Ordinal Constraint Binary Coding for Approximate Nearest Neighbor Search

Hong Liu, Student Member, IEEE, Rongrong Ji, Senior Member, IEEE, Jingdong Wang, Senior Member, IEEE, and Chunhua Shen, Senior Member, IEEE

Abstract—Binary code learning, a.k.a. hashing, has been successfully applied to the approximate nearest neighbor search in large-scale image collections. The key challenge lies in reducing the quantization error from the original real-valued feature space to a discrete Hamming space. Recent advances in unsupervised hashing advocate the preservation of ranking information, which is achieved by constraining the binary code learning to be correlated with pairwise similarity. However, few unsupervised methods consider the preservation of ordinal relations in the learning process, which serves as a more basic cue to learn optimal binary codes. In this paper, we propose a novel hashing scheme, termed Ordinal Constraint Hashing (OCH), which embeds the ordinal relation among data points to preserve ranking into binary codes. The core idea is to construct an ordinal graph via tensor product, and then train the hash function over this graph to preserve the permutation relations among data points in the Hamming space. Subsequently, an in-depth acceleration scheme, termed Ordinal Constraint Projection (OCP), is introduced, which approximates the n-pair ordinal graph by L-pair anchor-based ordinal graph, and reduce the corresponding complexity from \(O(n^4)\) to \(O(L^3)\) \(L \ll n\). Finally, to make the optimization tractable, we further relax the discrete constrains and design a customized stochastic gradient descent algorithm on the Stiefel manifold. Experimental results on several large-scale benchmarks demonstrate that the proposed OCH method can achieve superior performance over the state-of-the-art approaches.

Index Terms—Binary Code Learning, Hashing, Image Retrieval, Ordinal Preserving, Tensor Graph, Discrete Optimization.

1 INTRODUCTION

COMING with the growth of dataset scales, approximate nearest neighbor (ANN) search has attracted extensive research attention in computer vision and machine learning [1], [2], [3]. Given a query point and a predefined distance metric, ANN aims at finding the most similar points approximately from the dataset. Comparing to traditional solutions like linear scanning, ANN merits in superior efficiency when facing massive data set in both time and memory, which however consumes a certain accuracy degeneration. Recently, binary code learning, a.k.a. hashing, has been popular for ANN search and has demonstrated promising performances. The basic idea is to learn a set of hash functions \(\{h_k: \mathbb{R}^d \rightarrow \{0, 1\}\}_{k=1}^r\), which encodes the original data from a \(d\)-dimensional real-valued feature space to an \(r\)-bit Hamming space, such that the nearest neighbors among the original data can be approximately and efficiently identified by using the Hamming distance.

Previous works in binary code learning mainly focus on finding random projections or permutations to produce the binary codes, e.g., Locality Sensitive Hashing (LSH) [4] and Min-wise Hashing (Min-Hash) [5]. LSH can accommodate a variety of distances or similarity metrics, e.g., \(l_p\) distance for \(p \in (0, 2]\), cosine similarity [6], and kernel similarity [7], [8]. Similarly, Min-Hash can approximate the Jaccard set similarity in the Hamming space. However, such hashing methods typically require long hash bits to achieve satisfactory results.

Apart from the above data-independent hashing, learning-based hashing has been well advocated, which can be subdivided into unsupervised and supervised methods. Unsupervised hashing, such as Spectral Hashing [9], Anchor Graph Hashing [10], Complementary hashing [11], Iterative Quantization [12], Order Preserving Hashing [13], Spherical Hashing [14], Discrete Graph Hashing [15], Scalable Graph Hashing [16], and Ordinal Embedding Hashing [17], model the data structure or distribution as constraints to learn short binary codes with high search accuracy. In contrast, supervised hashing, such as Binary Reconstruction Embedding [18], Minimal Loss Hashing [19], Kernel-based Supervised Hashing [20], FastHash [21] and Supervised Discrete Hashing [22], preserve the label relations in the produced Hamming space to learn binary codes.

Supervised hashing have shown superior performance over alternative schemes. However, the cost of supervised labels limits its extension and flexibility in many large-scale applications. In contrast, a more scalable solution is to replace the supervised labels with ranking information. In such a scenario, the triplet ranking relations are directly embedded into the learning process of hash functions [23], [24], [25]. Several ranking-preserved hashing algorithms have been proposed recently to learn more discriminative binary codes, e.g., Hamming distance metric learning (HDML) [26], Ranking-based Supervised Hashing (RSH) [23], Structure-based Hashing (StructHash) [24], and Top-Rank Supervised Binary Coding (Top-RSBC) [25]. However, most methods follow a triplet constraint setting with a stochastic gradient decent (SGD) optimization, which is less efficient and needs a large number of iterations. Comparing with ranking lists based constraint, we argue that a more practical and compact constraint is to use distance-
based ordinal information, which can recover the massive semantic ranking lists. An example of ordinal information is shown in the far left of Fig.1, in which the ordinal information among three samples $x_1$, $x_3$ and $x_4$ can be presented via comparing with $D(x_1, x_3)$ and $D(x_1, x_4)$, where $D(\cdot, \cdot)$ is the distance measure (e.g. Euclidean distance). However, few work consider the modeling of this ordinal cues in the problem of binary code learning. Concretely, firstly how to efficiently modeling the ordinal information is still left unexploited. Secondly, the complexity of the model is too high with reference to the existing works [13], [17]. Finally, the discrete constraint of the binary codes makes the model hard to optimize.

To address the challenges raised above, this paper present a novel unsupervised hashing method, termed Ordinal Constraint Hashing (OCH), which achieves an efficient and accurate ordinal-preserved hashing with three innovations detailed as below:

Firstly, OCH builds an ordinal set to capture the quartic ordinal relations for ranking information, rather than the pairwise or triplet relations used in the existing methods [10] [17]. To make use of the quartic relations, ranking lists for individual queries are converted to a quartic tuple that can be presented by using Eq.1 as below. Formally, let $x$ be the data point and $q$ be the query point, a quartic tuple is defined as:

$$
J = \{ (q; x_i, x_j, x_k) | D(q, x_i) < D(q, x_j) < D(q, x_k) \}. \tag{1}
$$

Building an ordinal graph by using such quartic tuples needs the comparison among all data in the training set, which is time-consuming. To tackle such issue, we further propose to construct an ordinal graph with tensor product, which captures the ordinal relation among samples in the original feature space. Such a construction avoids the time-consuming selection to model the ordinal tuples.

Secondly, to further handle the discrete constraint, we propose an optimization framework to solve the objective function with a novel updating rule. In the proposed optimization process, we relax the discrete coding function to a real-valued approximation. And, we have theoretically proven in Sec.3.6 that the orthogonal constraint is needed to hold during the model updating, which further makes the objective function hard to optimize. To handle this challenge, we propose a customized Stochastic Gradient Descent (SGD) algorithm, which formulates the corresponding problem as a combination of the traditional SGD and the Stiefel manifold optimization. The entire framework of the proposed OCH scheme is shown in Fig.1.

The proposed OCH method is compared against various state-of-the-art unsupervised hashing methods (including [4], [10], [12], [14], [16], [17], [27]) on several widely-used ANN search benchmarks, i.e., LabelMe, Tiny100K, VLAD500K, GIST1M, and NUS-WIDE. Quantitative results demonstrate that OCH outperforms the existing unsupervised hashing methods with a significant margin.

