mathematics

Article

A New Bilinear Supervised Neighborhood Discrete Discriminant Hashing

Xueyu Chen [1], Minghua Wan [1,2,3,*], Hao Zheng [2], Chao Xu [1], Chengli Sun [4] and Zizhu Fan [5]

1 School of Information Engineering, Nanjing Audit University, Nanjing 211815, China; cxy23a@163.com (X.C.); xuchao@nau.edu.cn (C.X.)
2 Key Laboratory of Intelligent Information processing, Nanjing Xiaozhuang University, Nanjing 211171, China; zhenghao@njxzc.edu.cn
3 Jiangsu Key Laboratory of Image and Video Understanding for Social Safety, Nanjing University of Science and Technology, Nanjing 210014, China
4 School of Information Engineering, Nanchang Hangkong University, Nanchang 330063, China; sunchengli@nchu.edu.cn
5 School of Science, East China Jiaotong University, Nanchang 330013, China; zzfan@ecjtu.edu.cn
* Correspondence: wmh36@nau.edu.cn

Abstract: Feature extraction is an important part of perceptual hashing. How to compress the robust features of images into hash codes has become a hot research topic. Converting a two-dimensional image into a one-dimensional descriptor requires a higher computational cost and is not optimal. In order to maintain the internal feature structure of the original two-dimensional image, a new Bilinear Supervised Neighborhood Discrete Discriminant Hashing (BNDDH) algorithm is proposed in this paper. Firstly, the algorithm constructs two new neighborhood graphs to maintain the geometric relationship between samples and reduces the quantization loss by directly constraining the hash codes. Secondly, two small rotation matrices are used to realize the bilinear projection of the two-dimensional descriptor. Finally, the experiment verifies the performance of the BNDDH algorithm under different feature types, such as image original pixels and a Convolutional Neural Network (CNN)-based AlexConv5 feature. The experimental results and discussion clearly show that the proposed BNDDH algorithm is better than the existing traditional hashing algorithm and can represent the image more efficiently in this paper.

Keywords: feature extraction; perceptual hashing; discrete optimization; bilinear projection; neighborhood relationship

MSC: 68U10; 68U10

Citation: Chen, X.; Wan, M.; Zheng, H.; Xu, C.; Sun, C.; Fan, Z. A New Bilinear Supervised Neighborhood Discrete Discriminant Hashing. *Mathematics* **2022**, *10*, 2110. https://doi.org/10.3390/math10122110

Academic Editors: Xiaodong Yue, Shu Zhao and Jie Zhou

Received: 25 April 2022
Accepted: 15 June 2022
Published: 17 June 2022

1. Introduction

A series of problems that are caused by the "curse of dimensionality" make feature representation the core problem of the development of the visual field [1–3]. As a feature extraction algorithm with a low computational cost, the hashing algorithm has received more attention. The advantage of the hashing algorithm is that it uses binary hash code to represent each image, which is different from other subspace learning algorithms whose learned low-dimensional feature elements are floating point numbers. There is no doubt that the operation speed of binary is better than that of real numbers, and the storage efficiency of hash codes is higher.

Generally, hashing algorithms are divided into data-independent hashing and data-dependent hashing, according to whether training samples are considered or not. Data-dependent hashing has attracted much attention because it can obtain better hash codes more efficiently. In data-dependent hashing, many hashing algorithms obtain hash codes through linear projection, such as Spectral Hashing (SH) [4], Iterative Quantization (ITQ) [5] and Semi-supervised Hashing (SSH) [6]. In detail, the SH algorithm that is proposed by Weiss et al. [4] transforms the hash codes-solving problem into the feature-solving problem of the weighted Laplace–Beltrami operators in manifold learning by relaxing the constraints, and defines the concept of spectral relaxation. However, this algorithm is more suitable for the representation of dense matrices, and the data need to meet the uniform distribution, so its application range is very limited. The ITQ algorithm that is proposed by Gong et al. [5] is different from the Principal Component Analysis Hierarchical clustering method (PCAH) [7], which directly uses the Principal Component Analysis (PCA) [8] algorithm to reduce the dimension of the data, and then obtains the hash codes through symbol function quantization. The ITQ algorithm rotates the projected data set on the basis of PCA to reduce the loss of information in the quantization process. Therefore, the hash codes that are learned by the ITQ algorithm are more robust than the PCAH algorithm. In order to reduce the bad-code problem that is caused by the empirical error on the label set, the SSH [6] algorithm uses the distance in metric space or class label to obtain the sample relationship matrix, and constructs a regularizer using labeled and unlabeled data to reduce the over-fitting problem, and finally obtains the hash codes through simple linear mapping.

The hashing algorithms mentioned above usually transform discrete constraints into continuous problems without directly constraining the discrete variables. In order to further reduce the loss of data that is projected from the original space to the Hamming space, some more promising algorithms with performance-discrete constraints have been proposed, such as Discrete Supervised Hashing (SDH) [9], Supervised Discrete Hashing with Relaxation (SDHR) [10], and Supervised Discrete Discriminant Hashing (SDDH) [11]. The SDH algorithm that was proposed by Shen et al. [9] regresses the hash codes to the corresponding label through least square regression and uses the Discrete Cyclic Coordinate Descent (DCC) method to solve the hash codes. Obviously, the ordinary least-squares problem cannot achieve the optimal effect of classification. Therefore, in order to further improve the possibility of correct classification of each data point, Gui et al. [10] proposed SDHR, which learns the regression targets from the input data in a direct way and applies it to the learning of hash codes. Cui et al. [11] proposed the SDDH algorithm, which uses the given label information to construct a relationship matrix to directly learn a hashing function and effective hash codes at the same time.

One-dimensional feature extraction algorithms have apparent shortcomings because they need to manually convert two-dimensional images into one-dimensional vectors, resulting in the loss of image structure information. Yang et al. proposed the Two-Dimensional Principal Component Analysis (2DPCA) [12] on the basis of PCA, which makes it easier to accurately calculate the covariance matrix so as to obtain effective features more conveniently. Li et al. proposed the Two-Dimensional Linear Discriminant Analysis (2DLDA) [13] on the basis of the Linear Discriminant Analysis (LDA) [14], which not only reduced the calculation cost, but also improved the recognition accuracy. In manifold learning, similar ideas have attracted extensive attention, such as Two-Dimensional Local Preserving Projections (2DLPP) [15–17], which effectively improve the performance of feature extraction. These 2D feature-extraction algorithms make full use of the advantages of matrix features, so they show stronger robustness and save more space.

Inspired by the above Two-Dimensional methods, many hashing algorithms that are based on bilinear projection have been proposed. For example, Mao et al. proposed two-dimensional PCA hashing (2DPCAH) [18], which first uses the two-dimensional features of the image for principal component analysis, and then quantizes the reduced features to obtain the hash codes. Ding et al. proposed the 2D version of ITQ, Two-Dimensional Iterative Quantization (2DITQ) [19]. Based on the 2DLDA algorithm, they also proposed Linear Discrimination Discrete Hashing (LDDH) [19] to unify 2DLDA and SDH into one framework. 2DITQ alternately reduces the dimension of features from row direction and column direction by 2DPCA to project high-dimensional 2D image features into a low-dimensional matrix. Bilinear Random Projections for Locality-Sensitive Binary Code (BLSH-SIK) [20] proposed by Kim is a bilinear extension of LSH. Like LSH, BLSH-SIK can maintain the similarity between similar data, and the calculation time and memory consumption of binary code that are required by BLSH-SIK are significantly reduced compared with linear projection. Bilinear Projection-based Binary Codes (BPBC) [21] is a typical unsupervised bilinear projection-hashing algorithm. It transforms a large projection matrix into two small projection matrices to obtain better binary features with less calculating complexity. Bilinear Discriminant Analysis Hashing (BDAH) [22] uses the 2DLDA algorithm to reduce the dimension of features, and its performance is better than the unsupervised hashing algorithm. Another supervised bilinear projection hash is Bilinear Supervised Hashing (BSDH) [23], which no longer adopts the algorithm of continuous optimization quantization, but restricts discrete variables to obtain more competitive hash codes.

However, most of the existing 2D-hashing algorithms are divided into two steps. Firstly, the traditional 2D image-processing algorithm is used to reduce the dimension of the input image, and then the reduced features are quantified to obtain the hash codes. This shows that this algorithm has shortcomings, that is, directly binarizing the features after dimension reduction will lose a lot of useful information. Although many improved algorithms are proposed to solve this defect, such as 2DITQ, Global Similarity Preserving Hashing (GSPH) [24], and PCA-ITQ [25], the learning hash codes are still suboptimal. Therefore, based on the advantages of 2D image processing and the shortcomings of existing 2D-hashing algorithms, Bilinear Projection Supervised Neighborhood Discrete Discriminant Hashing (BNDDH) is proposed in this paper, in order to reduce the information loss in the process of projection into Hamming space, and to try to maintain the original geometric structure relationship between data, so as to obtain more efficient hash codes. Therefore, the algorithm that is proposed in this paper uses discrete constraints rather than continuous constraints to approximate, and in the projection process, the two-dimensional data are no longer transformed into one-dimensional vectors, but are directly projected by two orthogonal matrices.

