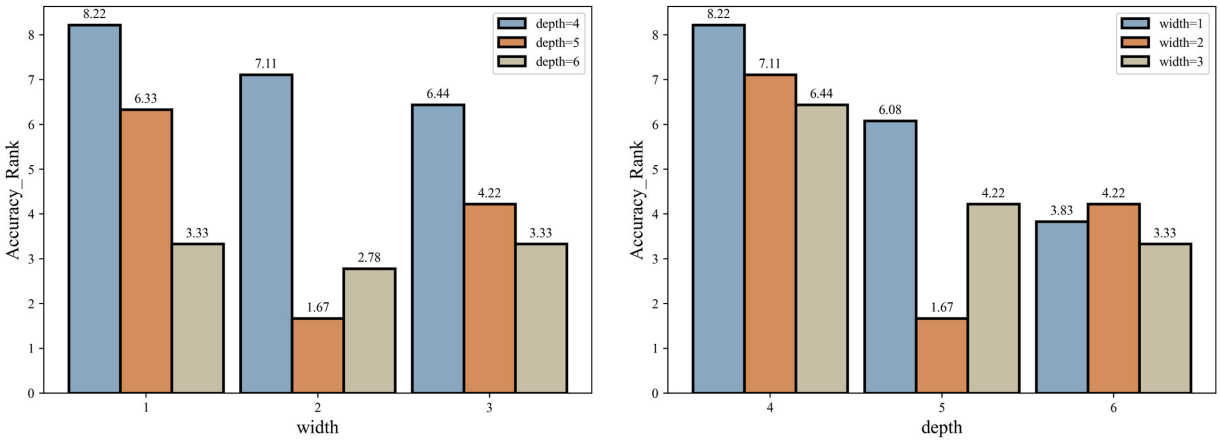
Appendix A

Table A1. Classification accuracy with different network structures on the Pavias dataset (%).

Feature Category	Depth of the Structure	Width of the Structure		
		1	2	3
C_1—Tarmac	4	79.47 (8)	81.22 (7)	84.83 (5)
	5	83.28 (6)	**87.99 (1)**	85.60 (4)
	6	86.79 (3)	87.60 (2)	86.79 (3)
C_2—Grass	4	82.11 (9)	84.27 (8)	87.49 (4)
	5	86.27 (6)	89.06 (2)	85.89 (7)
	6	88.37 (3)	**89.26 (1)**	87.33 (5)
C_3—Gravel roof	4	81.21 (8)	86.46 (7)	86.46 (7)
	5	88.17 (5)	**91.95 (1)**	87.33 (6)
	6	90.50 (2)	89.20 (4)	89.49 (3)
C_4—Trees	4	82.87 (8)	85.50 (7)	85.65 (6)
	5	85.50 (7)	88.63 (3)	89.87 (2)
	6	87.59 (4)	87.01 (5)	**90.05 (1)**
C_5—Metal roof	4	86.38 (7)	86.09 (8)	88.47 (6)
	5	85.56 (9)	91.68 (2)	91.24 (3)
	6	**92.28 (1)**	90.57 (4)	89.67 (5)
C_6—Bare land	4	83.05 (8)	86.77 (5)	84.19 (7)
	5	85.61 (6)	**89.98 (1)**	86.77 (5)
	6	89.59 (2)	88.70 (3)	88.03 (4)
C_7—Asphalt roof	4	84.54 (8)	88.21 (7)	83.10 (9)
	5	89.47 (6)	**92.89 (1)**	90.59 (5)
	6	91.23 (4)	92.65 (2)	91.88 (3)
C_8—Brick pavement	4	84.10 (9)	86.57 (7)	86.45 (8)
	5	88.69 (5)	90.08 (3)	90.46 (2)
	6	87.58 (6)	**90.70 (1)**	89.42 (4)
C_9—Shadow	4	83.21 (9)	85.37 (8)	87.49 (6)
	5	86.50 (7)	**90.72 (1)**	89.45 (4)
	6	88.74 (5)	90.00 (3)	90.08 (2)
The average ranking		5.96	**3.85**	4.67

Table A2. Rank of classification accuracies with different network structures on the Pavias dataset.

(Depth, Width)	1	2	3
4	8.22	7.11	6.44
5	6.33	**1.67**	4.22
6	3.33	2.78	3.33



(a) **Classification Accuracy With Different Depth On The Pavias Dataset** (b) **Classification Accuracy With Different Width On The Pavias Dataset**

Figure A1. Rank of classification accuracies with different widths and depths on the Pavias dataset.

Table A3. Classification accuracy with different network structures on the Salinas dataset (%).

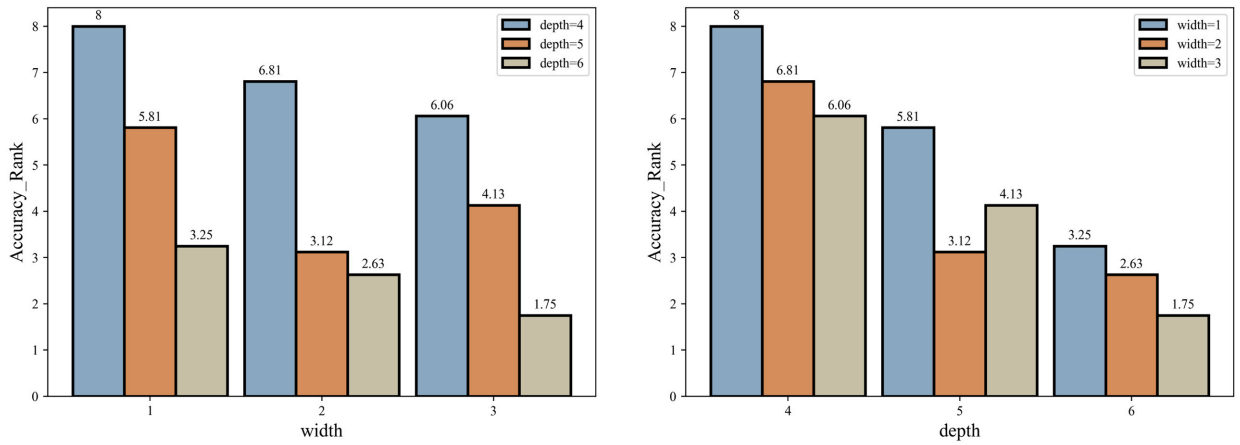
Feature Category	Depth of the Structure	Width of the Structure		
		1	2	3
Brocoli-green-weeds-1	4	82.90 (9)	87.43 (6)	86, 58 (8)
	5	86.79 (7)	91.78 (3)	89.97 (5)
	6	90.96 (4)	92.56 (2)	**93.25 (1)**
Brocoli-green-weeds-2	4	84.75 (8)	84.20 (9)	90.06 (6)
	5	88.65 (7)	90.82 (4)	90.36 (5)
	6	91.57 (2)	91.43 (3)	**92.78 (1)**
Fallow	4	82.37 (8)	86.34 (7)	89.43 (5)
	5	88.27 (6)	91.05 (2)	90.03 (4)
	6	89.43 (5)	**91.10 (1)**	90.72 (3)
Fallow-rough-plow	4	83.49 (7)	89.27 (6)	89.27 (6)
	5	90.20 (5)	92.17 (2)	91.43 (4)
	6	89.27 (6)	91.78 (3)	**92.48 (1)**
Fallow-smooth	4	82.24 (8)	86.59 (7)	88.45 (6)
	5	89.87 (5)	**91.86 (1)**	90.62 (4)
	6	90.62 (4)	91.09 (3)	91.77 (2)
Stubble	4	83.50 (8)	85.92 (7)	88.26 (6)
	5	89.69 (5)	90.04 (4)	85.30 (5)
	6	89.69 (5)	**91.64 (1)**	91.60 (2)
Celery	4	84.16 (9)	88.76 (8)	90.10 (6)
	5	89.14 (7)	91.56 (4)	91.39 (5)
	6	92.30 (2)	92.12 (3)	**92.88 (1)**
Grapes-untrained	4	82.91 (7)	87.64 (6)	89.02 (5)
	5	87.64 (6)	**91.48 (1)**	91.36 (2)
	6	91.06 (3)	90.84 (4)	90.84 (4)