This work extends from our previous work in [28]. The new contributions include: (1) a new subsection on comparing the OCP to the standard PCA on anchors (i.e., Sec.3.2.3), (2) a new model on the parameter selection (i.e., Sec.3.4), (3) more details about the optimization scheme that revises the traditional SGD to be worked on Steifel Manifold (i.e., Sec.3.5), (4) a detailed theoretical analysis (i.e., Sec.3.6) on the ordinal preservation with the proposed OCH, (5) a group of new experiments to validate the robustness, efficiency and practicability of OCH (i.e., Sec.4), and (6) a group of comprehensive preliminaries illustrated in terms of both tensor product graph and ordinal embedding (i.e., Sec.2).

2 Preliminaries

2.1 Tensor Product Graph

Given an edge-weighted graph $G = (V, S_i)$, where $V = \{v_1, v_2, \ldots, v_n\}$ is a set of vertices representing the data points in the dataset, and $S_{ij}$ is the graph weight between vertices $v_i$ and $v_j$ measuring their adjacency under a certain similarity metric $1$.

Given the graph $G$, its affinity matrix can be reconstructed with the Markov chain based transition probability as:

$$
P_{ij} = S_{ij}/D_i, \tag{2}
$$

where $D_i = \sum_{j=1}^{n} (S_{ij})$ is the degree of each vertex.

To reduce the influence of noisy neighbors, we further build a sparse k-nearest neighbor graph instead of the using fully-connected graph as following:

$$
P_{ij} = \begin{cases} 
P_{ij} & j = kNN(i), \\
0 & \text{otherwise}. \end{cases} \tag{3}
$$

1. The metric can be defined by using Euclidean distance or Cosine similarity.
Given the affinity matrix $P$ in Eq.3, the construction of tensor product graph (TPG) is done by as below (a graphical illustration is shown in Fig.2):

$$T = P \otimes P.$$  \hfill (4)

### 2.2 Ordinal Embedding

We introduce the concept of ordinal embedding that serves as the preliminary of the proposed OCH method. Ordinal embedding, also known as ordinal scaling, non-metric multi-dimensional scaling, or isotonic embedding [29], targets at finding an embedding of items by comparing their pairwise distances. Given a set of $n$ data samples each with dimension $d_1$, let $\delta_{ij} \geq 0$ be the dissimilarity between the $i$-th and the $j$-th sample, and let $\delta_{ii} = 0$ and $\delta_{ij} = \delta_{ji}$, an ordinal relation set $C$ can be written as follows:

$$\delta_{ij} < \delta_{kl}, \forall (i, j, k, l) \in C.$$  \hfill (5)

Given a new dimension $d_2$ for the new space, the goal of ordinal embedding is to transfer samples in $C$ into a new feature representation $\{x_1, ..., x_n\} \in \mathbb{R}^{d_2}$, such that the ordinal constraints are preserved:

$$\delta_{ij} < \delta_{kl} \Rightarrow \|x_i - x_j\| \leq \|x_k - x_l\|, \forall (i, j, k, l) \in C, x_i \in \mathbb{R}^{d_2},$$  \hfill (6)

where $\|\cdot\|$ denotes the Euclidean norm. Eq.6 is referred to the Euclidean embedding$^2$. There are two common situations: quadruple comparisons that the number of $(i, j, k, l) \in C$ is $O(n)^4$, and triplet comparisons that the number of $(i, j, k) \in C$ is $O(n)^3$.

We further denote $\Omega \subseteq \mathbb{R}^{d_2}$ and a transformation function $f : \Omega \rightarrow \mathbb{R}^{d_2}$, where for all $\{x, y\} \in \Omega$ and some $\lambda > 0$, we have $\|f(x) - f(y)\| = \lambda \|x - y\|$. Here, $f$ is isotonic if for all $\{x, y, z, w\} \in \Omega$, the following constraints hold as:

$$\|x - y\| < \|z - w\| \Rightarrow \|f(x) - f(y)\| < \|f(z) - f(w)\|.$$  \hfill (7)

Equivalently, $f$ can be defined as a linear transformation, given by $f(x) = Wx + b$ with an orthogonal matrix $W$ and an offset $b \in \mathbb{R}^{d_2}$. $f$ is weakly isotonic if Eq.7 holds only for $\{x, y, z, w\} \in \Omega$ with $x = z$. $f$ is locally isotonic for each points $x \in \Omega$ with its neighborhood set.

### 2.3 Learning Binary Codes

Given a data point $x \in \mathbb{R}^{d_2}$, one can employ a set of hash functions $H = \{h_1, ..., h_r\}$ to produce an $r$-bit binary code $y = [y_1, y_2, ..., y_r]$ as:

$$y = [h_1(x), h_2(x), ..., h_r(x)],$$  \hfill (8)

where the $k$-th hash bit is calculated by $y_k = h_k(x)$. Such hash functions can be regarded as a mapping process in combination with quantization, as is widely used in many off-the-shelf hashing algorithm, e.g., LSH. This problem is mathematically formulated as follows:

$$h_k(x) = \text{sgn}(f_k(x)), \quad \text{sgn}(\cdot) \text{ is the sign function, which returns 1 if } f_k(x) > 0 \text{ and 0 otherwise.}$$  \hfill (9)

where $\text{sgn}(\cdot)$ is the sign function, which returns 1 if $f_k(x) > 0$ and 0 otherwise.

The proposed OCH aims to learn a set of hash functions to preserve such ordinal constraint in the produced Hamming space. To this end, a straightforward method is to maximize the loss function between the ordinal relation set $C$ and the corresponding relation in the Hamming space. In other words, the Hamming distance between $H(x_q)$ and $H(x_j)$ should be less than that between $H(x_q)$ and $H(x_j)$, while the Hamming distance between the $H(x_q)$ and $H(x_j)$ should be further less than that between $H(x_q)$ and $H(x_k)$. Therefore, we write our objective function as:

$$\text{max} \sum_{(q,i,j,k)\in C} I(D_H(b_q, b_i) \leq D_H(b_q, b_j) \leq D_H(b_q, b_k))$$  \hfill (10)

s.t. $b_i = \text{sgn}(W^T x_i)$, $W^TW = I$, $W \in \mathbb{R}^{d_2 \times r}$, \hfill (11)

where $D_H(b_q, b_i)$ is the Hamming distance between hash codes $b_q$ and $b_i$.

### 3.2 Ordinal Constraint Projection

#### 3.2.1 Anchor-based Model

The total number of the global ordinal relation is $O(n^4)$, which is too large to be used for training. To solve this problem, inspired by the recent work in [29], we relax the above global ordinal constraint to the local ordinal constraint, which significantly reduces the constraint number in the subset $C$. Concretely, we adopt an anchor-based ordinal embedding [30] that considers the comparison from any data points to the anchors, which:

3. Note that Hamming distance can be computed by bitwise XOR operation, which makes exhaustive search more faster in Hamming space than the original space.
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2018.2819978, IEEE Transactions on Pattern Analysis and Machine Intelligence.

reduces the number of ordinal constraints from $O(n^4)$ to $O(nL^3)$ ($L << n$ is the number of anchors). We first introduce anchors into Eq.13 to reduce the scale of ordinal constraints, which is done by performing K-means clustering to generate an anchor set $A = \{a_1, ..., a_L\} \in \mathbb{R}^{d \times L}$. Formally speaking, we rewrite the objective function in Eq.13 by the following:

$$
\max_{C(t)=1, t \in \mathcal{C}} \sum_{C(t)=1, t \in \mathcal{C}} \mathcal{I}(D_H(b_q, \hat{b}_i) \leq D_H(b_q, \hat{b}_j) \leq D_H(b_q, \hat{b}_k)),
$$

where $\hat{b}_i$ is the corresponding binary code of anchors $a_i \in \mathbb{R}^d$, and $\mathcal{C}$ is the subset of ordinal constraints.