Our main contributions are outlined as follows:

(1) We present a novel hashing algorithm called the BNDDH algorithm, which can perform perceptual hashing, an optimal Laplacian graph, and supervised discrete-discriminant preserving simultaneously in a unified strategy.
(2) We propose an optimized Laplace matrix construction strategy to maintain the neighborhood structure information of samples, so as to narrow the scope of similarity retrieval from the same object category to the same instance.
(3) We perform bilinear projection on two-dimensional images, which overcomes the disadvantage of large amounts of calculation of high-dimensional linear projection in the past. In addition, we use bilinear projection to better maintain the two-dimensional features of the image.

The rest of this paper is planned as follows: Section 2 mainly presents the algorithms that are related to this paper, including SDDH and BPBC. The objective function and optimization algorithm of BNDDH are given in detail in Section 3. The experimental results on the CIFAR-10, Yale-B, MINIST and AR databases show the effectiveness of BNDDH in Section 4. Finally, Section 5 summarizes the full text.

2. Related Algorithms

2.1. Supervised Discrete Discriminant Hashing (SDDH)

To facilitate reading, the main notations that are used in this article are shown in Table 1.

SDDH [11] is characterized by paying attention to the direct learning of hash codes. It is a supervised hashing algorithm with direct constraints on discrete variables. For the input sample $X \in R^{d\times n}$, the relationship matrix $S \in \{-1,1\}^{n\times n}$ of the samples is constructed according to the given label information. $S_{ij} = 1$ indicates that x_i and x_j are of the same class, and $S_{ij} = -1$ indicates different classes. The similarity measurement matrix

$W \in R^{b\times n}$ is defined to describe the relationship between the hash codes. The objective function is in Equation (1), and b $(b << d)$ is the length of the hash codes:

$$\begin{gathered}\max_{B,W,F} \operatorname{tr}(W^T BSB^T W) - v\|b_i - F(X)\|^2 \\ s.t. \;\; b_i \in \{-1,1\}^b\end{gathered} \tag{1}$$

Table 1. Explanation of main notations.

Notations	Size	Explanation
q	-	Total number of classes
$l_j\,(j = 1,2,\ldots,q)$	-	Label information
$k_j\,(j = 1,2,\ldots,q)$	-	Number of samples per class
n	-	Total number of samples
d_1, d_2	-	Size of feature matrix
d	-	The product of d_1 and d_2
c_1, c_2	-	Size of feature matrix after projection
c	-	The product of c_1 and c_2
b	-	Length of hash codes
o	-	Number of nearest neighbors
$X = [x_1, \cdots, x_n]$	$d_1 \times d_2 \times n$	Input data matrix
M	$d_1 \times d_2$	Average value of all sample
M_j	$d_1 \times d_2$	Average value of sample for j^{th} class
$B = [b_1, \cdots, b_n]$	$b \times n$	Binary hashing matrix
Z	$c \times n$	Bilinear projection feature matrix
A	$b \times b$	Similarity measure matrix
S_b	$n \times n$	Between-class relationship matrix
S_w	$n \times n$	Within-class relationship matrix
P	$b \times c$	Projection matrix
R_1, R_2	$d_1 \times c_1$ $d_2 \times c_2$	Bilinear rotation matrix
$H(X)$	-	Hash function

The second term of the formula defines $F(x) = G^T\Phi(x)$, $G \in R^{m\times b}$ as the projection matrix, which needs continuous iterative updating to obtain a better projection effect. ν is the penalty parameter. Like the idea of using anchors to reduce time complexity in Anchor Graph Hashing (AGH) [26], in SDDH $\Phi(X) = [\phi(x_1), \phi(x_2), \ldots, \phi(x_n)] \in R^{m\times n}$ $(m < n)$ is an m-dimensional column vector, which is achieved by using Radial Basis Function (RBF) kernel function, where $\phi(x_i) = \left[\exp\left(-\frac{\|(x_i,a_1)\|^2}{t}\right), \ldots, \left(-\frac{\|(x_i,a_m)\|^2}{t}\right)\right]^T$. $\{a_i\}_{i=1}^m$ are randomly selected anchors from the training databases X, and t is the kernel bandwidth. Experiments show that SDDH can effectively ensure the minimum feature loss when the samples are mapped to Hamming space.

2.2. Bilinear Projection-Based Binary Codes (BPBC)

BPBC [21] is a typical unsupervised bilinear-projection hashing algorithm. Given sample $X = \{x_1, x_2, \cdots, x_n\}(x_i \in R^{d_1\times d_2})$, these samples are projected into Hamming space by using random rotation matrix R $(R \in R^{d_1 d_2 \times d_1 d_2})$.

$$H(x) = \operatorname{sgn}(R^T x) \tag{2}$$

where $\operatorname{sgn}(\cdot)$ is binary processing to learn binary codes. For positive numbers, output $+1$, otherwise output -1. In order to reduce the loss from the projection process, the goal of the BPBC algorithm is to minimize the angle θ between $vec(\hat{R}^T X)$ and $vec(\operatorname{sgn}(\hat{R}^T X))$, where $vec(\cdot)$ denotes vectorization.

$$\max \sum_{i=1}^{n} \cos(\theta_i) = \sum_{i=1}^{n} \frac{\operatorname{sgn}(\hat{R}^T X_i)^T (\hat{R}^T X_i)}{\sqrt{d_1 \times d_2}} \tag{3}$$

Then, based on Kronecker product $\otimes$, the complete rotation is transformed into bilinear rotation $\hat{R} = R_2 \otimes R_1 (R_1 \in R^{d_1 \times d_1}, R_2 \in R^{d_2 \times d_2})$. Thus, the projection matrix is reduced to reduce the computational complexity and further accelerate the projection speed. The Equation (2) can be rewritten as:

$$H(x) = \operatorname{sgn}({R_1}^T x R_2) \tag{4}$$

To obtain the low dimensional c-bits $(c = c_1 \times c_2)$ hash code, let $R_1 \in R^{d_1 \times c_1}$ and $R_2 \in R^{d_2 \times c_2}$, the objective function can be transformed into:

$$\begin{aligned} \max_{B, R_1, R_2} \sum_{i=1}^{n} \cos(\theta_i) &= \sum_{i=1}^{n} \frac{vec(\operatorname{sgn}({R_1}^T X_i R_2))^T vec({R_1}^T X_i R_2)}{\sqrt{c}} \\ &= \frac{1}{\sqrt{c}} \sum_{i=1}^{n} tr(b_i {R_2}^T {X_i}^T R_1) \\ s.t.\ \ {R_1}^T R_1 &= I,\ {R_2}^T R_2 = I \end{aligned} \tag{5}$$

where $tr(\cdot)$ denotes the trace of matrix, I is the identity matrix. $b_i = \operatorname{sgn}({R_1}^T X_i R_2)$ and $B = [b_1, b_2, \cdots, b_n]$. In order to optimize the projection, the variables B, R_1 and R_2 need to be updated. SVD is used to obtain R_1 and R_2 for each iteration. Further details of optimization can be found in reference [21].

3. Proposed Methods

3.1. The Objective Function

Considering the labeled image dataset $X = [x_1, \cdots, x_n] \in R^{d_1 \times d_2 \times n}$, for each sample x_i, its corresponding label is l_j $(j = 1, 2, \ldots, q)$. Our goal is to learn the binary hash code b_i to represent x_i, and set $B = [b_1, \cdots, b_n]$.