Table A4. Rank of classification accuracies with different network structures on the Salinas dataset (%).

Feature Category	Depth of the Structure	Width of the Structure		
		1	**2**	**3**
Soil-vineyard-develop	4	82.90 (8)	88.44 (6)	87.71 (7)
	5	87.71 (7)	88.89 (5)	91.24 (4)
	6	92.39 (2)	92.11 (3)	**93.00 (1)**
Corn-senesced-green-weeds	4	81.87 (9)	87.89 (7)	87.44 (8)
	5	88.67 (6)	92.24 (2)	90.94 (4)
	6	**92.43 (1)**	90.64 (5)	91.68 (3)
Lettuce-romaine-4wk	4	83.11 (8)	86.78 (6)	85.98 (7)
	5	88.45 (5)	90.47 (2)	89.28 (4)
	6	88.45 (5)	90.09 (3)	**90.84 (1)**
Lettuce-romaine-5wk	4	81.87 (8)	86.22 (7)	88.24 (6)
	5	89.69 (5)	90.06 (4)	88.24 (6)
	6	**91.05 (1)**	90.47 (3)	90.78 (2)
Lettuce-romaine-6wk	4	82.35 (7)	86.39 (6)	88.10 (5)
	5	86.39 (6)	90.47 (3)	89.69 (4)
	6	91.26 (2)	89.69 (4)	**91.38 (1)**
Lettuce-romaine-7wk	4	83.52 (8)	84.40 (7)	87.69 (5)
	5	90.47 (3)	87.44 (6)	91.57 (2)
	6	90.06 (4)	**92.22 (1)**	91.57 (2)
Vinyeard-untrained	4	80.87 (8)	85.38 (7)	86.00 (5)
	5	85.67 (6)	88.86 (2)	86.12 (4)
	6	87.47 (3)	88.86 (2)	**89.57 (1)**
Vineyard-vertical-trellis	4	82.10 (8)	85.48 (7)	87.56 (6)
	5	85.48 (7)	88.69 (5)	89.68 (4)
	6	91.00 (3)	**91.38 (1)**	91.24 (2)
The average ranking		5.69	4.19	**3.98**

Table A5. Rank of classification accuracies with different network structures on the Salinas dataset.

(Depth, Width)	1	2	3
4	8.00	6.81	6.06
5	5.81	3.12	4.13
6	3.25	2.63	**1.75**



(a) **Classification Accuracy With Different Depth On The Salinas Dataset** (b) **Classification Accuracy With Different Width On The Salinas Dataset**

Figure A2. Rank of classification accuracies with different widths and depths on the Salinas dataset.

Table A6. Classification accuracy with different network structures on the Houston 2013 dataset (%).

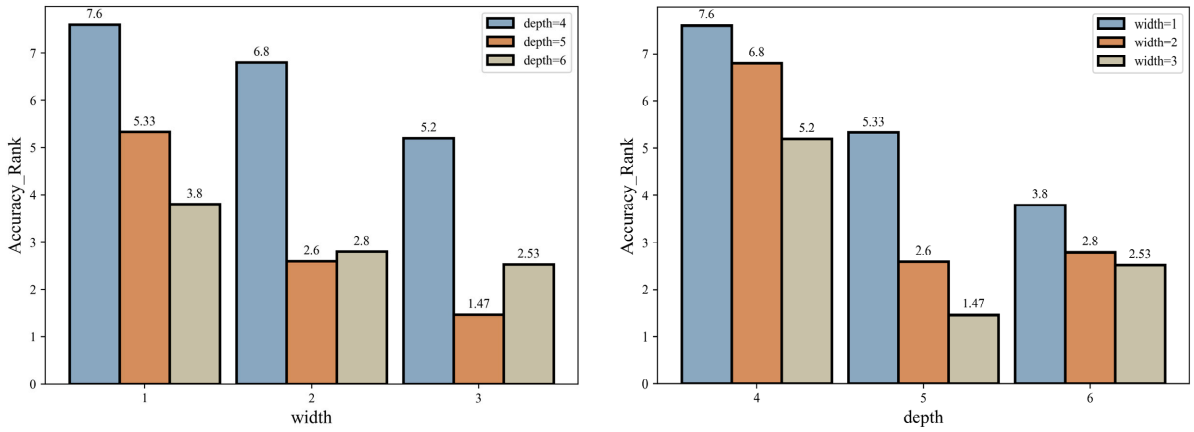
Feature Category	Depth of the Structure	Width of the Structure		
		1	**2**	**3**
Healthy grass	4	75.22 (8)	77.12 (7)	77.57 (6)
	5	78.54 (5)	79.69 (3)	**80.24 (1)**
	6	78.54 (5)	84.60 (2)	79.42 (4)
Stressed grass	4	78.35 (8)	80.00 (7)	80.27 (6)
	5	81.46 (4)	82.37 (3)	**83.23 (1)**
	6	81.28 (5)	80.27 (6)	82.85 (2)
Synthetic grass	4	80.84 (8)	84.79 (7)	86.00 (4)
	5	85.18 (6)	86.25 (3)	**87.28 (1)**
	6	85.67 (5)	**87.28 (1)**	86.46 (2)
Trees	4	78.88 (8)	81.59 (7)	81.59 (7)
	5	82.30 (6)	85.28 (2)	84.78 (3)
	6	82.44 (5)	83.60 (4)	**85.59 (1)**
Soil	4	82.10 (7)	85.41 (6)	87.34 (5)
	5	87.34 (5)	87.34 (5)	**89.47 (1)**
	6	88.49 (3)	89.17 (2)	88.06 (4)
Water	4	82.06 (8)	84.01 (7)	84.20 (6)
	5	84.89 (5)	86.36 (4)	87.00 (3)
	6	87.57 (2)	**88.20 (1)**	86.36 (4)
Residential	4	78.60 (8)	79.37 (7)	81.68 (4)
	5	80.09 (6)	83.24 (2)	**83.66 (1)**
	6	81.55 (5)	83.24 (2)	82.07 (3)
Commercial	4	70.18 (8)	72.45 (7)	72.59 (6)
	5	73.38 (5)	75.00 (3)	75.27 (2)
	6	74.75 (4)	73.38 (5)	**75.60 (1)**

Table A7. Classification accuracy with different network structures on the Houston 2013 dataset (%).