3.2.2 Ordinal Constrained Projection on Anchors

Despite the usage of anchor-based ordinal constraints, the scale of subset $\mathcal{C}$ still makes the calculation to be consuming, which makes the Eq.14 hard to optimize. To this end, we propose a new projection method, termed Ordinal Constrained Projection (OCP), which uses an approximated scheme to further reduce the scale of ordinal tuples and make optimization easily. According to the relations in set $\mathcal{C}$, we approximate the constraints $\{\delta_{qi} < \delta_{qj} < \delta_{qk} | \forall(q, i, j, k) \in \mathcal{C}\}$ as follows:

$$
O = \sum_{\forall(q, i, j, k)} \mathcal{I}((\|x_q - a_i\|_2^2 - \|x_q - a_j\|_2^2)^2 - (\|x_q - a_i\|_2^2 - \|x_q - a_k\|_2^2)^2).
$$

Consequently, minimizing Eq.15 corresponds to approximating the ordinal constraint set $\mathcal{C}$ over all data points. Since data points in the dataset have been normalized, we rewrite Eq.15 as:

$$
O = \sum_{\forall(q, i, j, k)} \mathcal{I}((x_q^T a_i - x_q^T a_j)^2 - (x_q^T a_i - x_q^T a_k)^2)
$$

$$
= \sum_{\forall(q, i, j, k)} \mathcal{I}(a_i - a_j)^T M (a_i - a_j) - (a_i - a_k)^T M (a_i - a_k),
$$

where $M = \sum_q x_q^T x_q$ is a positive semi-definite symmetrical matrix, and $(i, j, k, q)$ presents the ordinal constraint among all the anchor points. According to the Euclidean triangle inequality, the triplet relation between original data $x_q$ and anchors $A$ can be rewriting as following:

$$
\|x_q - a_i\|_2^2 - \|x_q - a_j\|_2^2 \leq \|a_i - a_j\|_2^2.
$$

Therefore, the quartic ordinal relation can be approximated by the triplet ordinal relation among anchors, which are represented by the following:

$$
\|a_i - a_j\|_2^2 \leq \|a_i - a_k\|_2^2, \forall(i, j, i, k) \in \mathbb{T}, a_i \in \mathbb{R}^d,
$$

where $\mathbb{T}$ represents the ordinal relation set for anchors. As a result, the original ordinal constraints can be approximated by a much smaller set of ordinal constraints among anchors, which significantly reduces the scale of ordinal constraints from $[nL^3]$ to $[L^3]$.

4. Here $\delta_{qi}$ presents the dissimilar degree between $x_q$ and $a_i$. Without loss of generality, we assume that the data $x$ is normalized and mean-centered.

It is convenient to use SVD to decompose $M$ into $Z \in \mathbb{R}^{d \times d}$ such that $M = Z^T A Z$. Then Eq.16 is rewritten as:

$$
O = \sum_{\forall(i, j, i, k)} \mathcal{I}((a_i - a_j)^T Z^T A Z (a_i - a_j) - (a_i - a_k)^T Z^T A Z (a_i - a_k))
$$

$$
= \sum_{\forall(i, j, i, k)} \mathcal{I}((u_j - u_i)^T A (u_j - u_i) - (u_j - u_k)^T A (u_j - u_k))
$$

$$
\leq \sum_{\forall(i, j, i, k)} \mathcal{I}(\|A^2 (u_i - u_j)\|_2^2 - \|A^2 (u_i - u_k)\|_2^2)
$$

$$
\leq \sum_{\forall(i, j, i, k)} \mathcal{I}(\|u_i - u_j\|_2^2 - \|u_i - u_k\|_2^2)
$$

where $u_i = Z a_i \in \mathbb{R}^{d \times d}$.

And then, we embed the anchors set $A$ into the corresponding low-dimensional representation via projection $Z$ as:

$$
U = \{u_1, u_2, ..., u_L\} \in \mathbb{R}^{d \times d \times L}.
$$

It retains difficult to determine the optimal dimension $d_{svd}$, which is in fact a model selection problem. As discussed in [31], the dimension $d_{svd}$ of such subspace is a natural measure of the model complexity, while the least-squares error ($\|X - \hat{X}\|_F^2 = \sum_{i=d+1}^{D} \sigma_i^2$) can reflect the data fidelity. Following this criterion, we choose $d_{svd}$ as the smallest number such that the fidelity is less than a threshold $\tau > 0$:

$$
d_{svd} = \min_d \frac{\sum_{i=d+1}^{D} \sigma_i^2}{\sum_{i=1}^{D} \sigma_i^2} < \tau.
$$

Discussion: We give the corresponding analysis of the inequality in Eq.19 as below. We first define three matrices: $L = A^2$, $B = u_i - u_j$, and $C = u_i - u_k$. According to the third line in Eq.19, we count the right side where $\|LB\|_2 \geq \|LC\|_2 \geq 0$. Due to the constraint in Eq.18, we have another inequality that $\|B\|_2^2 \geq \|C\|_2^2$. Following the matrix inequality $|LB|_2 \leq |L|_2 |B|_2$ and $|LC|_2 \leq |L|_2 |C|_2$, we have the following equation:

$$
\|LB\|_2 \leq |L|_2 |B|_2 \leq \|LC\|_2 \leq |L|_2 |C|_2 \geq 0.
$$

Therefore, the fourth line of inequality in Eq.19 states the statistical upper bound of the ordinal relationship.

3.2.3 OCP vs. PCA on Anchors

The OCP is similar to the well-known Principle Component Analysis (PCA), both of which aim to find a subspace of anchors. The proposed OCP targets at using a series of approximated operation to find a subspace that has the ability to approximate the ordinal constraints between the original data and the anchors. Such a subspace can reduce the scale of the ordinal constraints. From Eq.19 to Eq.29, OCP performs the eigen-decomposition on the covariance matrix of $X$, and then transforms the anchors to the corresponding subspace. In contrast, PCA performs on the covariance matrix of $A$ rather than on the covariance matrix of $X$.

Interestingly, when we replace the query data $x_q$ with its corresponding anchor point $a_q$, Eq.15 can be rewritten as:

$$
\hat{O} = \sum_{\forall(q, i, j, k)} \mathcal{I}(\|a_q - a_i\|_2^2 - \|a_q - a_j\|_2^2)^2 - (\|a_q - a_i\|_2^2 - \|a_q - a_k\|_2^2)^2).
$$
That is, the global ordinal constraints among features $\mathbf{X}$ can be approximated by their corresponding anchors, and the scale of the ordinal relation can be reduced to $O[L^4]$. Following the similar pipeline from Eq.16 to Eq.21, the standard PCA can be conducted on the eigen-decomposition of the new matrix $\hat{M} = \sum L a_i^T a_i$. As a result, the final scale of the ordinal constraints is identical to that of OCP, which is also $O[L^4]$. However, such a replacement will result in a large loss of ordinal information, since the ordinal relation between different clusters is ignored.

Note that the anchor set $\mathbf{A}$ is obtained via K-means clustering of the dataset $\mathbf{X}$. Therefore, similar to [32], we first consider the case of $K = 2$ as an example, and calculate the sum of squared distances between two anchors $a_k$ and $a_l$:

$$d(a_k, a_l) := \frac{1}{n_k n_L} \sum_{i \in A_k} \sum_{j \in A_L} \|x_i - x_j\|^2 \leq 0,$$

$$\|a_K - a_L\|^2 = d(a_k, a_L) - d(a_k, a_K) - d(a_L, a_k),$$

where $n_k$ and $n_L$ are the number of the points in $a_k$ and $a_L$.