In order to enhance the scalability of the algorithm, we obtain the binary hash codes of each input two-dimensional image through the hash function $H(X)$. The purpose of our learning is to maximize the similarity between classes and minimize the similarity within classes. In order to better measure the relationship between hash codes, we define the similarity measurement matrix A. We hope to learn the optimal hash function $H(X)$ by maximizing Equation (6); the specific definition of $H(X)$ can be found in Equation (20).

$$\max \sum_{(x_i, x_j) \notin S} H(X)^T M H(X) - \sum_{(x_i, x_j) \in S} H(X)^T M H(X) \tag{6}$$

We need to define a formulation to represent the relationship between the input data. Some hashing algorithms define the relationship matrix $S \in \{0, 1\}^{n \times n}$, where the value of the matrix is equal to 1, indicating that the two samples at the corresponding position belong to the same class, with 0 indicating different classes. Previous hashing algorithms only care about the label information of the sample when describing or representing the sample, so the scope of similarity retrieval is the same object label. However, in the real world, the given labels are not always accurate, and obtaining high-quality labels through Subject Matter Experts (SMEs) is often expensive and difficult [27,28], which affects the effectiveness of hash-learning to a certain extent. In order to avoid the loss that is caused by unreliable label information, we relax the relationship matrix S and define $S_b \in R^{n \times n}$ and $S_w \in R^{n \times n}$ to represent the relationship between classes and within classes, respectively. In this relationship matrix, we also notice the sample points with the same label but not belonging to the nearest-neighbor relationship, and accurately describe the geometric relationship between the samples by reducing their weight. Firstly, we define a between-class relationship matrix graph as:

$$S_b = \begin{cases} 1,\ x_i \in ONo^-(x_j)\ or\ x_j \in ONo^-(x_i) \\ \exp(-||x_i - x_j||/\varepsilon),\ x_i \in No^-(x_j)\ or\ x_j \in No^-(x_i) \\ 0,\ otherwise \end{cases} \tag{7}$$

where $ONo^{-}(x_i)$ represents a set, which belongs to different classes of x_i and does not belong to the o nearest neighbor of x_i; we give it the maximum weight. $No^{-}(x_i)$ represents the set of o nearest neighbors of x_i; we reduce the weight of elements in this set. The value of the weight is determined by calculating the direct Euclidean distance between x_i and x_j, which is recorded as $\|x_i - x_j\|$. $\exp(\cdot)$ is an exponential function and ε is the parameter.

Similarly, the within-class relationship matrix is defined as:

$$S_w = \begin{cases} 1,\ x_i \in No^{+}(x_j)\ \ or\ x_j \in No^{+}(x_i) \\ \exp(-||\mathrm{x}_i - \mathrm{x}_j|| / \varepsilon),\ \ otherwise \end{cases} \tag{8}$$

where $No^{+}(x_i)$ represents the set of o nearest neighbors belonging to the same classes of x_i.

Through the above definition, Equation (6) can be converted to the following form:

$$\begin{gathered} \max_{B,U} \operatorname{tr}(U^T B S_w B^T U\Big) - \operatorname{tr}(U^T B S_b B^T U) \\ s.t.\ \ B \in \{-1, 1\}^{b \times n} \end{gathered} \tag{9}$$

where U is obtained by decomposing the similarity matrix A, and $A = UU^T$.

Transform this problem into the solution of continuous variables, which brings steep losses. In this paper, we directly constrain binary codes to obtain better hash functions and hash codes. Our goal is to learn hash function $H(X)$ with a good scalability. Therefore, we need to add constraints to the above objective function and convert it into Equation (10)

$$\max_{B,U,H} \operatorname{tr}(U^T B S_w B^T U\Big) - \operatorname{tr}(U^T B S_b B^T U)\ - v\|B - H(X)\|^2 \tag{10}$$

where ν is the penalty parameter, and we hope that B and $H(X)$ are as similar as possible. However, in the real world, B slight differing from $H(X)$ is acceptable.

3.2. Bilinear Projection Learning

Firstly, we define bilinear rotation matrix $R_1 \in R^{d_1 \times c_1}$ and $R_2 \in R^{d_2 \times c_2}$ $(c_1 < d_1,\ c_2 < d_2)$; input data $X \in R^{d_1 \times\, d_2 \times\, n}$ will be converted to ${R_1}^T X R_2 \in R^{c \times n}$.

Referring to the Fisher Discriminant Analysis algorithm (FDA), we define the between-class scatter matrix as Equation (11), and the within-class scatter matrix as Equation (12):

$$\begin{aligned} D_b &= \sum_{j=1}^{q} k_j \|R_1^T (M_j - M) R_2\|^2 \\ &= tr(\sum_{j=1}^{q} k_j R_1^T (M_j - M) R_2 R_2^T (M_j - M)^T R_1) \\ &= tr(R_1^T (\sum_{j=1}^{q} k_j (M_j - M) R_2 R_2^T (M_j - M)^T) R_1) \end{aligned} \tag{11}$$

$$\begin{aligned} D_w &= \sum_{j=1}^{q} \sum_{i=1}^{k_j} \|R_1^T (X_i - M_j) R_2\|^2 \\ &= tr(\sum_{j=1}^{q} \sum_{i=1}^{k_j} R_1^T (X_i - M_j) R_2 R_2^T (X_i - M_j)^T R_1) \\ &= tr(R_1^T (\sum_{j=1}^{q} \sum_{i=1}^{k_j} (X_i - M_j) R_2 R_2^T (X_i - M_j)^T) R_1) \end{aligned} \tag{12}$$

In order to disperse the data between classes as much as possible, the data within classes should be similar as much as possible, which equals to maximizing D_b/D_w in Equation (13).

$$\begin{aligned} \max D_b/D_w = \ & tr(R_1^T(\sum_{j=1}^{q} k_j(M_j - M)R_2R_2^T(M_j - M)^T)R_1)/ \\ & tr(R_1^T(\sum_{j=1}^{q}\sum_{i=1}^{k_j}(X_i - M_j)R_2R_2^T(X_i - M_j)^T)R_1) \end{aligned} \quad (13)$$

To simplify the expression, we define $H_b^{Q_1}$, $H_w^{Q_1}$, $H_b^{Q_2}$ and $H_w^{Q_2}$ in Equation (14), Equation (15), Equation (16), and Equation (17), respectively.

$$H_b^{Q_1} = \sum_{j=1}^{q} k_j(M_j - M)^T R_1 R_1^T (M_j - M) \quad (14)$$

$$H_w^{Q_1} = \sum_{j=1}^{q}\sum_{i=1}^{k_j}(X_i - M_j)^T R_1 R_1^T (X_i - M_j) \quad (15)$$

$$H_b^{Q_2} = \sum_{j=1}^{q} k_j(M_j - M) R_2 R_2^T (M_j - M)^T \quad (16)$$

$$H_w^{Q_2} = \sum_{j=1}^{q}\sum_{i=1}^{k_j}(X_i - M_j) R_2 R_2^T (X_i - M_j)^T \quad (17)$$

Because the eigenvector with larger a eigenvalue carries more identification information, we calculate the eigenvalue of $\frac{H_b^{Q_2}}{H_w^{Q_2}}$ and use the eigenvector corresponding to the larger eigenvalue to form R_1, and by calculating the eigenvalue of $\frac{H_b^{Q_1}}{H_w^{Q_1}}$ to obtain R_2. After obtaining the bilinear projection matrices R_1 and R_2, we project the high-dimensional 2D image features to obtain a smaller two-dimensional feature matrix as shown in Equation (18). This operation can reduce the computing cost.

$$z_i = vec(R_1^T X_i R_2) \quad (18)$$

$$Z = [z_1, z_2, \cdots, z_n] \in R^{c \times n} \quad (19)$$

To ensure bilinear projection, we define the embedding function in Equation (20), where P is the projection matrix that projects Z into a lower dimensional space.

$$H(X) = PZ \quad (20)$$

Finally, the objective function is transformed into:

$$\max_{B,U,P} \mathrm{tr}(U^T B S_w B^T U) - \mathrm{tr}(U^T B S_b B^T U) \ - v\|B - PZ\|^2 \quad (21)$$

3.3. Optimization

In order to alternately optimize U, B and P to obtain better data representation, we need to fix two of them and update the remaining one. The optimization process is divided into the following steps:

1. Update U: U only depends on B. By keeping B fixed, we can achieve U by the optimization problem in Equation (22):

$$\max_{\mathrm{B,U}} \mathrm{tr}(U^T B S_w B^T U) - \mathrm{tr}(U^T B S_b B^T U) \quad (22)$$

U consists of the eigenvalue vector corresponding to the largest eigenvalues of $BS_wB^T - BS_bB^T$, which will be simplified into the eigenvalue–eigenvectors problem.