Feature Category	Depth of the Structure	Width of the Structure		
		1	**2**	**3**
Road	4	71.68 (8)	72.09 (7)	74.35 (4)
	5	75.22 (5)	**80.04 (1)**	80.00 (2)
	6	76.93 (4)	78.64 (3)	78.64 (3)
Highway	4	58.47 (4)	64.09 (6)	66.05 (4)
	5	65.53 (5)	68.22 (2)	**68.65 (1)**
	6	66.43 (3)	68.22 (2)	68.22 (2)
Railway	4	75.69 (8)	79.83 (7)	80.00 (6)
	5	80.67 (5)	81.47 (2)	**81.85 (1)**
	6	81.25 (3)	**81.85 (1)**	81.20 (4)
Parking lot 1	4	79.43 (8)	81.22 (7)	82.74 (5)
	5	81.46 (6)	84.59 (2)	**85.00 (1)**
	6	83.88 (3)	83.07 (4)	**85.00 (1)**
Parking lot 2	4	80.00 (8)	81.11 (7)	81.11 (7)
	5	82.30 (6)	85.11 (2)	**85.23 (1)**
	6	83.36 (4)	82.46 (5)	84.34 (3)
Tennis court	4	81.02 (7)	83.77 (6)	86.53 (3)
	5	84.24 (5)	85.90 (4)	**88.24 (1)**
	6	86.77 (2)	**88.24 (1)**	86.53 (3)
Running track	4	79.36 (8)	83.49 (7)	84.79 (5)
	5	84.48 (6)	**88.03 (1)**	87.42 (2)
	6	86.14 (4)	87.00 (3)	**88.03 (1)**
The average ranking		5.58	4.07	**3.07**

Table A8. Rank of classification accuracies with different network structures on the Houston 2013 dataset.

(Depth, Width)	1	2	3
4	7.60	6.80	5.20
5	5.33	2.60	**1.47**
6	3.80	2.80	2.53



(a) **Classification Accuracy With Different Depth On The Houston 2013 Dataset** (b) **Classification Accuracy With Different Width On The Houston 2013 Dataset**

Figure A3. Rank of classification accuracies with different widths and depths on the Houston 2013 dataset.

Table A9. Classification accuracy with different network structures on the Houston 2018 dataset (%).

Feature Category	Depth of the Structure	Width of the Structure		
		1	2	3
Healthy grass	4	80.91 (8)	83.58 (7)	83.58 (7)
	5	84.22 (6)	84.57 (5)	86.43 (3)
	6	**87.21 (1)**	86.78 (2)	85.60 (4)
Stressed grass	4	76.48 (7)	79.24 (6)	79.24 (6)
	5	80.00 (5)	81.57 (2)	80.08 (4)
	6	**81.98 (1)**	**81.98 (1)**	81.26 (3)
Artificial turf	4	84.22 (9)	88.34 (6)	86.80 (8)
	5	87.25 (7)	91.29 (3)	90.05 (5)
	6	91.75 (2)	**91.98 (1)**	90.73 (4)
Evergreen trees	4	77.77 (8)	81.70 (7)	83.20 (5)
	5	82.49 (6)	**86.77 (1)**	84.59 (4)
	6	84.59 (4)	86.45 (2)	85.38 (3)
Deciduous trees	4	70.54 (8)	74.70 (7)	75.46 (6)
	5	76.30 (5)	76.30 (5)	77.57 (4)
	6	**79.30 (1)**	78.29 (3)	78.56 (2)
Bare earth	4	83.77 (8)	86.94 (7)	87.48 (6)
	5	88.69 (5)	**92.11 (1)**	91.25 (3)
	6	91.84 (2)	91.25 (3)	89.47 (4)
Water	4	81.60 (9)	88.54 (6)	86.96 (8)
	5	87.75 (7)	91.68 (2)	90.73 (5)
	6	**92.42 (1)**	91.48 (3)	91.08 (4)
Residential buildings	4	60.41 (8)	64.73 (7)	67.43 (6)
	5	64.73 (7)	68.99 (4)	68.57 (5)
	6	70.37 (2)	**71.40 (1)**	69.58 (3)

Table A9. *Cont.*

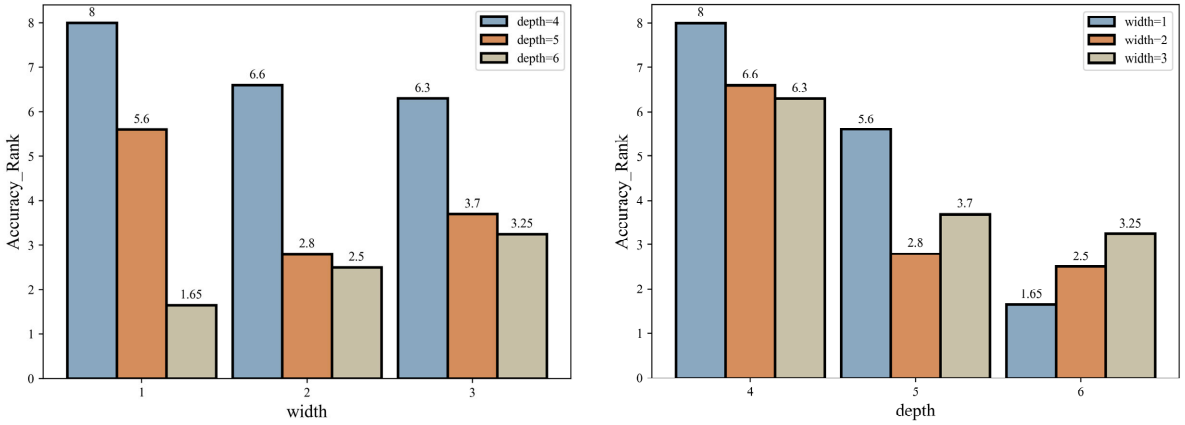
Feature Category	Depth of the Structure	Width of the Structure		
		1	**2**	**3**
Nonresidential buildings	4	54.10 (8)	58.32 (7)	60.73 (6)
	5	60.94 (5)	62.48 (2)	61.28 (4)
	6	**62.74 (1)**	61.66 (3)	62.48 (2)
Roads	4	28.49 (7)	31.27 (5)	31.00 (6)
	5	31.27 (5)	32.44 (2)	31.33 (4)
	6	32.44 (2)	31.69 (3)	**32.46 (1)**

Table A10. Classification accuracy with different network structures on the Houston 2018 dataset (%).