We then mainly consider the first items in Eq.15 and Eq.23 as example, which reflect the triplet ordinal relation between queries and anchors, and it can be easily extended to quartic ordinal relation. Obviously, taking Eq.25 into Eq.23, we can get the following equation for all anchors:

$$\sum_{k,i,j=1}^K \left[\|a_k - a_i\|^2 - \|a_k - a_j\|^2\right] = \sum_{k,i,j=1}^K \left[d(a_k, a_i) - d(a_k, a_j) - d(a_i, a_j) + d(a_j, a_i)\right].$$

Taking Eq.25 into Eq.15, the following equation can be obtained:

$$\sum_{q=1}^N \sum_{i,j=1}^K [\|x_q - a_i\|^2 - \|x_q - a_j\|^2],$$

$$\sum_{k,i,j=1}^K \sum_{q \in \mathbf{S}^L} [\|x_q - a_i\|^2 - \|x_q - a_j\|^2],$$

$$\sum_{k,i,j=1}^K \left[d(a_k, a_i) - d(a_k, a_j)\right].$$

Considering the above formulas Eq.26 to Eq.27, we can find that the $d(a_i, a_j) - d(b_i, a_j)$, which reflects the ordinal relation of features in the $i$-th and $j$-th anchors, will be ignored with standard PCA. In contrast, the proposed OCP can completely estimate the global ordinal relation. It therefore explains why OCP achieves better performance than PCA, which can better approximate the ordinal relationship. The experimental results will be further verified in Sec.4.6 below.

### 3.3 Reformulation

According to the definition of OCP, we reformulate our objective function of OCH in Eq.14 to approximate a continuous version that can be solved far more efficiently. After the usage of OCP, the global ordinal constraint can be approximated by the constraint among anchors. Let $\mathcal{T} = \{\forall (ij, ik) \mid (a_i, a_j, a_k)\}$ denotes the collection of quartet tuples, which are independently and uniformly sampled from the anchor set $\mathbf{A}$.

As the previous work in [17], this ordinal relation is represented by the triplet form $(a_i, a_j, a_k)$, where the pair $(a_i, a_j)$ is derived from the nearest neighbors using Euclidean distance, while $(a_i, a_k)$ is the dissimilar pair. Fig.3 (a) shows an example of such ordinal relation, where the red line represents the dissimilar pair and the blue line represents the similar pair. This triplet representation needs to randomly select the corresponding triplet tuples. And then all data points need to be compared side-by-side, which is both time and space consuming. The construction of quartic ordinal relation is even more complex than that of triplet ones. Moreover, it is hard to define similar/dissimilar pairs under the circumstance of unsupervised learning.

We address the above issue by proposing a novel ordinal graph with tensor product, dubbed Tensor Ordinal Graph (TOG), in which quartic ordinal relation can be efficiently represented. Given dataset $\mathbf{X}$ and the similarity measure (e.g., Euclidean distance), the affinity graph $\mathbf{S} \in \mathbb{R}^{n \times n}$ is constructed as follows:

$$\mathbf{S}(i, j) = \begin{cases} 0, & e^{-\|a_i - a_j\|^2/2\sigma^2}, \quad i = j, \\ \frac{1}{\mathbf{S}(i, j)}, & otherwise, \end{cases}$$

where $\mathbf{S}(i, j)$ represents the similarity between two data points. We further define a dissimilarity graph $\hat{\mathbf{S}} \in \mathbb{R}^{n \times n}$, with each entry as $\hat{\mathbf{S}}(i, j) = \frac{1}{\mathbf{S}(i, j)}$ and $\hat{\mathbf{S}}(i, i) = 0$.

Then, a tensor ordinal graph (TOG) $\mathbf{G}$ is defined as:

$$\mathbf{G} = \mathbf{S} \otimes \hat{\mathbf{S}},$$

where $\otimes$ is the Kronecker product of matrices defined as $\mathbf{G}(ij, kl) = \mathbf{S}(i, j) \cdot \hat{\mathbf{S}}(k, l)$. Each entry of $\mathbf{G}$ relates to four data points. Since $\mathbf{S}$ and $\hat{\mathbf{S}}$ are two $n \times n$ matrices, $\mathbf{G}$ is an $n^2 \times n^2$ matrix. Therefore, the ordinal relation between the quartic items in $(i, j, k, l) \in \mathcal{T}$ can be represented as:

$$\left\{ \begin{array}{ll} \delta_{ij} < \delta_{kl}, & \mathbf{G}(ij, kl) > 1, \\ \delta_{ij} > \delta_{kl}, & \mathbf{G}(ij, kl) < 1. \end{array} \right.$$ (30)

Fig.3 (b) shows a toy example, where the ordinal relation in the original feature space is $\delta_{ij} < \delta_{kl} < \delta_{ik}$. Such a relation can be simply calculated and compared with Euclidean distance. And for the quartic item $(i, k, i, j)$, the $(ik, ij)$-th entry is $\mathbf{G}(ik, ij) = \mathbf{S}(i, k) \cdot \hat{\mathbf{S}}(i, j) = \mathbf{S}(i, k)/\mathbf{S}(i, i)$. Due to the relation of $\mathbf{S}(i, k) < \mathbf{S}(i, j)$, we have $\mathbf{G}(ik, ij) < 1$, which reflects the truth ordinal relation $\delta_{ik} > \delta_{ij}$ in the original space.

Subsequently, for each $t = (ij, ik) \in \mathcal{T}$, an independent variable can be observed as:

$$C(t) = \left\{ \begin{array}{ll} 1, & \mathbf{G}(t) > 1 \\ -1, & \mathbf{G}(t) \leq 1. \end{array} \right.$$ (31)

Therefore, the overall objective function in Eq.14 for the proposed OCH approach is rewritten as follows:

$$\min_{\mathbf{t} \in \mathcal{T}} \max_{t} \left[0, C(t)(D_H(b_i, b_j) + \beta - D_H(b_i, b_k))\right]$$

$$s.t. \ b_i = \text{sgn}(\mathbf{V}^T \mathbf{u}_i), \mathbf{V}\mathbf{V}^T = I, \mathbf{V} \in \mathbb{R}^{d_{red} \times r},$$ (32)
Algorithm 1 Gradient Optimization over the Stiefel Manifold

1: Input the gradient $\nabla F$ of the objective function;
2: repeat
3: Project the gradient into the tangent space by Eq.37;
4: while $F'(\mathbf{R}(\eta\mathbf{P}(−\nabla F)))$ is not sufficiently smaller than $F'(\mathbf{V})$ do
5: Adjust the step size $\eta$;
6: end while
7: Update the orthogonal matrix $\mathbf{V}$ by Eq.39;
8: until Convergence.
9: Return the projection matrix $\mathbf{V}$.

where $\beta$ is a non-negative scaling parameter. Given $\mathbf{W}^T\mathbf{W} = \mathbf{I}$ with $\mathbf{W} = \mathbf{Z}^T\mathbf{V}$, we can easily get the new orthogonal constraint in Eq.32, which needs to be hold during optimization. Note that, the objective function in Eq.32 is discrete and hard to optimize. To solve this problem, we further relax the discrete constraints from the Hamming space to an approximated continuous space in the following subsection.

3.4 Optimization

Given the orthogonal constraints of the discrete variables and the projection matrix $\mathbf{V}$, the objective function is non-convex, which is hard to optimize. In this subsection, we introduce an alternative stochastic gradient descent algorithm on the Stiefel Manifold to solve the above problems efficiently.