2. Update P: Through the objective function (21), we find that the value of P is only related to B, and can define the function $L(P)$ as shown in Equation (23). By fixing B, the projection matrix P can be easily calculated by regression:

$$L(P) = \|PZ - B\|^2 \tag{23}$$

Take the partial derivative of P and set it equal to 0, then we can achieve Equation (25):

$$\frac{L(P)}{\partial P} = 2(PZ - B)Z^T = 0 \tag{24}$$

$$PZZ^T = BZ^T \tag{25}$$

We can easily obtain:

$$P = BZ^T(ZZ^T)^{-1} \tag{26}$$

3. Update B: By keeping Z and U constant, we make the objective function transform into the following problem to optimize:

$$\begin{aligned} &tr(U^TBS_wB^TU) - tr(U^TBS_bB^TU) - \nu tr((B - PZ)^T(B - PZ)) \\ &= tr(B^T(UU^TB(S_w - S_b)) + \nu PZ) \end{aligned} \tag{27}$$

In the t-th iteration, we define the learned hash codes as B^t. Then, we can update B^{t+1} from Equation (28):

$$B^{t+1} = \text{sgn}((UU^TB^tS_w - UU^TBS_b) - \nu PZ) \tag{28}$$

Algorithm 1 gives the concrete steps of BNDDH:

Algorithm 1 BNDDH

```
Input: Training databases $X = \{x_i\}_{i=1}^n$, label information $l_i$, bilinear projection rotation size $c_1$
and $c_2$, number of iterations $t_1$ and $t_2$, hash bits $b$, parameter $\nu$.
1. Compute $M$ and $M_j$, respectively. Initialize diagonal matrix $R_1$.
2. Repeat the following steps $t_1$ times:
    Compute $H_b^{Q_1}$ by Equation (14);
    Compute $H_w^{Q_1}$ by Equation (15);
    Update $R_2$ by solving the eigenvalue–eigenvector problem of $\frac{H_b^{Q_1}}{H_w^{Q_1}}$;
    Compute $H_b^{Q_2}$ by Equation (16);
    Compute $H_w^{Q_2}$ by Equation (17);
    Update $R_1$ by solving the eigenvalue–eigenvector problem of $\frac{H_b^{Q_2}}{H_w^{Q_2}}$.
3. Initialize $B$ randomly. Compute $Z$ by Equation (18).
4. Repeat the following steps $t_2$ times:
    Update $U$ by Equation (22);
    Solve $P$ by Equation (26);
    Solve $B$ by Equation (28).
Output: Rotation matrix $R_1$ and $R_2$, the hash encoding matrix $B = \{b_i\}_{i=1}^n$.
```

The optimization process is mainly divided into two parts. First, update R_1 and R_2, with a computational complexity of $O({d_1}^3)$ and $O({d_2}^3)$. The second part can be divided into three sub processes, in which the time complexity of updating U is $O(bn^2)$, the time complexity of updating P is $O(c^2n)$, and the time complexity of updating B is $O(bn^2)$. Therefore, the total time complexity of the optimization process is $O(t_1({d_1}^3 + {d_2}^3) + t_2(2bn^2 + c^2n))$.

4. Experiment and Discussion

In order to verify the effectiveness of the BNDDH algorithm, we first introduce three experimental databases, including CIFAR-10 and Yale-B, MNIST and AR database. Then, we provide evaluation indexes to evaluate the effects of different algorithms. Finally, for each database, we give the experimental results and corresponding analysis.

4.1. Databases and Setting

As shown in Figure 1, the CIFAR-10 database involves 10 different classes, including birds, deer, cats, automobiles, airplanes, etc., all of which are 32 × 32 RGB, 60,000 pictures in total. The database is used to verify the ability of the BNDDH algorithm to classify the common coarse-grained label data in real life.

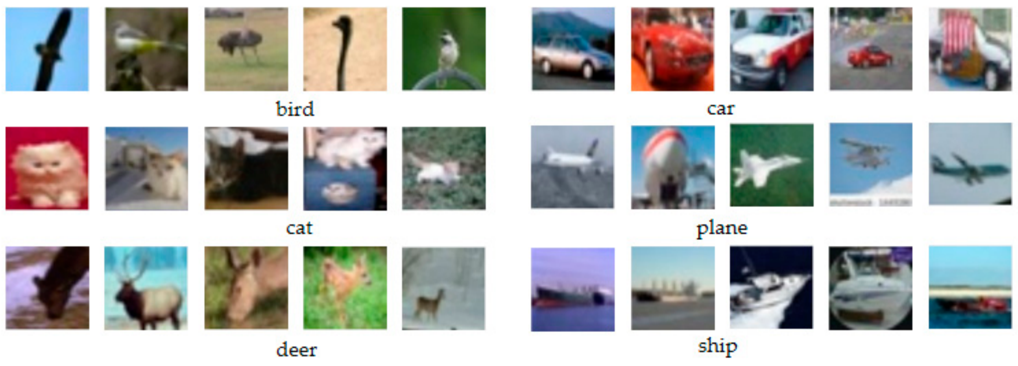


Figure 1. Some samples of the CIFAR-10 database.

At present, facial recognition is widely used, and it is very useful to process high-dimensional face data. Therefore, we choose the Yale-B face database to evaluate the performance of the proposed algorithm. As shown in Figure 2, this database includes 15 people, and each of them has 11 pictures, which is used to study facial expression changes. Each image is 50 × 50 pixels, and 165 pictures in total.

Figure 2. Some samples of the Yale-B database.

The other database we selected is the handwritten numeral set, as shown in Figure 3. The MNIST database is composed of 10 different categories of pictures, which are handwritten numerals 0 to 9, and the size of each picture is 28 × 28 pixels. If the handwritten data set can be recognized quickly and accurately by a computer, it will liberate the labor force of the need for manual input.

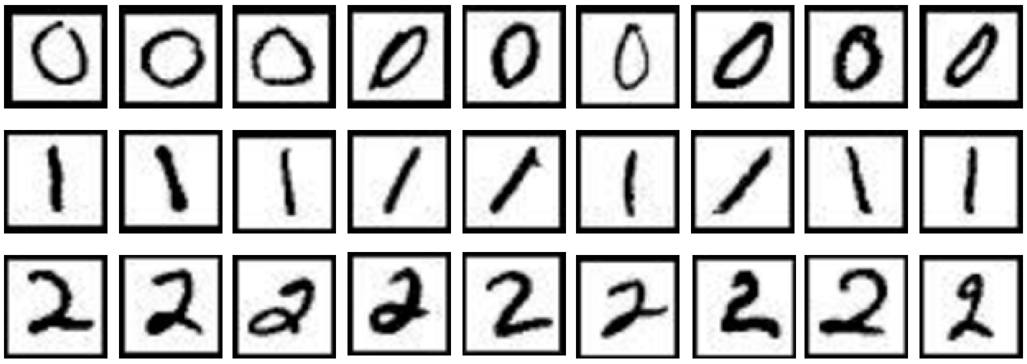


Figure 3. Some samples of the MNIST database.

The accuracy of facial recognition in the real world will be affected by expression, occlusion, and lighting conditions. The AR face database is used to verify the robustness of the proposed algorithm. As shown in Figure 4, the database contains 120 people, each with 26 pictures. The size of each picture is 50 × 40 pixels.

Figure 4. Some samples of the AR database.

For the sake of verifying the effectiveness of the BNDDH algorithm that is proposed in this paper, we select several classical hashing algorithms for performance comparison, including SH [4], PCAH [7], SDDH [11], 2DITQ [19], LDDH [19], BPBC [21], BSDH [23], LSH [29], SKLSH [30], SP [31], and DSH [32]. During the experiment, different hash code lengths are set and tested on five different hash code lengths: 8, 16, 32, 48 and 64. To ensure that the results are reliable in our experiment, we set $t_1 = 10$ and $t_2 = 5$. The iteration of other comparison algorithms adopts the convergence times that are set in the experiment of the original paper.

In the experiment, the Hamming radius is uniformly set to 2. That is, within the range of Hamming radius 2, judge whether the nearest data are retrieved correctly. We evaluate the difference between the classification of the algorithm and the ground truth. The precision rate, recall rate, precision–recall curves and mean average precision (MAP) are used to evaluate the performance of the algorithm.

4.2. Experimental Results

4.2.1. Experiment on the CIFAR-10 Database

In this part, we test the proposed algorithm on the CIFAR-10 database. Inspired by reference [8], we extract the depth two-dimensional features of the CIFAR-10 database and extract the image features by a Convolutional Neural Network (CNN) [33]. The two-dimensional AlexConv5 feature (256 × 36) is used to replace the original 32 × 32 RGB image in the experiment. This paper selects 10,000 data as the training set and 1000 data as the test sample. The precision–recall curve of the experiment is shown in Figure 5 below. The BNDDH algorithm shows good performance on different hash bits. Although the hash code is short, the BNDDH algorithm is still better than other comparative hashing algorithms.

