Discrete Spectral Hashing for Efficient Similarity Retrieval

Di Hu, Feiping Nie, and Xuelong Li, *Fellow, IEEE*

***Abstract*— To meet the required huge data analysis, organization, and storage demand, the hashing technique has got a lot of attention as it aims to learn an efficient binary representation from the original high-dimensional data. In this paper, we focus on the unsupervised spectral hashing due to its effective manifold embedding. Existing spectral hashing methods mainly suffer from two problems, i.e., the inefficient spectral candidate and intractable binary constraint for spectral analysis. To overcome these two problems, we propose to employ spectral rotation to seek a better spectral solution and adopt the alternating projection algorithm to settle the complex code constraints, which are therefore named as *Spectral Hashing with Spectral Rotation* and *Alternating Discrete Spectral Hashing*, respectively. To enjoy the merits of both methods, the spectral rotation technique is finally combined with the original spectral objective, which aims to simultaneously learn better spectral solution and more efficient discrete codes and is called as *Discrete Spectral Hashing*. Furthermore, the efficient optimization algorithms are also provided, which just take comparable time complexity to existing hashing methods. To evaluate the proposed three methods, extensive comparison experiments and studies are conducted on four large-scale data sets for the image retrieval task, and the noticeable performance beats several state-of-the-art spectral hashing methods on different evaluation metrics.**

***Index Terms*— Spectral rotation, discrete spectral hashing.**

I. Introduction

With the popularization of digital devices, the visual, audio, and text data can be easily recorded, transmitted, and stored. That is, we have already entered the big data era nowadays. To satisfy the required huge analysis, organization and storage demand in dealing with such big data, hashing technique are being paid more and more attention. This is because the existing digital devices are almost based on the binary mode, and hashing method exactly aims to learn efficient binary representation from the original high-dimensional data. Such projection can vastly reduce the storage space and processing time. These advantages make it widely used in various machine learning and computer vision problems, such as multi-view learning [1], multi-task learning [2], image retrieval [3], classification [4], and image patch matching [5].

Concretely, hashing focuses on projecting the original high-dimensional, real-valued data into the low-dimensional, binary codes while preserving the similarity structure across the data points in the database [6]. Among the numerous hashing techniques, the unsupervised ones draw great attention due to its advantages of label-free. To efficiently generate binary codes from the original data, one of the early representative unsupervised method, *Locality Sensitive Hashing* (LSH) [7], randomly projects the data into binary codes based on certain similarity measures, such as *p-norm* distance [8], and cosine similarity [9]. Besides, *Kernel LSH* (KLSH) [10] is also proposed for capturing more complex correlation. However, such works just utilize random projection without modeling the data structure in the original space, which results in the inefficient data-independent hashing function. In other words, more bits are required to achieve high recall and precision [11], which actually lies in the opposite direction of hashing objective.

To generate more compact codes, data-dependent hashing methods have attracted much attention recently [12]. These works develop the hashing functions by modeling the data structure in different view points. Andoni and Razenshteyn [13] extends the original LSH into the data-dependent scenario with theoretical guarantee. Iterative Quantization (ITQ) [14] aims to maximize the data variance of each bit via PCA projection, while Isotropic Hashing [15] targets to make the code dimensions into equal variance. Apart from the conventional linear projections, deep Binary Autoencoder (BA) network is also proposed to reconstruct the original data from the embedded short codes [16], which aims to learn a common semantic space [17]. These methods under different motivations attempt to capture different properties of the training data to achieve more efficient hashing function.

Recently, another category based on spectral graph shows considerable performance and is paid more attention, where the Laplacian eigenmap is employed to learn the intrinsic manifold structure and nonlinearly embed the data into proper codes. Concretely, *Spectral Hashing* (SH) [18] firstly introduces the spectral analysis into the hashing technique, which aims to make the similar items have similar binary codes. To accelerate the building of affinity matrix, Anchor Graph Hashing (AGH) [19] proposes to construct a sparse and low-rank neighborhood graph, then performs eigenmap

Manuscript received March 26, 2018; revised July 28, 2018; accepted September 30, 2018. Date of publication October 10, 2018; date of current version October 29, 2018. This work was supported in part by The National Key Research and Development Program of China under Grant 2018YFB1107400, and in part by The National Natural Science Foundation of China under Grant 61761130079, Grant 61772427, and Grant 61751202. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Husrev T. Sencar. *(Corresponding author: Feiping Nie.)*

D. Hu and F. Nie are with the School of Computer Science and Engineering, Northwestern Polytechnical University, Xi'an 710072, China (e-mail: hdui831@mail.nwpu.edu.cn; feipingnie@gmail.com).

X. Li is with the Xi'an Institute of Optics and Precision Mechanics, Chinese Academy of Sciences, Xi'an 710119, China (e-mail: xuelong_li@opt.ac.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIP.2018.2875312

based on it to acquire the spectral embedding. Different from AGH, Inductive Manifold Hashing (IMH) [20] proposes to embed the codes via t-SNE instead of Laplacian eigenmap. Although these spectral methods enjoy the merits of manifold embedding, the optimal hashing function is actually difficult to learn as it is intractable to optimize the spectral objective under the binary constraint [16].