3.4.1 Relaxation

We first relax the hashing function $H(\mathbf{u}_i) = \text{sgn}(\mathbf{V}^T\mathbf{u}_i)$ as:

$$H(\mathbf{u}_i) = \tanh(\mathbf{V}^T\mathbf{u}_i), \quad (33)$$

where $\tanh(\cdot)$ is the hyperbolic tangent function that is a good approximation for discrete $\text{sgn}(\cdot)$. It transforms the binary codes from $\{0, 1\}$ to $\{-1, 1\}$. According to the same definition in [20], the Hamming distance is approximated as:

$$D_H(\mathbf{b}_i, \mathbf{b}_j) \approx \frac{1}{2}(r - H^T(\mathbf{u}_i) \cdot H(\mathbf{u}_j)). \quad (34)$$

Finally, we use the Sigmoid function to replace the maximum function for efficient optimization. Based upon the above relaxations, the objective function in Eq.32 can be rewritten as:

$$F(\mathbf{V}) = \arg\min_{\mathbf{V}} \sum_{t \in \mathcal{T}} p(t), \quad \text{s.t.} \quad \mathbf{V}\mathbf{V}^T = \mathbf{I}, \quad (35)$$

where the $p(t)$ is the Sigmoid function defined as follows:

$$p(t) = \frac{1}{1 + \exp\left(C(t)(D_H(\mathbf{b}_i, \mathbf{b}_j) + \beta - D_H(\mathbf{b}_i, \mathbf{b}_j))\right)}. \quad (36)$$

3.4.2 Stochastic Gradient Descent on Stiefel Manifold

The optimization with respect to the orthogonal constraints has been recently studied in [33], [34]. In particular, to solve Eq.35, a straightforward method is to use gradient descent on the Stiefel manifold defined by $\mathcal{O} = \{\mathbf{V} \in \mathbb{R}^{d_{in} \times d_{out}} : \mathbf{V}\mathbf{V}^T = \mathbf{I}\}$ [35]. An off-the-shelf iterative solver has been developed in [33], which however is sensitive to the initialization and is difficult to be integrated into our optimization.

Recent advance in [36] aims to optimize the objective function in Eq.35 by performing a first-order, projected gradient optimization over the Stiefel manifold $\mathcal{O}$. Following [36], we define a tangent space to form the basis for the manifold projection based gradient methods: First, a curve on the manifold $\mathcal{O}$ is defined as a smooth map $\gamma(\cdot) : \mathbb{R} \rightarrow \mathcal{O}$. Second, the tangent space is formed via:

$$T_V \mathcal{O} = \{\dot{\gamma}(0) : \gamma(0) = \mathbf{V}\}, \quad (36)$$

where $\dot{\gamma}$ is the derivative of $\gamma(\cdot)$. The gradients $\nabla F$ of the objective function in Eq.35 are then calculated in the full space $\mathbb{R}^{d_{in} \times d_{out}}$. Such gradients are projected into the tangent space $T_V \mathcal{O}$ with a transformation $\mathbf{P} : \mathbb{R} \rightarrow T_V \mathcal{O}$ as:

$$\mathbf{P}(\nabla F) = \mathbf{V} \text{skew}(\mathbf{V}^T\nabla F) + (\mathbf{I} - \mathbf{V}\mathbf{V}^T)\nabla F, \quad (37)$$

where the function $\text{skew}((\mathbf{V}^T\nabla F)$ returns the skew-symmetric part of the matrix $\mathbf{V}^T\nabla F$. Any non-zero transformation $\mathbf{P}(\nabla F)$ in this tangent space will depart from the nonlinear constraint set. Therefore, a retraction is further needed to map this transformation onto the constraint set of Stiefel manifold constraint set, with a new transformation $\mathbf{R} : T_V \mathcal{O} \rightarrow \mathcal{O}$:

$$\mathbf{R}(\mathbf{M}) = \arg\min_{\mathbf{N} \in \mathcal{O}} \|\mathbf{N} - \mathbf{M}\|. \quad (38)$$

A simple solution is that $\mathbf{R}(\mathbf{M}) = \mathbf{OQ}^T$, where $\mathbf{M} = \mathbf{OSQ}^T$ is the singular value decomposition. This generic algorithm offers a global convergence proof by a linear time search [35]. Finally, the updating rule of the optimal projection matrix can be defined:

$$\mathbf{V} = \mathbf{R}(\eta\mathbf{P}(−\nabla F)), \quad (39)$$

where $\eta$ is the choice of the convergence parameter which is widely used in the first-order optimization. In this way, the multi-batch gradient of Eq.34 in the full space is given by:

$$\nabla F = \sum_{t \in \mathcal{T}} (p(t)(1 - p(t))) \left[\frac{\partial D_H(\mathbf{b}_i, \mathbf{b}_j)}{\partial \mathbf{V}} \cdot \frac{\partial D_H(\mathbf{b}_i, \mathbf{b}_j)}{\partial \mathbf{V}}\right], \quad (40)$$

and the gradient of Hamming distance is formulated as:

$$\frac{\partial D_H(\mathbf{b}_i, \mathbf{b}_j)}{\partial \mathbf{V}} = -\frac{1}{2} \{ \mathbf{a}_i \cdot [(1 - H^2(\mathbf{u}_i)) \otimes H(\mathbf{u}_i)]^T + \mathbf{a}_j \cdot [(1 - H^2(\mathbf{u}_j)) \otimes H(\mathbf{u}_j)]^T \}. \quad (41)$$

In Eq.41, $\otimes$ is the Hadamard product which represents the element-wise product.

The details of the projected gradient optimization and the proposed SGD over the Stiefel Manifold are shown in Alg.1, and Fig.4 respectively.
The hash functions \( h(x) \) can be approximated by a number of previous binary coding methods [20]. We consider such function \( h(x) \) as a discrete hash function, which has been used in semantic hashing. The experiments shown in Sec.4 prove that the proposed OCH has superior efficiency during the model training, dataset indexing, and online search. The details of the proposed OCH is shown in Alg.2.

**Algorithm 2 Ordinal Constraint Hashing (OCH)**

**Input:** Data set \( X = \{ x_1, x_2, \ldots, x_n \} \), parameters \( \gamma \) and \( \eta \).

**Output:** The hash functions \( h(x_i) = \text{sgn}(V^T Z x_i) \).

1. Generate centers \( A \) by K-means clustering algorithm;
2. Generate matrix \( Z \) by SVD decomposition and model selection in Eq.20;
3. For each \( a_i \) in \( A \), embed it to the subspace as \( u_i = Z a_i \);
4. Generate ordinal relations set \( T \) by TOG in Eq.29;
5. repeat
   6. Randomly select a multi-batch triplet from \( T \);
   7. Calculate the gradient according to Eq.40;
   8. Update \( V \) according to Alg.1;
9. until convergence or reaching the maximum iteration number.

3.5 Discussions

Unlike most existing unsupervised hashing algorithms, where the objective function is to preserve the pairwise relation in the Hamming space, our ranking-preserved hashing aims at preserving the distance ordinates in the Hamming space. Below we show that the final objective function in Eq.32 can preserve the ordinal relation as well.

Comparing Eq.32 to Eq.19, we approximate the Euclidean distance with Hamming distance, and then we use the hyperbolic tangent function to approximate the discrete sign(·) function. The key problem, however, is whether these two approximations can help to maintain the ordinal relations. According to the definition in Eq.7, the hash function should be weakly isotonic to maintain the ordinal constraints in the new distance measure.