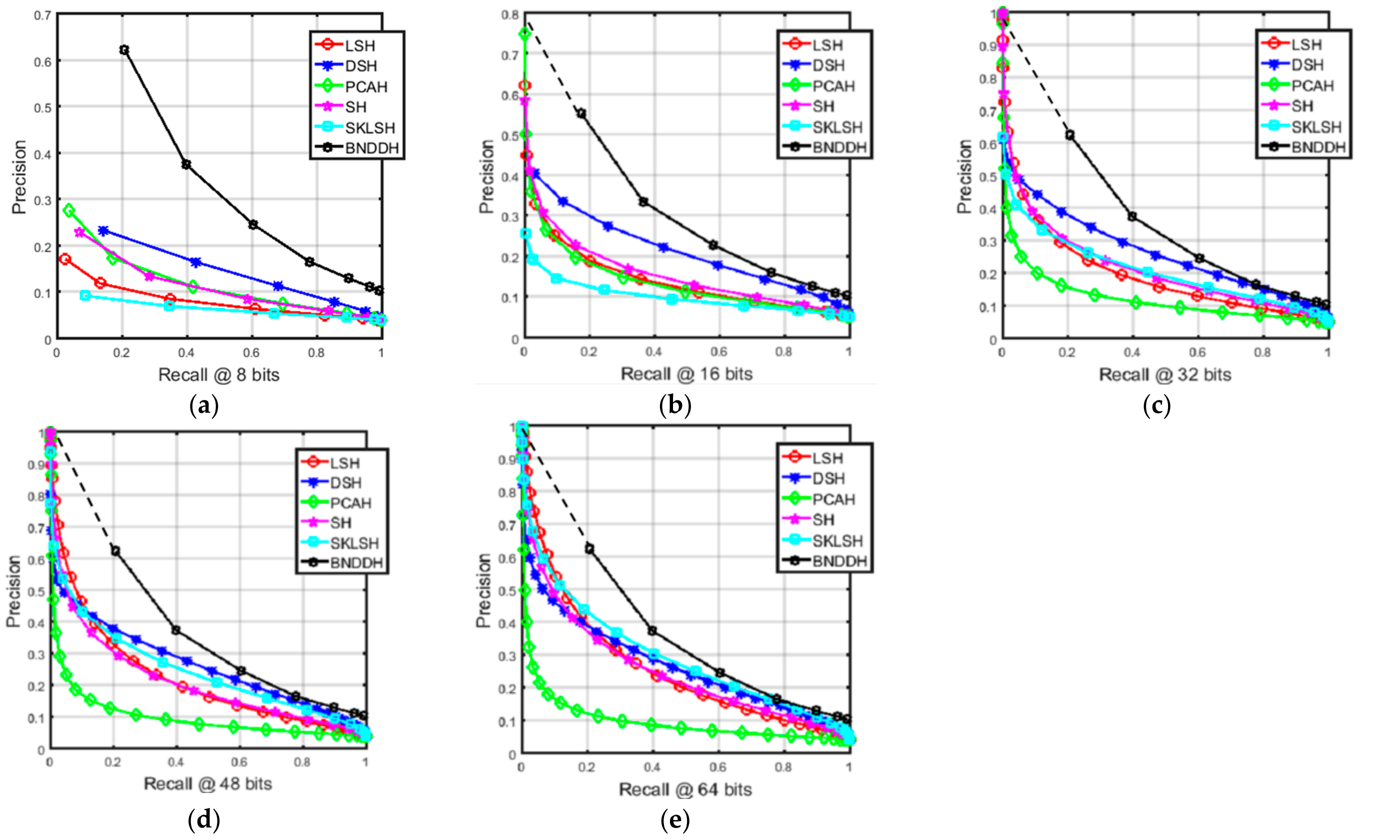


Figure 5. Precision–recall on the CIFAR-10 database (**a**) @8-bits (**b**) @16-bits(**c**) @32-bits (**d**) @48-bits (**e**) @64-bits.

Figure 6 shows the results of the precision and recall rate of the experiment on different hash bits from 8 to 64. When the hash bit is 32, the effect of BNDDH algorithm reaches its best; the precision rate is about 60%, and the recall rate is close to 60%. Among all the comparison algorithms, the hash codes that are learned by the PCAH algorithm are the worst, possibly because the PCAH algorithm relaxes the discrete constraint in the learning process, transforms the hash codes-solving problem into a continuous function-solving problem, and obtains the suboptimal hashing representation. Therefore, it is necessary to consider the direct constraint on discrete hash codes.

The MAP results are shown in Table 2. It can be found that the MAP of the BNDDH algorithm is optimal under different hash bits and is about 10% higher than that of SDDH and 2% higher than that of LDDH. The SDDH algorithm is a supervised hashing algorithm. It learns binary hash codes according to class label, which is better than the unsupervised hashing algorithm that was compared in the experiment. Because the same category label in the CIFAR-10 database is not the same object, its effect is inferior to the BNDDH algorithm. We believe that the retrieval of the BNDDH algorithm is more accurate. LDDH adopts the idea of 2DLDA to reduce the dimension of two-dimensional images and discretizes the hash codes with reference to SDH. From the experimental results, it can be seen that LDDH is better than other comparison algorithms; the results prove that it is effective to consider the two-dimensional features.

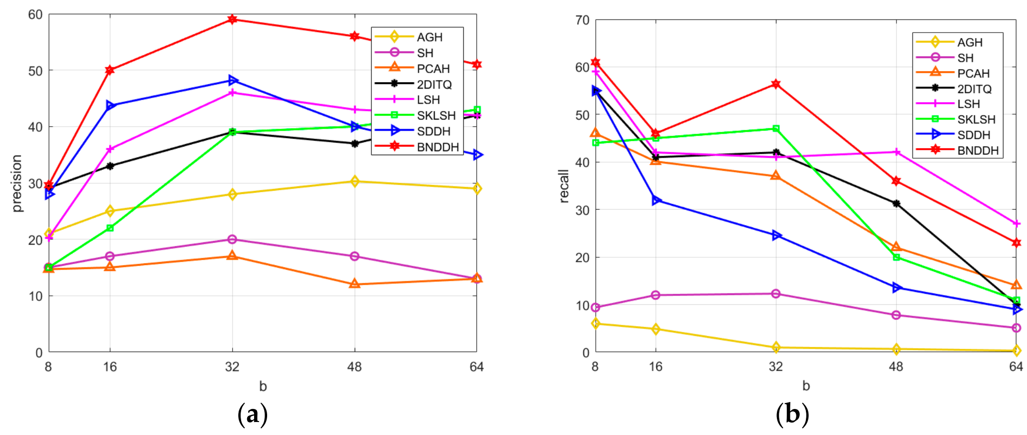


Figure 6. Precision and recall on the CIFAR-10 database (**a**) precision (**b**) recall.

Table 2. MAP of different hash code lengths on the CIFAR-10 database.

	Method	**8 bits**	**16 bits**	**32 bits**	**48 bits**	**64 bits**
MAP	**LSH**	0.0819	0.1240	0.1792	0.2232	0.2589
	AGH	0.0917	0.1421	0.1922	0.1544	0.1675
	DSH	0.1516	0.1959	0.3185	0.2971	0.3717
	PCAH	0.1202	0.1164	0.1057	0.1020	0.1117
	SH	0.1151	0.1306	0.1889	0.2137	0.2490
	SKLSH	0.0653	0.0900	0.1434	0.2711	0.2926
	SDDH	0.3270	0.4472	0.4228	0.4093	0.4801
	LDDH	0.4204	0.4842	0.5337	0.5459	0.5526
	BNDDH	**0.4375**	**0.5030**	**0.5519**	**0.5624**	**0.5644**

4.2.2. Experiment on the Yale-B Database

Different from CIFAR-10, the same object in the Yale-B database is marked with the same label. On this database, we analyze the performance of the BNDDH algorithm. The accuracy and recall under different hash bits are shown in Figure 7. It can be seen that when the hash code is equal to 8, the accuracy of the BNDDH algorithm is not as good as the SDDH algorithm and the 2DITQ algorithm. However, the BNDDH algorithm is better than other comparison algorithms in general.

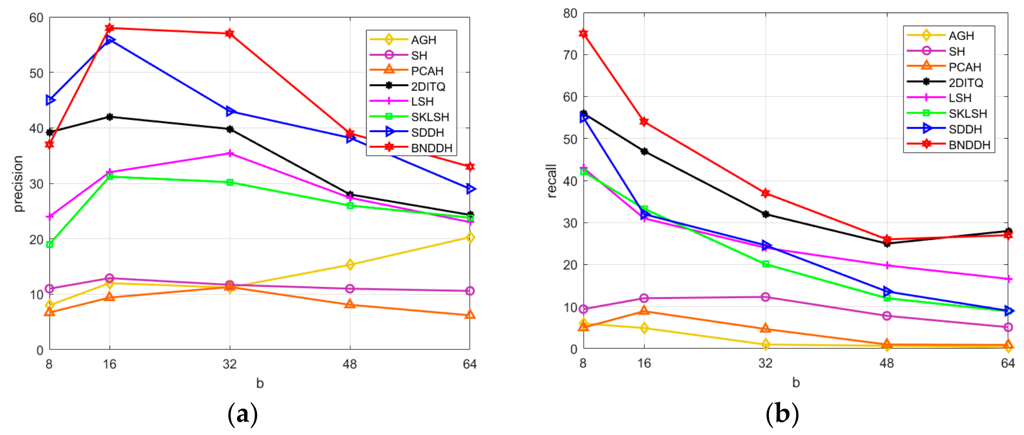


Figure 7. Precision and recall on the Yale-B database (**a**) precision (**b**) recall.

As shown in Table 3, when the hash codes are 8, 16, 32, 48 and 64, the MAP of BNDDH exceeds 0.65; when the hash code is 48, the MAP value is close to 0.8. The performance of the ITQ algorithm is generally consistent with that of the 2DITQ algorithm, and the MAP value of the 2DITQ algorithm is higher than that of the ITQ algorithm, except when the hash code is 48. We think it is useful to consider the two-dimensional characteristics of the image.