Feature Category	Depth of the Structure	Width of the Structure		
		1	**2**	**3**
Sidewalks	4	328.89 (8)	42.19 (7)	43.07 (6)
	5	43.20 (5)	46.44 (2)	45.39 (3)
	6	**46.95 (1)**	44.68 (4)	44.68 (4)
Crosswalks	4	56.31 (8)	60.00 (7)	61.74 (5)
	5	60.45 (6)	62.06 (4)	**63.92 (1)**
	6	62.78 (3)	63.33 (2)	61.74 (5)
Major thoroughfares	4	40.26 (8)	44.39 (7)	45.22 (6)
	5	46.05 (5)	47.28 (3)	**48.58 (1)**
	6	**48.58 (1)**	46.85 (4)	47.66 (2)
Highways	4	71.70 (8)	76.32 (6)	75.40 (7)
	5	76.32 (6)	76.97 (5)	77.91 (4)
	6	79.47 (2)	78.80 (3)	**80.40 (1)**
Railways	4	81.20 (8)	85.39 (6)	84.24 (7)
	5	87.08 (5)	90.00 (2)	87.93 (4)
	6	**90.64 (1)**	88.74 (3)	87.08 (5)
Paved parking lots	4	73.35 (9)	79.21 (7)	78.25 (8)
	5	80.00 (6)	**82.90 (1)**	80.63 (5)
	6	81.47 (3)	82.12 (2)	81.30 (4)
Unpaved parking lots	4	82.42 (9)	86.02 (8)	86.49 (7)
	5	88.36 (6)	89.47 (5)	90.08 (4)
	6	**92.40 (1)**	91.48 (2)	90.76 (3)
Cars	4	76.41 (7)	81.70 (6)	82.40 (5)
	5	82.71 (4)	85.76 (2)	84.40 (3)
	6	**86.37 (1)**	84.40 (3)	84.40 (3)
Trains	4	78.81 (8)	84.70 (7)	84.74 (6)
	5	86.38 (5)	**90.06 (1)**	88.25 (3)
	6	89.76 (2)	87.40 (4)	86.38 (5)
Stadium seats	4	80.01 (7)	84.57 (6)	85.99 (5)
	5	84.57 (6)	87.19 (4)	88.69 (2)
	6	**90.00 (1)**	**90.00 (1)**	88.02 (3)
The average ranking		5.55	4.08	4.67

Table A11. Rank of classification accuracies with different network structures on the Houston 2018 dataset.

(Depth, Width)	1	2	3
4	8.00	6.60	6.30
5	5.60	2.80	3.70
6	**1.65**	2.50	3.25



(a) **Classification Accuracy With Different Depth On The Houston 2018 Dataset** (b) **Classification Accuracy With Different Width On The Houston 2018 Dataset**

Figure A4. Rank of classification accuracies with different widths and depths on the Houston 2018 dataset.

Table A12. Feasibility analysis of different optimization strategies on the Indian Pines dataset.

Multiple Optimization Strategy Combinations	Algorithm Name	Feature Extraction Dimension				
		10	20	30	40	50
Operator optimization	Basic Edmond–Karp accounting (EK)	69.33	75.40	80.19	83.41	85.01
	Basic Markov calculation (MK)	71.34	77.23	82.56	85.67	87.45
	Quasiconformal mapping Markov calculators (QMK)	73.12	79.81	84.76	87.62	89.19
Operator optimization and node optimization	QMK and linear node calculation 1 (LC1)	73.24	78.57	82.35	85.78	88.25
	QMK and linear node calculation 2 (LC2)	75.82	82.47	86.72	88.28	91.45
	QMK and nonlinear node computation (NonLC)	77.46	84.75	88.39	90.79	93.28
Operator, node, and structure optimization	QMK, NonLC, and network structure calculation	80.39	86.94	90.95	93.69	95.94

Table A13. Feasibility analysis of different optimization strategies on the University of Pavia dataset (%).

Multiple Optimization Strategy Combinations	Algorithm Name	Feature Extraction Dimension				
		10	20	30	40	50
Operator optimization	Basic Edmond–Karp accounting (EK)	72.86	77.99	82.45	86.54	88.14
	Basic Markov calculation (MK)	73.78	78.92	83.21	87.82	90.11
	Quasiconformal mapping Markov Calculators (QMK)	76.21	80.22	85.89	89.99	92.67
Operator optimization and node optimization	QMK and linear node calculation 1 (LC1)	76.23	82.08	86.26	89.14	90.25
	QMK and linear node calculation 2 (LC2)	78.35	83.35	87.57	92.58	93.47
	QMK and nonlinear node computation (NonLC)	80.47	85.22	89.64	94.34	95.75
Operator, node, and structure optimization	QMK, NonLC, and network structure calculation	83.53	88.56	92.32	95.63	96.68

Table A14. Feasibility analysis of different optimization strategies on the Salinas dataset (%).

Multiple Optimization Strategy Combinations	Algorithm Name	Feature Extraction Dimension				
		10	20	30	40	50
Operator optimization	Basic Edmond–Karp accounting (EK)	68.23	76.34	81.25	84.98	86.36
	Basic Markov calculation (MK)	70.62	78.35	83.39	86.93	88.84
	Quasiconformal mapping Markov calculators (QMK)	72.68	80.46	85.67	88.47	90.78
Operator optimization and node optimization	QMK and linear node calculation 1 (LC1)	73.56	79.37	82.86	86.28	87.32
	QMK and linear node calculation 2 (LC2)	75.57	82.75	87.57	89.32	91.25
	QMK and nonlinear node computation (NonLC)	76.43	85.93	88.79	90.51	93.94
Operator, nnode, and structure optimization	QMK, NonLC, and network structure calculation	78.40	88.89	90.62	92.89	95.78

Table A15. Feasibility analysis of different optimization strategies on the Houston 2013 dataset (%).

Multiple Optimization Strategy Combinations	Algorithm Name	Feature Extraction Dimension				
		10	20	30	40	50
Operator optimization	Basic Edmond–Karp accounting (EK)	62.86	67.89	73.65	76.74	78.45
	Basic Markov calculation (MK)	64.84	69.22	75.57	78.57	80.65
	Quasiconformal mapping Markov calculators (QMK)	66.67	71.43	77.57	80.65	82.32
Operator optimization and node optimization	QMK and linear node calculation 1 (LC1)	67.46	72.36	77.39	80.93	81.42
	QMK and linear node calculation 2 (LC2)	72.36	74.85	79.27	83.69	84.37
	QMK and nonlinear node computation (NonLC)	74.75	76.45	81.83	85.45	86.38
Operator, node, and structure optimization	QMK, NonLC, and network structure calculation	76.93	75.93	83.92	87.93	88.93

Table A16. Feasibility analysis of different optimization strategies on the Houston 2018 dataset (%).

Multiple Optimization Strategy Combinations	Algorithm Name	Feature Extraction Dimension				
		10	20	30	40	45
Operator optimization	Basic Edmond–Karp accounting (EK)	55.28	57.44	62.85	67.89	72.47
	Basic Markov calculation (MK)	56.43	58.54	63.86	68.76	73.65
	Quasiconformal mapping Markov calculators (QMK)	57.45	59.57	64.78	69.69	74.68
Operator optimization and node optimization	QMK and linear node calculation 1 (LC1)	59.68	60.63	65.74	70.25	72.38
	QMK and linear node calculation 2 (LC2)	62.26	64.84	67.27	73.54	75.76
	QMK and nonlinear node computation (NonLC)	64.68	65.53	68.39	75.93	77.27
Operator, node, and structure optimization	QMK, NonLC, and network structure calculation	64.93	65.93	69.92	76.93	78.93

Table A17. Performance comparison of the different algorithms on the University of Pavia dataset.