Fortunately, we use the hyperbolic tangent function to approximate the discrete hash function, which has been used in previous binary coding method [20]. We consider such function \( h(x) = \tanh(x) \) with its first Maclaurin expansion as following:

\[
h(x) = h(0) + \frac{h'(0)}{1!} x + R_n(x) = x + R_n(x), \quad (42)
\]

where \( h' \) is the first derivative of function \( h \). By substituting the above first-order Maclaurin expansion into Eq.7, we obtain the following equation:

\[
\|h(x) - h(y)\| = \|x - y + R_n(x) - R_n(y)\| \approx \|x - y\|, \quad (43)
\]

where \( R_n(x) \approx 0 \) and \( R_n(y) \approx 0 \) as \( x, y \in [-1, 1]^{d \times r} \). Combining such constraint with the orthogonal constraint \( V^T V = I \), the similar constraints to Eq.7 can be generated:

\[
\|x - y\| < \|z - w\| \Rightarrow \|h(x) - h(y)\| < \|h(z) - h(w)\|. \quad (44)
\]

Thus, the hyperbolic tangent function is approximated to be weakly isotonic, which can maintain the ordinal constraints from the Euclidean distance to the Hamming distance. From another perspective, it explains why we need to guarantee the orthogonal constraint of projection matrix \( V \) in the final objective function.

At last, we discuss the time complexity of the proposed OCH algorithm. The overall training complexity of the proposed algorithm is \( O(t r L^2 d_{\text{wd}} + n L) \), where \( t \) is the number of iterations. It is linear to the scale of training set and is related to the complexity of the K-means step in the ranking-preserved hashing. The experiments shown in Sec.4 prove that the proposed OCH has superior efficiency during the model training, dataset indexing, and online search. The details of the proposed OCH is shown in Alg.2.

4 Experiments

In this section, we evaluate the proposed OCH hashing with comparisons to the state-of-the-art methods [4], [10], [12], [14], [27] and [16]. We also evaluate OCH in the task of semantic search, as well as the implementation of multiple hash tables.

4.1 Datasets

We perform extensive experiments on the following five benchmark datasets: LabelMe consist of 22,019 images, each of which is represented by a 512-dimensional GIST feature [37]. Tiny-100K-384D is a set of 1,00K 384-dimensional GIST descriptors, which is consisted of a subset of Tiny Images [38]. VLAD-500K-128D contains 500,000 VLAD descriptors [39] extracted from natural images [40]. The descriptors are PCA-compressed (with whitening) [41] to 128 dimensions. GIST-1M-960D [42] dataset is also widely used to evaluate the quality of nearest neighbor search, which consists of one million GIST descriptors. NUS-WIDE [43] contains 269,648 images, where each image is represented by a 128-dimensional VLAD feature with PCA-whitening [39], [41]. We select 181,365 labeled image-tag pairs from the whole dataset according to the top-21 largest concepts, as adopted in [10].

4.2 Evaluation Protocols and Baseline Methods

To evaluate the proposed hashing algorithm, we follow a series of widely-used evaluation protocols in recent papers [12], [14], [44].

The first one is to evaluate the nearest neighbor search by using Euclidean neighbors as ground-truth on the above four dataset, i.e., LabelMe, Tiny100K, VLAD500K, and GIST1M. For a given query, the top-K ranking items with Euclidean distance are defined as the similar data of the query. We set \( K \) to 1,000 as the default setting in most of our experiments. Then, based on the Euclidean ground-truth, we compute the recall curve, precision-recall curve, mAP score, and Precision@100. However, using smaller \( K \) is generally more challenging, which better characterizes the retrieval performance. To this end, we further set \( K \) as 1 and 10, and report the corresponding recall curves termed Recall-1 and Recall-10. The second one is to evaluate the performance of semantic retrieval, which are conducted on a large-scale label dataset, i.e., NUS-WIDE. For this task, we evaluate the precision curve and the precision-recall curve. The third one is to evaluate the performance with multiple hash tables. Actually, when aiming at a high recall, the locating time (the time spend on scanning the hash table) will increase with the increasing number of code length. Following the recent work in [45], one simple way to reduce the locating time is to use multiple hash tables, which is a better choice to cope with overwhelming long codes. In our settings, we use 16 tables with 16-bits and 20-bits in each hash tables, and then evaluate the precision of the 100-nearest neighbor, the dataset index time, and the average query time on both, i.e., VLAD500K and GIST1M datasets. For these three tasks, 1,000 data points are randomly selected as the test set, and the remaining are used as the training set. We further randomly select 10,000 points as the training set for all the algorithms, which has been used in [46]. The training data can not be gathered only one time to train the model. By such, it can more accurately reflect the model generalization by using the sampling data to train the model and then evaluate the performance. We repeat all the experiments 10 times and report the average performance over all runs.

We compared the proposed methods with the state-of-the-art hashing methods as follows: LSH (Local Sensitive Hashing) [4],
AGH (Anchor Graph Hashing) [10], IsoHash (Isotropic Hashing) [27], ITQ (Iterative Quantization) [12], SpH (Spherical Hashing) [14], SGH (Scalable Graph Hashing) [16], OEH (Ordinal Embedding Hashing) [17]. All the hashing methods above are unsupervised including the proposed one. In details, LSH is a random based hashing, IsoHash and ITQ are point-wise hashing, AGH and SGH are pairwise based hashing, and OEH is triplet based hashing. The source codes of all the compared methods are provided by the authors kindly. We implement our OCH hashing using MATLAB on a single PC with Duo Core I7-3421 and 64G memory, where the complete dataset can be stored.

According to the experimental analysis below, $V$ is randomly initialized with the Uniform distribution in a finite interval $[0, 1]$. For the constraints in Eq.32, the landmarks are consisted of 300 points obtained by K-means clustering. We give the analysis of this parameter in Sec.4.6. It is worth to note that landmarks generated by K-means reflect the distribution and structure of training data. Therefore, the ordinal information among landmarks is considered as vital as the sample to landmarks. We will give experimental analysis on whether such parameter is important. Finally, the parameter $\beta$ is set as 1 in all experiments to ensure better search accuracy.

### 4.3 Evaluation of Nearest Neighbor Search

As shown in Fig.5, Fig.6, Fig.7, Tab.1, and Tab.2 with code length varied from 32 to 128, the proposed OCH consistently achieves superior performance over all compared methods on the four datasets. In fact, the average mAP gain of OCH is more than 7% across the four datasets.

Furthermore, in the LabelMe benchmark, OCH beats the best baseline by up to 5.16% mAP score on 64 bit, while some of the baselines are only slightly competitive in this benchmark. For instance, ITQ [12] performs best on LabelMe when the hash bit is 32, but it is not competitive with the same hash bit on other benchmarks. However, the scale of LabelMe is small, which may make recent hashing algorithms easily overfit.

In addition to the mAP score, Tab.1 also shows our remarkable results on Precision@100, which achieve on average 0.66% improvement over the second best hashing algorithm, i.e., SGH [16]. Note that SGH also performs comparable over all datasets.
by using the kernel trick to capture the non-linear data structure. In comparison with SGH, OCP is a kind of linear embedding, which can only capture the linear structure in the dataset. To solve this problem, we evaluate our OCH with kernel embedding as similar to [12]. For a data point $x$, its approximated kernel feature can be presented by a random Fourier feature (RFF), which can be written as follows:

$$\Phi(x) = [\Phi_{w_1, b_1}(x), ..., \Phi_{w_d, b_d}(x)],$$ (45)

where $\Phi_{w_i, b_i}(x) = \sqrt{2} \cos(w x + b)$, $w$ is the random projection vector drawn from a normal distribution, and $b$ is the bias drawn from a uniform distribution. We first transform every feature to a kernel feature with RFF, with the radius of the Gaussian kernel set to be the average distance of the top 50 nearest neighbors. Fig.8 reports the results in terms of the nearest neighbor search on LabelMe dataset with the hash bit of 64. For this case, OCH-Kernel shows significant advantage over SGH [16] and KLSH [8].