Table 3. MAP of different hash code lengths on the Yale-B database.

	Method	8 bits	16 bits	32 bits	48 bits	64 bits
MAP	LSH	0.3808	0.4698	0.5696	0.6811	0.6133
	DSH	0.4593	0.6309	0.6150	0.6222	0.6755
	PCAH	0.2850	0.2647	0.3239	0.3080	0.2959
	SH	0.2573	0.2637	0.2872	0.3444	0.3064
	SKLSH	0.2460	0.3854	0.5456	0.6031	0.7194
	SP	0.5285	0.6001	0.6436	0.6699	0.6507
	ITQ	0.5630	0.6480	0.6490	0.6916	0.7011
	2DITQ	0.5736	0.6493	0.6784	0.6632	0.7124
	SDDH	0.5891	0.6302	0.694	0.6786	0.6851
	BNDDH	**0.6595**	**0.6776**	**0.6934**	**0.7923**	**0.7687**

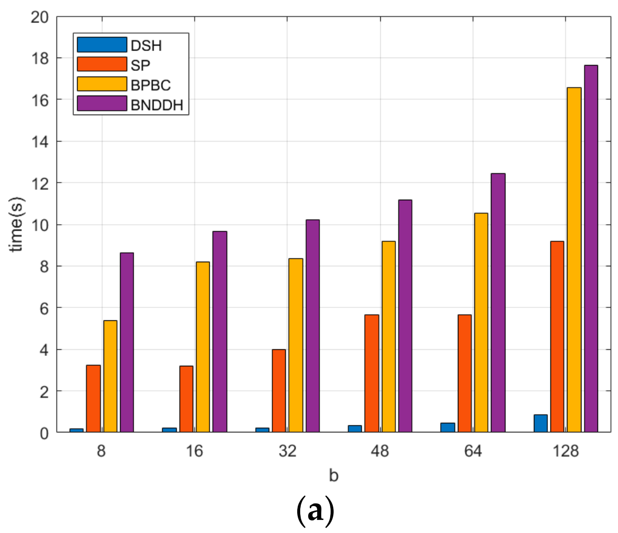
4.2.3. Experiment on the MNIST Database

The experimental results on the MINST database are shown in Table 4. It can be seen that when the length of the hash code is short, the algorithm advantage of the BNDDH algorithm is obvious. When the hash bit equals 8, the MAP of the comparison algorithm is lower than 0.3, while the MAP of the BNDDH that is proposed in this paper is close to 0.6. When the number of hash bits increases, the algorithm advantage of BNDDH is slight, which may be because the size of each picture in the MNIST database is 28 $\times$ 28, while the BNDDH algorithm is more suitable for a high-dimensional database. Among all the comparison algorithms, the best performance is from the BPBC algorithm, which is an unsupervised hashing algorithm based on bilinear projection. We believe that it is rational to integrate the idea of bilinear projection into a hashing algorithm, and the experimental results support this point of view.

Table 4. MAP of different hash code lengths on the MNIST database.

	Method	8 bits	16 bits	32 bits	48 bits	64 bits
MAP	LSH	0.0812	0.2103	0.3220	0.4586	0.5015
	DSH	0.0840	0.1968	0.3916	0.5540	0.6500
	PCAH	0.1987	0.3524	0.3833	0.3654	0.3427
	SH	0.1410	0.2971	0.4077	0.4573	0.4624
	SKLSH	0.0583	0.0557	0.2134	0.2265	0.3093
	SP	0.2055	0.4763	0.6382	0.6811	0.7356
	BPBC	0.2533	0.5158	0.6829	0.7365	**0.7783**
	2DITQ	0.2731	0.3653	0.3715	0.4598	0.4641
	BNDDH	**0.5859**	**0.6676**	**0.6934**	**0.7409**	0.7686

We choose an unsupervised algorithm with relatively good MAP and the BNDDH algorithm to analyze the running time. The training time and testing time are shown in Figure 8. It can be seen that the training time of all algorithms increases with the increase in hash code. The training time of the DSH algorithm is the shortest, probably because the method is relatively simple. The DSH algorithm uses K-means to partition data, and learns projection vectors through the geometric structure of the data to obtain hash codes. The training time of the BNDDH algorithm is similar to that of another unsupervised bilinear projection BPBC algorithm, but the testing time of BNDDH is relatively long.



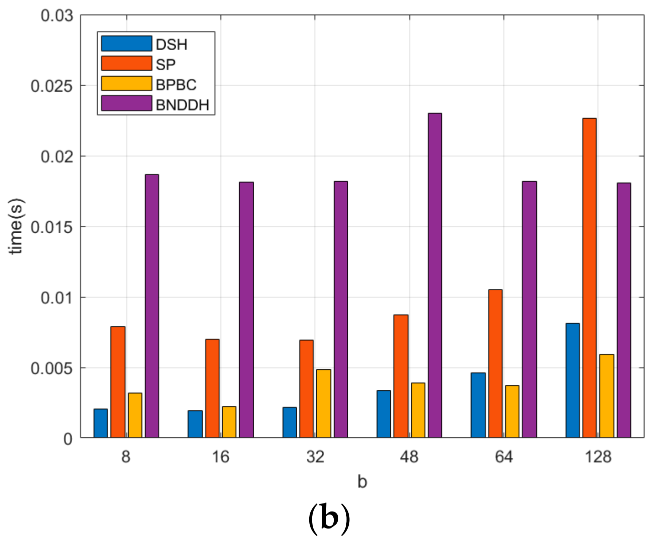
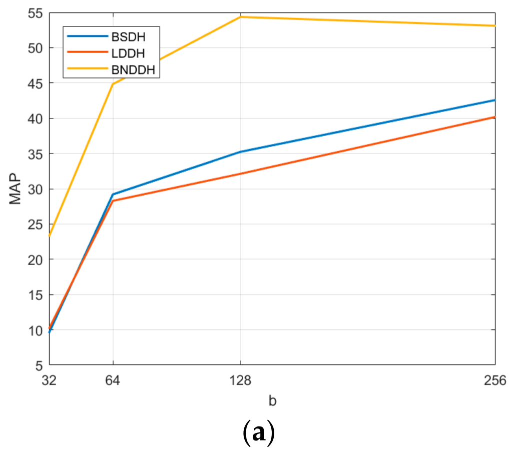


Figure 8. Training time on the MNIST database. (**a**) training time (**b**) testing time.



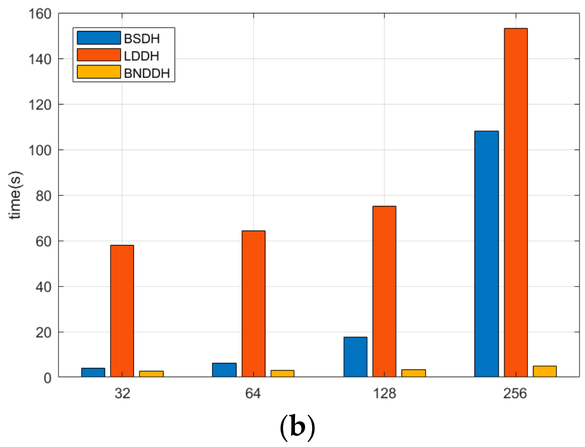


Figure 9. MAP and training time on the AR database (**a**) MAP (**b**) training time.

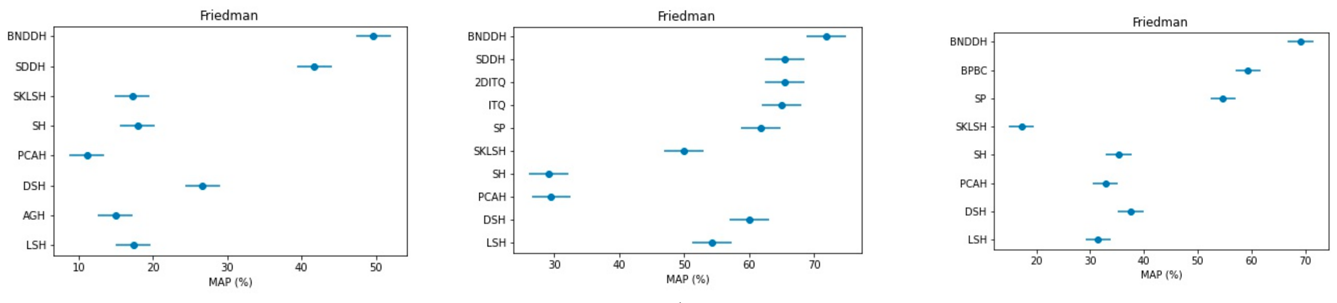
4.2.4. Experiment on the AR Database

On the AR face database, we test the performance of the proposed algorithm BNDDH. BSDH and LDDH are selected as the comparison algorithms. These two comparison algorithms are both supervised hashing algorithms based on two-dimensional image learning. The experimental results are shown in Figure 9. Under different hash bits, the MAP of BNDDH is higher than that of the two comparison algorithms. However, the MAP of the three algorithms is not good, which may be due to many interference factors in the AR database, such as facial expression, illumination, and facial occlusion. It is worth mentioning that the training time of BNDDH is significantly better than that of BSDH and LDDH. With the increase in hash-code length, the training time of BSDH and LDDH increases significantly, while the training time of BNDDH increases very little. This may be because both BSDH and LDDH use the Discrete Cyclic Coordinate Descent method when learning hash codes, which only updates one bit hash code at a time, thus, it takes a lot of time when the hash code is long. It can be said that the time advantage of BNDDH algorithm isobvious.