Serial Number	Methods	AA (%)	OA (%)	KC (%)
1	EasyMKL [24]	90.43	90.56	85.45
2	SimpleMKL [25]	91.45	90.57	86.83
3	SM1MKL [34]	92.59	92.43	87.67
4	L2MKL [35]	93.57	92.49	87.64

Table A17. *Cont.*

Serial Number	Methods	AA (%)	OA (%)	KC (%)
5	RBF-SKL [28]	88.34	88.63	83.25
6	POLY-SKL [28]	89.43	88.43	84.75
7	Euclidean-MKL1 [29]	92.35	92.32	88.83
8	Euclidean-MKL2 [30]	93.86	92.78	87.45
9	SK-CV-RBF-SKL [31]	93.24	92.45	87.38
10	SK-POLY-SKL [31]	91.93	91.03	88.69
11	Mahal-RBF-SKL [32]	92.36	92.25	88.75
12	Mahal-Poly-SKL [32]	93.49	92.82	87.25
13	NMF-MKL [33]	92.79	92.56	87.73
14	KNMF-MKL [33]	93.45	92.72	87.83
15	QMKL [36]	95.74	94.96	88.85
16	DMKL [37]	96.83	96.32	89.74
17	SparseDKL [38]	96.35	95.37	89.28
18	HessianMKL [39]	96.36	95.98	89.96
19	In this paper, methods	97.42	96.68	90.97

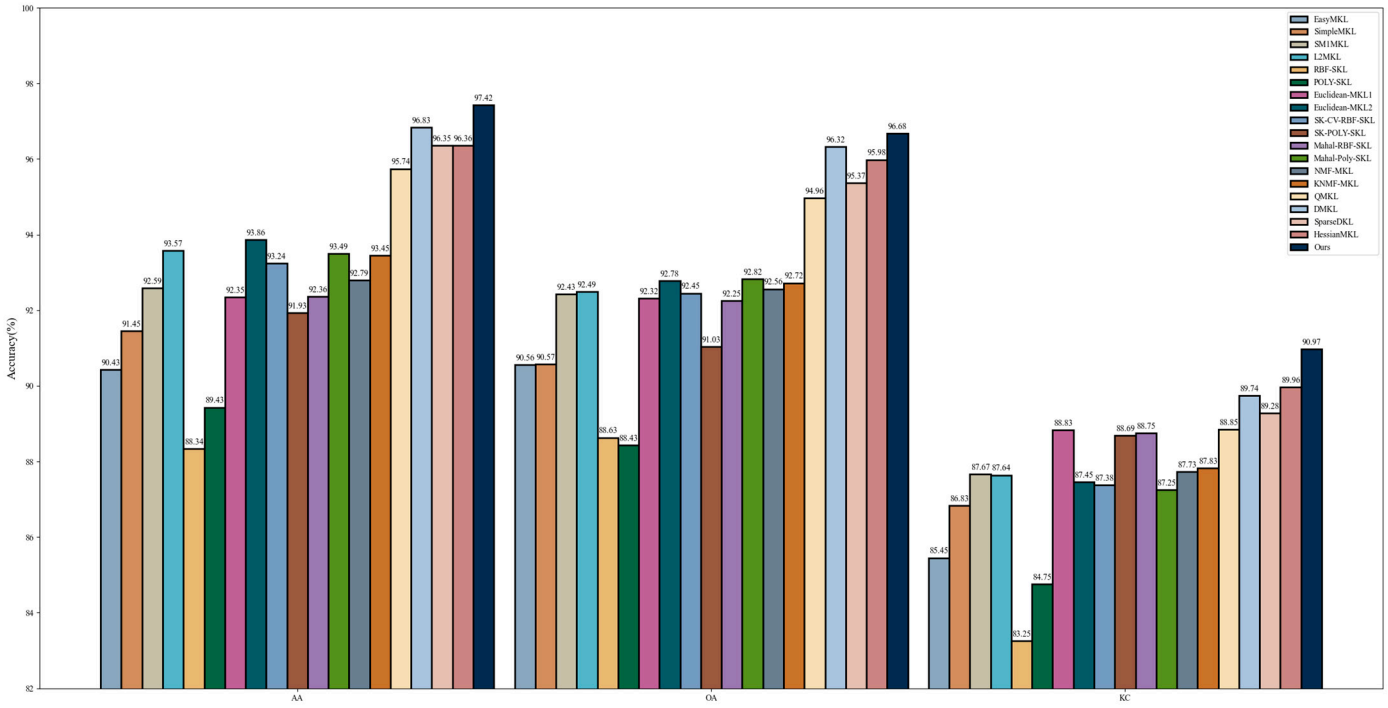


Figure A5. Performance comparison of the different algorithms on the University of Pavia dataset.

Table A18. Performance comparison of the different algorithms on the Salinas dataset.

Serial Number	Methods	AA (%)	OA (%)	KC (%)
1	EasyMKL [24]	90.10	89.15	83.53
2	SimpleMKL [25]	90.89	89.71	84.83
3	SM1MKL [34]	92.45	91.29	85.83
4	L2MKL [35]	92.46	91.54	86.45
5	RBF-SKL [28]	87.35	87.37	81.75
6	POLY-SKL [28]	88.86	87.43	83.34
7	Euclidean-MKL1 [29]	92.57	91.03	86.57
8	Euclidean-MKL2 [30]	92.82	91.32	86.56
9	SK-CV-RBF-SKL [31]	92.28	91.37	86.38

Table A19. Performance comparison of the different algorithms on the Salinas dataset.

Serial Number	Methods	AA (%)	OA (%)	KC (%)
10	SK-POLY-SKL [31]	91.37	90.56	87.86
11	Mahal-RBF-SKL [32]	92.93	91.63	86.66
12	Mahal-Poly-SKL [32]	92.32	91.80	86.30
13	NMF-MKL [33]	92.25	91.37	85.94
14	KNMF-MKL [33]	92.89	91.71	86.83
15	QMKL [36]	94.81	93.57	88.53
16	DMKL [37]	95.46	95.67	89.23
17	SparseDKL [38]	95.83	94.47	88.37
18	HessianMKL [39]	95.85	95.27	89.27
19	In this paper, methods	96.74	95.78	89.86