For large-scale datasets, such as Tiny100K and VLAD500K, OCH can achieve competitive result consistently for image retrieval. The result of Pre@100 is shown in Tab.2, the top-100 items of the ranking list generated by OCH is more accurate than the others (up by 50%). Thus, most of the top items is the true nearest neighbor in the Euclidean space, which validates our argument that the ordinal relations, rather than the pairwise relations, should be preserved in the Hamming space. On the larger-scale GIST1M dataset, as shown in the second row of Fig.6 and Tab.2, OCH still achieves the highest search accuracy. With the increase of hash bits, the performance gain of OCH becomes more obvious.

Moreover, we report the results of Recall-1 and Recall-10 curves on four benchmarks in Fig.7. Analyzing among Fig.5, Fig.6 and Fig.7, we have found that OCH still achieves the best performance on four datasets. However, the performance gain gradually increases along with the number of $K$. The quartic tuples can reflect and preserve a wider range of ranking relationships in the Hamming space. Furthermore, comparing to another triplet-based hashing, e.g., OEH, we have found that the performance of OEH drop significantly when $K$ is set to a smaller number. Comparing OCH with OEH, a significant difference is in the orthogonally constraint of the projection matrix. According to the discussion in Sec.3.6, the orthogonal constraint for the projection

Fig. 6. ANN search of performance of different hashing methods on VLAD500K and GIST1M datasets. (Best view in color.)

<table>
<thead>
<tr>
<th>Methods</th>
<th>VLAD500K mAP</th>
<th>Pre@100</th>
<th>VLAD500K Recall @ 128 bits</th>
<th>GIST1M mAP</th>
<th>Pre@100</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>32</td>
<td>64</td>
<td>128</td>
<td>32</td>
<td>64</td>
</tr>
<tr>
<td>LSH</td>
<td>0.0335</td>
<td>0.0695</td>
<td>0.1213</td>
<td>0.0976</td>
<td>0.2058</td>
</tr>
<tr>
<td>AGH</td>
<td>0.0719</td>
<td>0.0929</td>
<td>0.0983</td>
<td>0.2016</td>
<td>0.2875</td>
</tr>
<tr>
<td>IsoHash</td>
<td>0.0811</td>
<td>0.1087</td>
<td>0.1607</td>
<td>0.2369</td>
<td>0.3065</td>
</tr>
<tr>
<td>SpH</td>
<td>0.0634</td>
<td>0.1113</td>
<td>0.1935</td>
<td>0.1805</td>
<td>0.3175</td>
</tr>
<tr>
<td>SGH</td>
<td>0.0835</td>
<td>0.1320</td>
<td>0.1847</td>
<td>0.2482</td>
<td>0.3600</td>
</tr>
<tr>
<td>ITQ</td>
<td>0.0883</td>
<td>0.1305</td>
<td>0.1691</td>
<td>0.2580</td>
<td>0.3591</td>
</tr>
<tr>
<td>OEH</td>
<td>0.0509</td>
<td>0.1100</td>
<td>0.2028</td>
<td>0.1581</td>
<td>0.3074</td>
</tr>
<tr>
<td>OCH</td>
<td>0.0883</td>
<td>0.1471</td>
<td>0.2110</td>
<td>0.2489</td>
<td>0.3790</td>
</tr>
</tbody>
</table>
Recall-1 curve on GIST1M.
Recall-10 curve on GIST1M.
Recall curve on VLAD500K.
Recall-10 curve on VLAD500K.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2018.2819978, IEEE Transactions on Pattern Analysis and Machine Intelligence

@64 gain over the second best baseline, achieves a best retrieval results comparing to baselines of [4], [10], [12], [217] the precision-

The results of the precision-recall curve and the precision-

In fact, such methods solve certain eigen-decomposition problems with orthogonality constraints to reduce the bit correlations. The proposed OCH uses the orthogonal constraints to maintain the ordinal relations in the Hamming space, which cannot reduce the performance with the increasing of hash bits. Furthermore, we have found that OCH has similar precision on 64 bit and 128 bit. It demonstrates that OCH can use more compact hash codes to get competitive performance of semantic retrieval.

4.5 Evaluation of Multiple Hash Tables

The above experiments focus on using single hash table in retrieval. However, in order to achieve a higher recall rate, most hashing schemes require longer hash codes, which results in lower search time. One way to reduce the search time (while keeping high recall) is to use multiple hash tables, each of which has more compact hash codes. Therefore, in this subsection we further evaluate the performance of integrating multiple low-bit hash tables. As mentioned in [45], AGH [10] and SGH [16] need to generate the kernel features, which is time consuming to encode the query data, so we neglect these two baselines here.

We evaluate the performance under two settings: 16 hash tables with 16 bit each table, and 16 hash tables with 20 bit each table. For a given query, we use the top-100 Euclidean nearest neighbors as the ground-truth, and locate 5,000 points in different Hamming distance (lower than 8) to return the candidate set, as similar to [45]. Then, we evaluate the top-10 precision scores of such candidate sets with the dataset indexing time (I-T) and the query time (Q-T) of per query5. It is worth to note that, Product Quantization (PQ) [42] is a well-known quantization method, which is also very efficient by exploiting lookup tables.

5. We use Matlab functions to learn the binary codes and C++ algorithm to search with the hashing index, since the Q-T and I-T cannot be combined together.

4.4 Evaluation of Semantic Search

We further evaluate the performance of OCH in the task of semantic search in NUS-WIDE, which has 10 different categories. The results of the precision-recall curve and the precision-# of retrieved samples curve are shown in Fig.9. OCH still obtains the best retrieval results comparing to baselines of [4], [10], [12], [14], [16], [17]. Based on the retrieved sample curves, OCH achieves a 45% accuracy in the top-2,000 ranking list, with a 10% gain over the second best baseline, i.e., ITQ [12] on 64 bit and OEH [17] on 128 bit. Comparing Fig.9 (c) with Fig.9 (d), the accuracy of the top-2,000 ranking list increases along with the increasing of the hash bits for most algorithms, except for AGH.

The above experiments focus on using single hash table in retrieval. However, in order to achieve a higher recall rate, most hashing schemes require longer hash codes, which results in lower search time. One way to reduce the search time (while keeping high recall) is to use multiple hash tables, each of which has more compact hash codes. Therefore, in this subsection we further evaluate the performance of integrating multiple low-bit hash tables. As mentioned in [45], AGH [10] and SGH [16] need to generate the kernel features, which is time consuming to encode the query data, so we neglect these two baselines here.

We evaluate the performance under two settings: 16 hash tables with 16 bit each table, and 16 hash tables with 20 bit each table. For a given query, we use the top-100 Euclidean nearest neighbors as the ground-truth, and locate 5,000 points in different Hamming distance (lower than 8) to return the candidate set, as similar to [45]. Then, we evaluate the top-10 precision scores of such candidate sets with the dataset indexing time (I-T) and the query time (Q-T) of per query5. It is worth to note that, Product Quantization (PQ) [42] is a well-known quantization method, which is also very efficient by exploiting lookup tables.

5. We use Matlab functions to learn the binary codes and C++ algorithm to search with the hashing index, since the Q-T and I-T cannot be combined together.

4.4 Evaluation of Semantic Search

We further evaluate the performance of OCH in the task of semantic search in NUS-WIDE, which has 10 different categories. The results of the precision-recall curve and the precision-# of retrieved samples curve are shown in Fig.9. OCH still obtains the best retrieval results comparing to baselines of [4], [10], [12], [14], [16], [17], [27]. Based on the retrieved sample curves, OCH achieves a 45% accuracy in the top-2,000 ranking list, with a 10% gain over the second best baseline, i.e., ITQ [12] on 64 bit and OEH [17] on 128 bit. Comparing Fig.9 (c) with Fig.9 (d), the accuracy of the top-2,000 ranking list increases along with the increasing of the hash bits for most algorithms, except for AGH.
We assign 8 bits to each subspace, resulting in 256 clusters in each subspace. For fair comparison, we set the number of subspace to be 32, which has the same code length to our first setting of 16bits-16tables. The distance computation of PQ can also be used by the lookup table for high-efficiency retrieval. Consequently, we use the recent fast search method for product-quantized codes, termed PQTable, which can achieve the same results as a linear PQ scan [47]. As a result, we use the original codes of both multi-index hashing methods, OCH-Kmeans and PQTable6 to evaluate above mentioned methods with precision, indexing time and query times.