Image feature extraction is crucial in the fields of pattern recognition and machine learning. The above experimental results and analysis show that the BNDDH algorithm that is proposed in this paper performs well in feature extraction. Using compressed binary hash codes that are learned by the BNDDH algorithm to represent images is very useful in image storage and retrieval.

4.3. Friedman Test

In order to more intuitively represent the performance of each algorithm, we conducted a Friedman test and Nemenyi post-hoc test on three databases, respectively, and the results are shown in Figure 10. $\delta = q_\alpha \sqrt{k(k+1)/6N}$ is the critical threshold, where k represents the number of all hashing algorithms and $N = 5$ represents the number of different hash codes. The test level α of 0.05, so we can achieve $q_{\alpha_1} = 3.031$ $(k = 8)$ and $q_{\alpha_2} = 3.164$ $(k = 10)$. For each algorithm, dots represent the average value of the MAP in different hash codes, and the length of the horizontal line represents the value of critical threshold δ , so we achieve $\delta_1 = 4.696$ and $\delta_2 = 3.498$.



(**a**) (**b**) (**c**)

Figure 10. The Friedman test of the MAP (%) on the (**a**) CIFAR-10 database; (**b**) Yale-B database; (**c**) MNIST database.

4.4. Parameter Analysis

There are three important parameters: c_1, c_2 and ν in this paper. We choose to analyze and compare the influence of these parameters when the hash code length is 8, 16, 32, 48 and 64 on CIFAR-10.

In order to reconcile the computational efficiency and performance of the BNDDH algorithm, we compare the relationship between size c_1 and c_2 of the projected characteristic matrix and the value of MAP under different hash bits. Since the size of the AlexConv5 feature of CIFAR-10 is 36×256, the experimental setting c_1 is much smaller than c_2. Theoretically, we hope to learn the optimal hash code when c_1 and c_2 are as small as possible. The smaller c_1 and c_2 indicate the lower complexity of the algorithm, but the MAP value of the algorithm is often lower. As can be seen from Figure 11a,e, the MAP increases with the increase in c_1 and c_2, but when the value of c_1 is greater than 16 and the value of c_2 is greater than 48, the MAP is basically stable. In practical application, it is quite straightforward to find such critical values to ensure that learning is better.

Figure 11f shows the change of MAP of the BNDDH algorithm on the CIFAR-10 database when ν in $[10^{-8}\ 10^{-6}\ 10^{-4}\ 10^{-2}\ 10^{-1}\ 10^{1}\ 10^{2}\ 10^{3}\ 10^{4}]$. We can observe that when ν is equal to 10^3 and 10^4, the hash-learning ability decreases significantly. When α is 10^2 to 10^{-8} the change of map value is not obvious. This finding shows that in practical application, ν can be controlled slightly without significant parameter tuning.

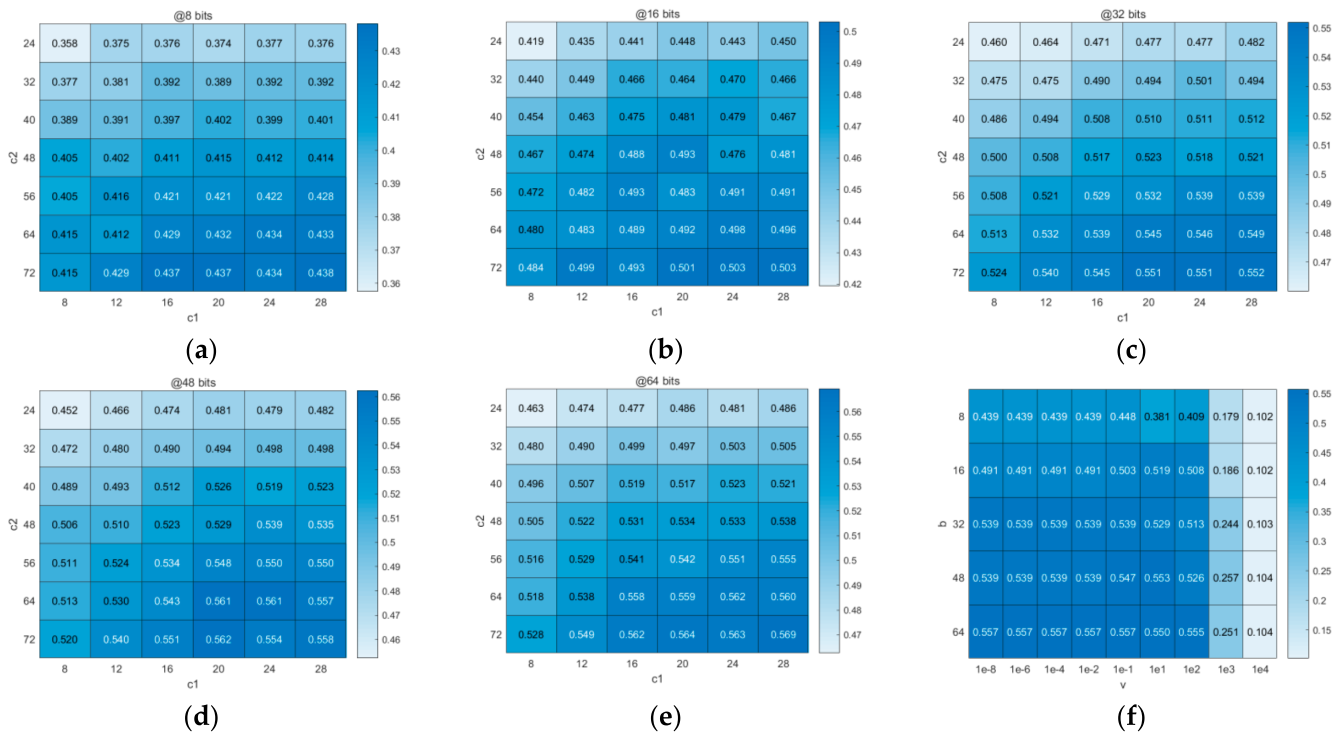


Figure 11. The heatmap of parameter selection experiments on the CIFAR-10 database. The MAP changes with c_1 and c_2 when: (**a**) hash bits equal 8; (**b**) hash bits equal 16; (**c**) hash bits equal 32; (**d**) hash bits equal 48; (**e**) hash bits equal 64. (**f**) The MAP changes with ν on different hash bits.

5. Conclusions

This paper proposed a Bilinear Supervised Neighborhood Discrete Discriminant Hashing algorithm, namely the BNDDH algorithm, which overcomes the shortcomings of the traditional manual transformation of two-dimensional pictures into one-dimensional vectors and applies the idea of bilinear projection to the supervised discrete discriminant hashing. Compared with one-dimensional projection, the projection matrix of bilinear projection is smaller, so the calculation memory is also smaller. Different from other supervised hashing, the proposed BNDDH algorithm maintains the relationship between the original data sample graphs by constructing the visual features of the data graph, and effectively solves the problem where different objects are marked as the same label. In order to reduce the projection loss from Euclidean space to Hamming space, the BNDDH algorithm directly restricts the binary hash code. Experimental results on four different databases show that compared with other hashing algorithms, the BNDDH algorithm performs better in similar image retrieval. However, the advantages of the BNDDH algorithm are not obvious in low dimensional data. In the future, we will explore the self-supervised learning of the BNDDH algorithm to further improve the application scope of the algorithm.

Author Contributions: Conceptualization, X.C. and M.W.; methodology, M.W. and H.Z.; software, X.C. and C.S.; validation, Z.F., M.W. and X.C.; formal analysis, C.S.; investigation, H.Z.; resources, C.X.; data curation, X.C.; writing—original draft preparation, X.C. and M.W.; writing—review and editing, X.C. and M.W.; visualization, X.C.; supervision, M.W.; project administration, C.S. and X.C.; funding acquisition, C.X. and M.W. All authors have read and agreed to the published version of the manuscript.