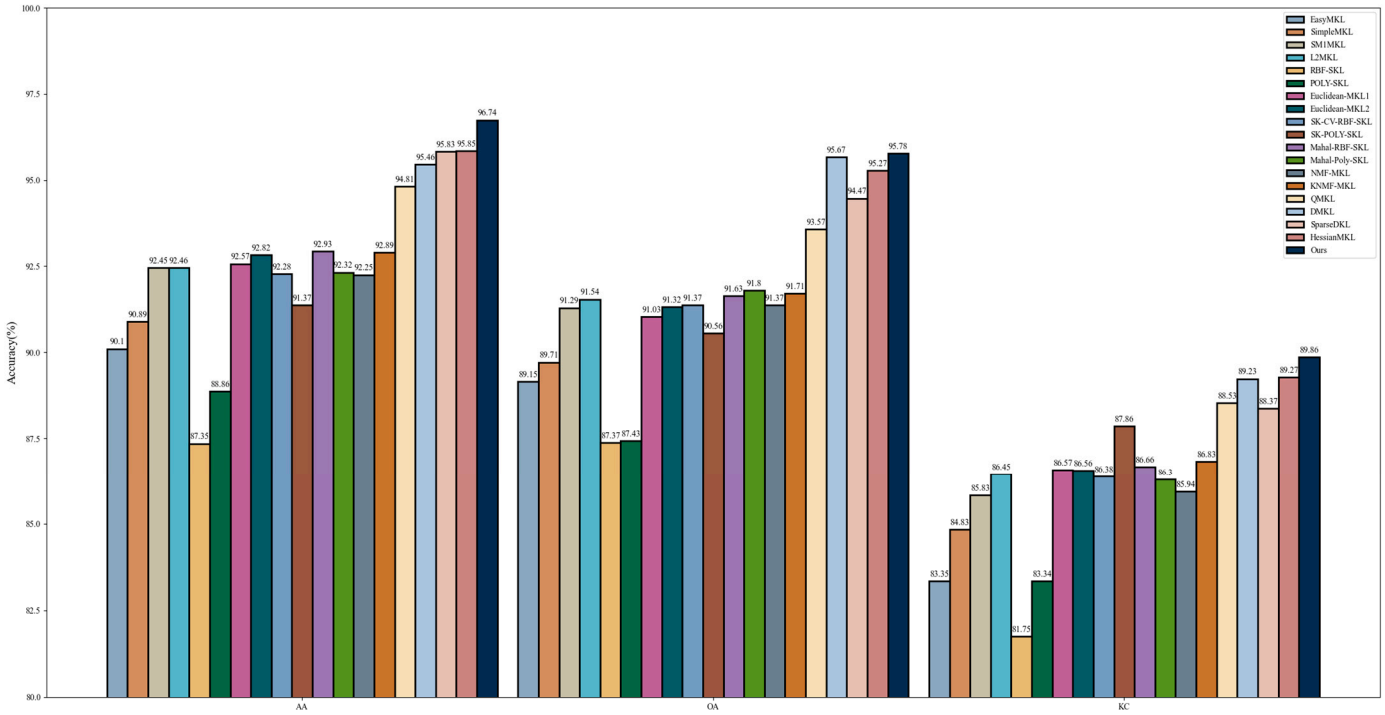


Figure A6. Performance comparison of the different algorithms on the Salinas dataset.

Table A20. Performance comparison of the different algorithms on the Houston 2013 dataset.

Serial Number	Methods	AA (%)	OA (%)	KC (%)
1	EasyMKL [24]	82.53	81.85	76.52
2	SimpleMKL [25]	83.48	82.52	77.87
3	SM1MKL [34]	84.83	83.82	78.23
4	L2MKL [35]	86.43	84.47	79.34
5	RBF-SKL [28]	80.64	80.69	76.43
6	POLY-SKL [28]	81.36	81.47	76.54
7	Euclidean-MKL1 [29]	84.54	83.80	79.32
8	Euclidean-MKL2 [30]	85.82	84.48	79.32
9	SK-CV-RBF-SKL [31]	85.47	84.49	79.24
10	SK-POLY-SKL [31]	84.98	83.82	78.74
11	Mahal-RBF-SKL [32]	84.59	83.51	79.59
12	Mahal-Poly-SKL [32]	85.32	84.79	79.94

Table A20. *Cont.*

Serial Number	Methods	AA (%)	OA (%)	KC (%)
13	NMF-MKL [33]	84.36	83.85	78.59
14	KNMF-MKL [33]	85.88	84.92	79.92
15	QMKL [36]	87.35	86.38	81.36
16	DMKL [37]	88.57	88.24	82.73
17	SparseDKL [38]	88.57	87.79	82.27
18	HessianMKL [39]	88.53	88.34	83.27
19	In this paper, methods	89.42	88.93	83.48

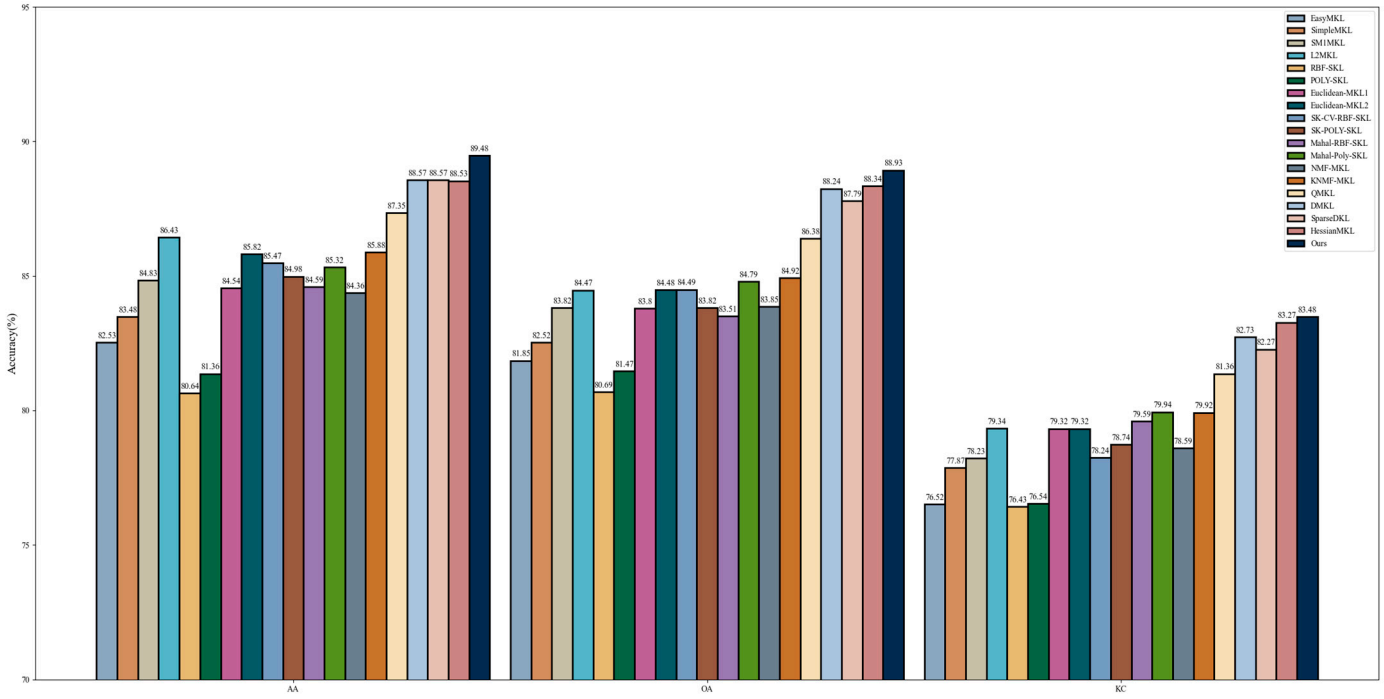


Figure A7. Performance comparison of the different algorithms on the Houston 2013 dataset.