The indexing and query time are shown in Tab.3. Note that the indexing stage is finished offline, which is not crucial as long as the indexing time is limited. Furthermore, both OCH and OEH need to use K-means clustering to generate the anchors as the first step. The time of K-means is 90.7s on VLAD500K dataset, and 860s on GIST1M dataset. PQ achieves the best precision than all competing methods. However, both the query and indexing time are both longer than those of other hashing methods. This is quite reasonable since the calculation, lookup and ranking are all based on the float vector, which consume more calculation and storage. Note that, the precision of the proposed OCH is higher than PQ with more quick query time, when using 20-bits per table. From another point of view, it also verifies the advantage of hashing with more quick query time, when using reasonable memory consumption.

It is worthy to note that, the model training and data indexing occupy a significant proportion of time cost for OCH, which however does not hold true for OEH. In fact, it is proved that the proposed OPC and TOG can significantly reduce the model training complexity, without decreasing any precision. Comparing to other methods, OCH achieves very excellent retrieval performance by constructing multiple hash tables8.

In Tab. 4, we replace the procedure to generate anchor points by using random sampling to accelerate the offline indexing. We have found that the precision will not decrease. To have an in-depth analysis, we use t-SNE [48] method to visualize the data distribution in a 2-dimension space. As shown in Fig. 10, these two datasets are densely distributed, and the distribution of random samples is similar to that of K-means centers. In fact, if the data distribution is dense, it is better to directly use random sampling to reduce the offline indexing time.

### 4.6 Analysis
Most previous hashing methods typically quantize the hash code by minimizing the loss between Euclidean distance and Hamming

---

7. https://github.com/matsui528/pqtable
8. All the algorithms have the similar query times, due to the similar hash index structure.
The proposed OCP is similar to the well-known PCA, but the two algorithms are not exactly the same as mentioned in Sec.3.2. The OCP is conducted on the original dataset $X$, while PCA is conducted on the anchor set $A$. Therefore, we mainly compare these two methods by keeping all other steps of the algorithm fixed, where the original OCH is termed $OCH-OCP$ and OCH with PCA is named as $OCH-PCA$. The evaluational results on two representative datasets (i.e., VLAD500K and GIST1M) are shown in Fig.12. From Fig.12, we can further clarify the theoretical benefits of OCP via experimental results.

We have also investigated the influence of anchors and the number of relations used in each iteration. Note that the anchors are generated by K-means clustering. In Fig.11 (a), we further show how $mAP$ varies with the increasing number of anchors from 100 to 1,000 for two datasets, i.e., GIST1M and VLAD500K. We have found that performances rarely change with the increasing number of centers. As a result, only a small number of anchors is used to approximate the overall ordinal relations, which is set to 100 for all five datasets. As in Fig.11 (b), we show the relation between $mAP$ and the number of ordinal relations, which are randomly selected in form of the batch set during each iteration. The result shows that $mAP$ does not change too much with the increasing number of batch set. Therefore, we can use a small batch set of ordinal relations in each iteration and maintain the overall search precision. Fig.11 (e) and Fig.11 (f) further report the $mAP$ results on GIST1M and VLAD500K by varying the size of training set. With the increasing number of training sets, all hash algorithms have similar performance increase/decrease trend, while in most cases the proposed OCH method still achieves the best accuracy. When we use the whole training data to train the hash function, all methods failed to achieve the best performance on both GIST1M and VLAD500K. Above quantitative results demonstrate that preserving such ordinal cues is a more fundamental goal for hashing. As for the comparison between our OCH and the state-of-the-art ranking preserve hashing OEH [17], both OEH and OCH adopt a two-step projection to find an optimal binary space. For OEH, PCA is used as the first step, which cannot reduce the scale of training space and still needs about $O(nL^2)$ triplet tuples to train the hash functions. On the contrary for the proposed OCH, we use the ordinal constraint projection in the first step, which not only reduces the original feature dimension, but also reduces the scale of training data. Moreover, OEH needs to construct the ordinal relations during each iteration, which is time consuming. In contrast, OCH constructs the ordinal relations by TOG only once before iteration, which is very convenient and efficient in iterative training. Tab.3 further shows the comparison of training plus indexing time between OEH and OCH methods, in which OCH approximately reduces one third training time of OEH but achieves better performance on VLAD500K and GIST1M.
reflect that the trained model has over-fitting to the training set, which is obvious on VLAD500K.

We further discuss the threshold parameter $\tau$ in Eq.21. The relation between $m$AP and the number of singular value is shown in Fig.11 (c), which demonstrates that the choice of such number is very important for retrieval performance, i.e., it cannot be too small or too large. According to Eq.21, we plot the curve of parameter $\tau$ vs. the number of singular values. Comparing Fig.11 (c) to Fig.11 (d), the proposed OCH gets competitive $m$AP score when $\tau$ is set to 0.2, which is set as a fixed parameter to all experiments. At last, due to the non-convex of object function in Eq.32, we evaluate the function error vs. the number of iterations. We initialize the parameter distribution with two common ways, e.g., Gaussian distribution $N(\mu, \tau)$ and Uniform distribution $U(a, b)$. As shown in Fig.11 (g), Fig.11 (h), and Tab. 5, the $m$AP of OCH is not sensitive to the initialization setting. Both initialization schemes are convergence in finite steps, which shows that the proposed OCH can find the optimized parameter quickly. The overall performance under Uniform distribution with an interval $[0, 1]$ is better than the others, which is used for all five datasets.

5 Conclusion

In this paper, we proposed a novel unsupervised hashing approach for large-scale nearest neighbor search, dubbed Ordinal Constraint Hashing (OCH), for large-scale nearest neighbor search. Unlike most previous unsupervised hashing, the proposed method exploits the ordinal constraint among data points, and preserves such relations into the Hamming space. Firstly, OCH adopted an ordinal constraint projection to significantly reduce the scale of the ordinal graph, which preserves the overall ordinal relation through a small set of anchors derived via clustering. Then, we propose a tensor ordinal graph to approximate the ordinal relations efficiently. In optimization, a novel iterative stochastic gradient descent algorithm on Stiefel manifold was further introduced. Finally, detailed theoretical analysis is given to explain why and how the proposed method can well preserve ordinal relations in the Hamming space. Extensive experiments on five benchmark datasets demonstrated that the proposed OCH method achieves the best performance in contrast with several representative and state-of-the-art methods [4], [10], [12], [14], [16], [17], [27]. In our future work, we will further extend the proposed method to the scenario of supervised hashing, as well as investigating the possibility of large-scale discrete optimization during the binary code learning.

6 Acknowledgement

This work is supported by the National Key R&D Program (No.2017YFC0113000, and No.2016YFB1001503), Nature Science Foundation of China (No.U1705262, No.61772443, and No.61572410), Post Doctoral Innovative Talent Support Program under Grant BX201600094, China Post-Doctoral Science Foundation under Grant 2017M612134, Scientific Research Project of National Language Committee of China (Grant No. YB135-49), and Nature Science Foundation of Fujian Province, China (No. 2017J01125).

References