Funding: This work is partially supported by the Postgraduate Research &Practice Innovation Program of Jiangsu Province Nos. KYCX21_1945; the National Science Foundation of China under Grant Nos. 61876213, 61861033, 61991401, 62172229, 61976117, 71972102, 61976118; the Key R&D Program Science Foundation in Colleges and Universities of Jiangsu Province Grant Nos. 18KJA520005, 19KJA360001, 20KJA520002; the Natural Science Fund of Jiangsu Province under Grants Nos. BK20201397, BK20191409, BK20211295; and the Jiangsu Key Laboratory of Image and Video Understanding for Social Safety of Nanjing University of Science and Technology under Grants J2021-4. The Future Network Scientific Research Fund Project SRFP-2021-YB-25, and China's Jiangxi Province Natural Science Foundation (No. 20202ACBL202007). This work is funded in part by the "Qinglan Project" of Jiangsu Universities under Grant D202062032.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Publicly available datasets were analyzed in this study. This data can be found here: [https://www.cs.toronto.edu/~kriz/cifar.html; http://vision.ucsd.edu/~leekc/ExtYaleDatabase/ExtYaleB.html; http://yann.lecun.com/exdb/mnist/; http://www2.ece.ohio-state.edu/~aleix/ARdatabase.html] (accessed on 24 April 2022).

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses or interpretation of data; in the writing of the manuscript or in the decision to publish the results.

References

1. Abu Khurma, R.; Aljarah, I.; Sharieh, A.; Abd Elaziz, M.; Damaševičius, R.; Krilavičius, T.S. A Review of the Modification Strategies of the Nature Inspired Algorithms for Feature Selection Problem. *Mathematics* **2022**, *10*, 464. [CrossRef]
2. Wan, M.; Yao, Y.; Zhan, T.; Yang, G. Supervised Low-Rank Embedded Regression (SLRER) for Robust Subspace Learning. *IEEE Trans. Circuits Syst. Video Technol.* **2022**, *32*, 1917–1927. [CrossRef]
3. Berisha, V.; Krantsevich, C.; Hahn, P.R.; Hahn, S.; Dasarathy, G.; Turaga, P.; Liss, J. Digital medicine and the curse of dimensionality. *NPJ Digit. Med.* **2021**, *4*, 1–8. [CrossRef]
4. Weiss, Y.; Torralba, A.; Fergus, R. Spectral hashing. *Adv. Neural Inf. Processing Syst.* **2009**, *282*, 1753–1760.
5. Gong, Y.; Lazebnik, S.; Gordo, A.; Perronnin, F. Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval. *IEEE Trans. Pattern Anal. Mach. Intell.* **2013**, *35*, 2916–2929. [CrossRef]
6. Wang, J.; Kumar, S.; Chang, S.F. Semi-supervised hashing for large-scale search. *Pattern Anal. Mach. Intell IEEE Trans.* **2012**, *34*, 2393–2406. [CrossRef] [PubMed]
7. He, Y.; Wang, J.; Zhong, X.; Mei, L.; Wu, Z. PCAH: A PCA-based hierarchical clustering method for visual words construction. In Proceedings of the 15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, Shenzhen, China, 4–7 May 2015; pp. 1009–1018.
8. Turk, M.; Pentland, A. Eigenfaces for recognition. *Cognit. Neurosci.* **1991**, *3*, 71–86. [CrossRef]
9. Shen, F.; Shen, C.; Liu, W.; Shen, H.T. Supervised Discrete Hashing. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 37–45.
10. Jie, G.; Liu, T.; Sun, Z.; Tao, D.; Tan, T. Supervised discrete hashing with relaxation. *IEEE Trans. Neural Netw. Learn. Syst.* **2016**, *29*, 608–617.
11. Cui, Y.; Jiang, J.; Lai, Z.; Hu, Z.; Wong, W.K. Supervised discrete discriminant hashing for image retrieval. *Pattern Recognit.* **2018**, *78*, 79–90. [CrossRef]
12. Yang, J.; Zhang, D.; Frangi, A.F.; Yang, J.Y. Two-dimensional PCA: A new approach to appearance-based face representation and recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **2004**, *26*, 131–137. [CrossRef] [PubMed]
13. Li, M.; Yuan, B. 2D-LDA: A statistical linear discriminant analysis for image matrix. *Pattern Recogn. Lett.* **2005**, *26*, 527–532. [CrossRef]
14. Belhumeur, P.N.; Hespanha, J.P.; Kriengman, D.J. Eigenfaces vs. Fisherfaces: Recognition using class specific linear projection. *IEEE Trans. Pattern Anal. Mach. Intell.* **1997**, *19*, 711–720. [CrossRef]
15. Hu, D.; Feng, G.; Zhou, Z. Two-dimensional locality preserving projections (2DLPP) with its application to palmprint recognition. *Pattern Recognit.* **2007**, *40*, 339–342. [CrossRef]
16. Wan, M.; Chen, X.; Zhan, T.; Xu, C.; Yang, G.; Zhou, H. Sparse Fuzzy Two-Dimensional Discriminant Local Preserving Projection (SF2DDLPP) for Robust Image Feature Extraction. *Inf. Sci.* **2021**, *563*, 1–15. [CrossRef]
17. Lu, Y.; Yuan, C.; Lai, Z.; Li, X.; Wong, W.K.; Zhang, D. Nuclear Norm-Based 2DLPP for Image Classification. *IEEE Trans. Multimed.* **2017**, *19*, 2391–2403. [CrossRef]
18. Liang, X.; Tang, Z.; Xie, X.; Wu, J.; Zhang, X. Robust and fast image hashing with two-dimensional PCA. *Multimed. Syst.* **2021**, *27*, 389–401. [CrossRef]
19. Ding, Y.; Wong, W.K.; Lai, Z. Study on 2D Feature-Based Hash Learning. *IEEE Trans. Multimed.* **2020**, *22*, 1298–1309. [CrossRef]

20. Kim, S.; Choi, S. Bilinear random projections for locality-sensitive, binary codes. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1338–1346.
21. Gong, Y.; Kumar, S.; Rowley, H.A.; Lazebnik, S. Learning binary codes for high-dimensional data using bilinear projections. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Portland, OR, USA, 23–28 June 2013; pp. 484–491.
22. Liu, Y.; Bai, X.; Yan, C.; Zhou, J. Bilinear discriminant analysis hashing: A supervised hashing approach for high-dimensional data. In Proceedings of the Asian Conference of Computer Vision, Taipei, Taiwan, 21–23 November 2016; pp. 297–310.
23. Ding, Y.; Wong, W.K.; Lai, Z.; Zhang, Z. Bilinear Supervised Hashing Based on 2D Image Features. *IEEE Trans. Circuits Syst. Video Technol.* **2020**, *30*, 590–602. [CrossRef]
24. Liu, Y.; Feng, L.; Liu, S.; Sun, M. Global similarity preserving hashing. *Soft Comput.* **2018**, *22*, 2105–2120. [CrossRef]
25. Mao, M.; Zheng, Z.; Chen, Z.; Liu, H.; He, X.; Ye, R. Two-dimensional pca hashing and its extension. In Proceedings of the 2016 23rd International Conference on Pattern Recognition, Cancun, Mexico, 4–8 December 2016; pp. 1624–1629.
26. Liu, W.; Wang, J.; Kumar, S.; Chang, S.F. Hashing with Graphs. In Proceedings of the 28th International Conference on Machine Learning, Bellevue, WA, USA, 28 June–2 July 2011; pp. 1–8.
27. Zhang, L.; Du, G.; Liu, F.; Tu, H.; Shu, X. Global-Local Multiple Granularity Learning for Cross-Modality Visible-Infrared Person Reidentification. *IEEE Trans. Neur. Net. Lear.* **2021**, 1–11. [CrossRef]
28. Shen, F.; Zhou, X.; Yu, J.; Yang, Y.; Liu, L.; Shen, H.T. Scalable Zero-Shot Learning via Binary Visual-Semantic Embeddings. *IEEE Trans. Image Process.* **2019**, *28*, 3662–3674. [CrossRef] [PubMed]
29. Datar, M. Locality-sensitive hashing scheme based on p-stable distributions. In Proceedings of the Twentieth Annual Symposium on Computational Geometry, Brooklyn, NY, USA, 8–11 June 2004; pp. 253–262.
30. Raginsky, M.; Lazebnik, S. Locality-sensitive binary codes from shift-invariant kernels. *Adv. Neural Inf. Processing Syst.* **2009**, *22*, 1509–1517.
31. Yan, X.; He, K.; Kohli, P.; Sun, J. Sparse projections for high-dimensional binary codes. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 3332–3339.
32. Jin, Z.; Li, C.; Lin, Y.; Cai, D. Density Sensitive Hashing. *IEEE Trans. Cybern.* **2017**, *44*, 1362–1371. [CrossRef] [PubMed]
33. Albawi, S.; Mohammed, T.A.; Al-Zawi, S. Understanding of a convolutional neural network. In Proceedings of the 2017 International Conference on Engineering and Technology, Antalya, Turkey, 21–23 August 2017; pp. 1–6.