Table A21. Performance comparison of the different algorithms on the Houston 2018 dataset.

Serial Number	Methods	AA (%)	OA (%)	KC (%)
1	EasyMKL [24]	72.43	72.12	68.24
2	SimpleMKL [25]	73.92	72.92	68.87
3	SM1MKL [34]	74.46	74.32	70.25
4	L2MKL [35]	75.57	74.58	71.45
5	RBF-SKL [28]	70.45	70.75	67.45
6	POLY-SKL [28]	71.72	70.43	67.75
7	Euclidean-MKL1 [29]	74.43	74.44	71.88
8	Euclidean-MKL2 [30]	75.23	74.86	70.24
9	SK-CV-RBF-SKL [31]	75.43	74.69	70.38
10	SK-POLY-SKL [31]	74.62	73.24	69.92
11	Mahal-RBF-SKL [32]	74.58	74.94	71.68
12	Mahal-Poly-SKL [32]	75.39	74.47	70.02
13	NMF-MKL [33]	74.89	74.68	70.24
14	KNMF-MKL [33]	75.57	74.92	70.87
15	QMKL [36]	77.35	76.27	71.59
16	DMKL [37]	78.47	78.25	73.13
17	SparseDKL [38]	78.57	77.35	72.73
18	HessianMKL [39]	78.25	78.34	73.78
19	In this paper, methods	79.62	78.93	73.84

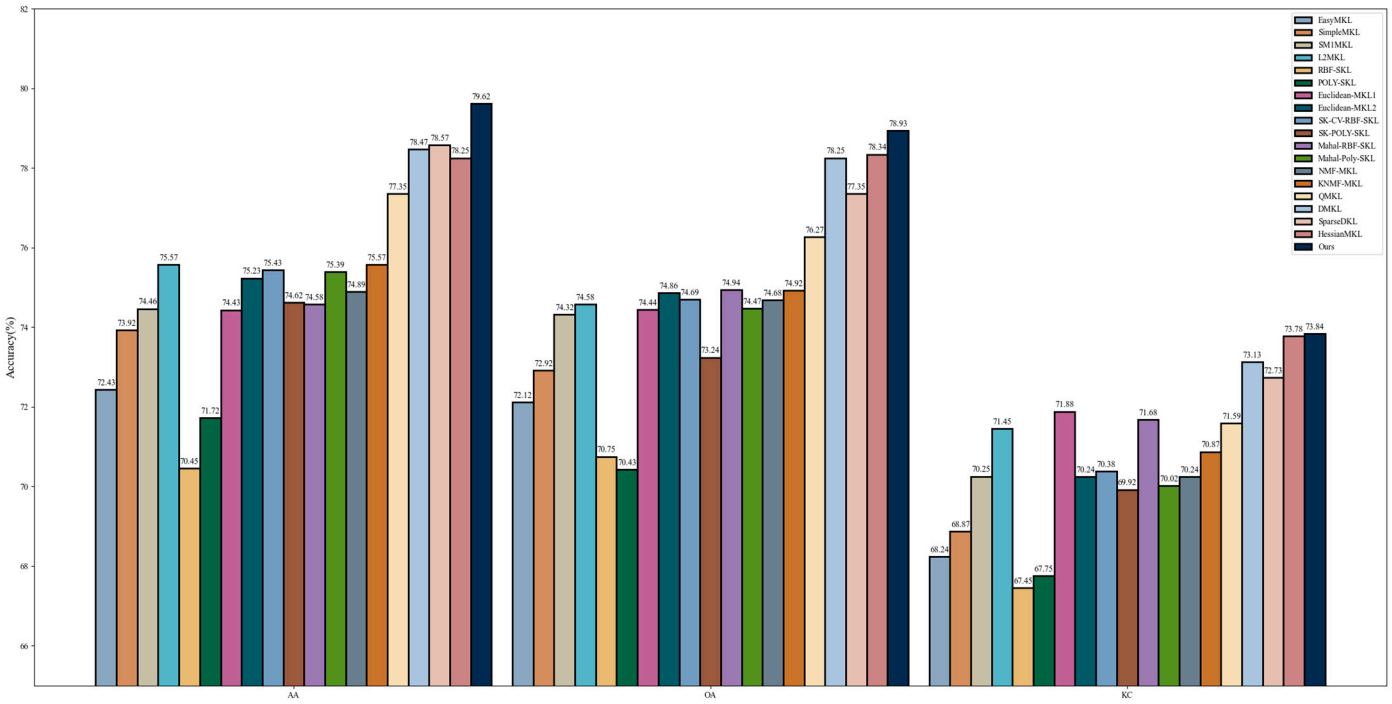


Figure A8. Performance comparison of the different algorithms on the Houston 2018 dataset.

References

1. Yu, K.; Xu, W.; Gong, Y. Deep Learning with Kernel Regularization for Visual Recognition. *Adv. Neural Inf. Process. Syst.* **2008**, *21*, 1889–1896.
2. Cho, Y.; Saul, L.K. Kernel Methods for Deep Learning. *Diss. Theses-Gradworks* **2012**, *28*, 342–350.
3. Brahma, P.P.; Wu, D.; She, Y. Why Deep Learning Works: A Manifold Disentanglement Perspective. *IEEE Trans. Neural Netw. Learn. Syst.* **2015**, *27*, 1997–2008. [CrossRef] [PubMed]
4. Wong, W.K.; Mingming, S. Deep learning regularized Fisher mappings. *IEEE Trans. Neural Netw.* **2011**, *22*, 1668–1675. [CrossRef]
5. Wang, S.; Jiang, Y.; Chung, F.L.; Qian, P. Feedforward kernel neural networks, generalized least learning machine, and its deep learning with application to image classification. *Appl. Soft Comput.* **2015**, *37*, 125–141. [CrossRef]
6. Ni, K.; Prenger, R. Learning features in deep architectures with unsupervised kernel k-means. In Proceedings of the IEEE Global Conference on Signal and Information Processing, Austin, TX, USA, 3–5 December 2013; pp. 981–984.
7. Harikrishnan, V.; Paulose, A. A Deep Learning Kernel K-Means Method for Change Detection in Satellite Image. *Int. J. Sci. Res.* **2014**, *3*, 1220–1226.
8. Varma, M. Local Deep Kernel Learning for Efficient Non-linear SVM Prediction. In Proceedings of the International Conference on Machine Learning, Atlanta, GA, USA, 16–21 June 2013; pp. 486–494.
9. Strobl, E.V.; Visweswaran, S. Deep Multiple Kernel Learning. In Proceedings of the International Conference on Machine Learning and Applications, Miami, FL, USA, 4–7 December 2013; pp. 414–417.
10. Bi, H.; Deng, J.; Yang, T.; Wang, J.; Wang, L. CNN-based target detection and classification when sparse SAR image dataset is available. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2021**, *14*, 6815–6826. [CrossRef]
11. Shao, Z.; Yin, Y.; Lyu, H.; Soares, C.G.; Cheng, T.; Jing, Q.; Yang, Z. An efficient model for small object detection in the maritime environment. *Appl. Ocean. Res.* **2024**, *152*, 104194. [CrossRef]
12. Bergman, L.; Hoshen, Y. Classification-Based Anomaly Detection for General Data. *arXiv* **2020**, arXiv:2005.02359.
13. Ye, H.; Li, W.; Lin, S.; Ge, Y.; Lv, Q. A framework for fault detection method selection of oceanographic multi-layer winch fibre rope arrangement. *Measurement* **2024**, *226*, 114168. [CrossRef]
14. Sellami, A.; Abbes, A.B.; Barra, V.; Farah, I.R. Fused 3-d spectral-spatial deep neural networks and spectral clustering for hyperspectral image classification. *Pattern Recognit. Lett.* **2020**, *138*, 594–600. [CrossRef]
15. Shi, C.; Pun, C.-M. Superpixel-based 3d deep neural networks for hyperspectral image classification. *Pattern Recognit.* **2018**, *74*, 600–616. [CrossRef]
16. Li, Y.; Xie, W.; Li, H. Hyperspectral image reconstruction by deep convolutional neural network for classification. *Pattern Recognit.* **2017**, *63*, 371–383. [CrossRef]
17. Zhao, W.; Du, S. Spectral Image classification based on spatial Feature Extraction: A Dimension Reduction approach for Deep learning. *IEEE Trans. Geosci. Remote Sens.* **2016**, *54*, 4544–4554. [CrossRef]
18. Liang, M.; Jiao, L.; Yang, S.; Liu, F.; Hou, B.; Chen, H. Deep multiscale spectral-spatial feature fusion for hyperspectral images classification. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2018**, *11*, 2911–2924. [CrossRef]

19. Zhong, Z.; Li, J.; Luo, Z.; Chapman, M. Spatial residual network for hyperspectral image classification: A 3-D Deep Learning Framework. *IEEE Trans. Geosci. Remote Sens.* **2018**, *56*, 847–858. [CrossRef]
20. Chen, Y.; Zhao, X.; Jia, X. SPATIAL classification of hyperspectral data based on deep belief network. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2015**, *8*, 2381–2392. [CrossRef]
21. Li, J.; Zhao, X.; Li, Y.; Du, Q.; Xi, B.; Hu, J. Classification of hyperspectral imagery using a new fully convolutional neural network. *IEEE Geosci. Remote Sens. Lett.* **2018**, *15*, 292–296. [CrossRef]
22. Li, S.; Song, W.; Fang, L.; Chen, Y.; Ghamisi, P.; Benediktsson, J.A. Deep learning for hyperspectral image classification. *IEEE Trans. Geosci. Remote Sens.* **2019**, *57*, 6690–6709. [CrossRef]
23. Chen, Y.; Lin, Z.; Zhao, X.; Wang, G.; Gu, Y. Deep learning-based classification of hyperspectral data. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2014**, *7*, 2094–2107. [CrossRef]
24. Aiolli, F.; Donini, M. EasyMKL: A scalable multiple kernel learning algorithm. *Neurocomputing* **2015**, *169*, 215–224. [CrossRef]
25. Rakotomamonjy, A.; Bach, F.R.; Canu, S.; Grandvalet, Y. Simple MKL. *J. Mach. Learn. Res.* **2008**, *9*, 2491–2521.
26. Xu, X.; Tsang, I.W.; Xu, D. Soft margin multiple kernel learning. *IEEE Trans. Neural Netw. Learn. Syst.* **2013**, *24*, 749–761.
27. Cortes, C.; Mohri, M.; Rostamizadeh, A. L2 regularization for learning kernels. In Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, Montreal, QC, Canada, 18–21 June 2009; AUAI Press: Arlington, VA, USA, 2009; pp. 109–116.
28. Wang, D.; Yeung, D.S.; Tsang, E.C.C. Weighted Mahalanobis Distance Kernels for Support Vector Machines. *IEEE Trans. Neural Netw.* **2007**, *18*, 1453–1462. [CrossRef]
29. Gonen, M.; Alpaydin, E. Multiple kernel learning algorithms. *J. Mach. Learn. Res.* **2011**, *12*, 2211–2268.
30. Gu, Y.; Wang, C.; You, D. Representative Multiple Kernel Learning for Classification in Hyperspectral Imagery. *IEEE Trans. Geosci. Remote Sens.* **2012**, *50*, 2852–2865. [CrossRef]
31. Li, L.; Sun, C.; Lin, L.; Li, J.; Jiang, S. A dual-layer supervised Mahalanobis kernel for the classification of hyperspectral images. *Neurocomputing* **2016**, *214*, 430–444. [CrossRef]
32. Li, L.; Sun, C.; Lin, L.; Li, J.; Jiang, S. A Mahalanobis metric learning-based polynomial kernel for classification of hyperspectral images. *Neural Comput. Appl.* **2018**, *29*, 1103–1113. [CrossRef]
33. Gu, Y.; Wang, Q.; Wang, H.; You, D.; Zhang, Y. Multiple kernel learning via low-rank nonnegative matrix factorization for classification of hyperspectral imagery. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2014**, *8*, 2739–2751. [CrossRef]
34. Li, J.; Ma, L. Short-term wind power combined prediction based on EWT-SMMKL methods. *Arch. Electr. Eng.* **2021**, *70*, 801–817.
35. Yu, S.; Falck, T.; Daemen, A.; Tranchevent, L.C.; Suykens, J.A.; De Moor, B.; Moreau, Y. L 2-norm multiple kernel learning and its application to biomedical data fusion. *BMC Bioinform.* **2010**, *11*, 309. [CrossRef] [PubMed]
36. Liu, J.; Qiao, Y. Quasiconformal Mapping Kernel Machine Learning-Based Intelligent Hyperspectral Data Classification for Internet Information Retrieval. *Wirel. Commun. Mob. Comput.* **2020**, *2020*, 8873366. [CrossRef]
37. Wang, Q.; Gu, Y.; Tuia, D. Discriminative multiple kernel learning for hyperspectral image classification. *IEEE Trans. Geosci. Remote Sens.* **2016**, *54*, 3912–3927. [CrossRef]
38. Zhao, H.; Wu, J.; Li, Z.; Chen, W.; Zheng, Z. Double sparse deep reinforcement learning via multilayer sparse coding and nonconvex regularized pruning. *IEEE Trans. Cybern.* **2022**, *53*, 765–778. [CrossRef]
39. Chapelle, O.; Rakotomamonjy, A. Second order optimization of kernel parameters. In Proceedings of the NIPS Workshop on Kernel Learning: Automatic Selection of Optimal Kernels, 2008; Volume 19.