To overcome the above weakness, most methods adopt a two-stage strategy to generate hashing codes. That is, they first optimize the spectral objective without the binary constraint, then perform the binarization over the real-valued spectral solution. As the binarization operation takes no consideration of learned manifold structure, it could destroy the embedded neighborhood structure and lower the code efficiency. To be able to directly generate the binary codes along with learning the low-dimensional structure, some researchers consider to employ a quantization term between the real-valued and binary codes to relax the binary constraint. Concretely, ITQ [14] reduces the quantization error between the rotated PCA-embedded data and the discrete codes, while Discrete Graph Hashing (DGH) [11] minimizes the distance between the real-valued set and the binary set, similarly for [21]. As the code constraints are actually performed on the real-valued solution instead of the hashing codes, the final codes are not efficient as expected, even lower than the binarized ones. On the other hand, there still remains another problem that the obtained spectral solution may deviate from the discrete one, as there is no guarantee that such yielded solution best approximates the binary codes. As a matter of fact, there exist numerous solutions to the relaxed spectral hashing objective and we wish to find a better one for code generation.

In this paper, we propose to extremely exploit the ability of unsupervised spectral hashing method in learning efficient binary codes. To accomplish such purpose, we need to solve the aforementioned problems step by step, i.e., the inefficient spectral solution and intractable binary constraint. To this end, our contributions are summarized as follows,

- We attempt to seek better spectral solution to the discrete codes instead of the original candidate. For such purpose, the spectral rotation technique is taken consideration into the graph hashing method, which could transform the candidate into a more proper one via a learnable orthogonal matrix. Meanwhile, to make the hashing codes satisfy the constraints, an efficient algorithm is proposed to solve the new spectral objective and corresponding hashing codes, which is named as *Spectral Hashing with Spectral Rotation*[1] (SHSR).
- To directly learn the compact binary codes and avoid the quantization error caused by the relaxation of discrete constraint, we propose to divide all the code constraints into two sets and perform alternating projection over them in an iteratively re-weighted framework, which is named as *Alternating Discrete Spectral Hashing* (ADSH). Such method can reduce the intractable discrete spectral problem into two easy-to-handle sub-problems that can be effectively solved by off-the-shelf methods.
- As the above two methods just focus on the partial weakness of spectral hashing, it is highly expected to enjoy both advantages in learning hashing function. Hence, we propose to simultaneously learn better spectral solution with the rotation technique and generate more efficient discrete codes under required code constraints, which is called as *Discrete Spectral Hashing* (DSH).
- Extensive experiments are conducted on four large-scale benchmark datasets, which also include the evaluations in the code efficiency, parameter effects, and out-of-sample strategies. The results show that the three proposed methods effectively solve the conventional problems of spectral hashing, so that they can learn more efficient codes than other methods on various metrics.

In the following sections, we first revisit the conventional spectral-based hashing methods in Section 2. In Section 3, we perform the spectral rotation technique over the original spectral solution for seeking more efficient codes. Then we propose the alternating projection algorithm for directly solving the discrete hashing problem in Section 4. Section 5 introduces an united framework to integrate the merits of above two methods and the detailed optimization is also provided. Extensive experiments are conducted for evaluating the three methods on the benchmark datasets in Section 6. In the end, Section 7 concludes this paper.

II. Spectral Hashing Revisited

Unsupervised hashing targets to map the high-dimensional data $\mathbf{x}_i \in R^d$ into the binary codes $\mathbf{b}_i \in \{-1, 1\}^r$, while preserving the similarity structure across the data points $\{\mathbf{x}_i\}_{i=1}^n$. As the Hamming distance indicates the similarity between binary codes, the hashing objective function can be formulated as

$$\begin{aligned} &\min_{\mathbf{B}} \sum_{i,j=1} W_{ij} \left\| \mathbf{b}_i - \mathbf{b}_j \right\|^2 \\ &s.t.\ \mathbf{B} \in \{-1, 1\}^{n \times r}, \quad \mathbf{B}^T\mathbf{B} = n\mathbf{I},\ \mathbf{B}^T\mathbf{1} = 0, \end{aligned} \tag{1}$$

where $\mathbf{B} = [\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n]^T$, $\mathbf{W}$ is the affinity matrix indicating the similarity between each two data points in $\{\mathbf{x}_i\}_{i=1}^n$. To make the codes efficient, the three constraints require the codes to be discrete, orthogonal, and balanced (i.e., each bit to have equal chance to be -1 or 1).

Theorem 1: For a single bit, solving Eq. 1 is equivalent to balanced graph partitioning and is NP-hard.

However, according to [18, Th. 1], eq. 1 is difficult to solve. A tractable approach is to turn Eq. 1 into a spectral problem, which can be written as

$$\begin{aligned} &\min_{\mathbf{B}}\ Tr\left(\mathbf{B}^T(\mathbf{D} - \mathbf{W})\mathbf{B}\right) \\ &s.t.\ \mathbf{B} \in \{-1, 1\}^{n \times r}, \quad \mathbf{B}^T\mathbf{B} = n\mathbf{I},\ \mathbf{B}^T\mathbf{1} = 0, \end{aligned} \tag{2}$$

where $\mathbf{D}$ is a diagonal matrix whose entries $d_{ii} = \sum_j W_{ij}$, $\mathbf{L} = \mathbf{D} - \mathbf{W}$ is the graph Laplacian matrix [23], and Tr denotes the trace operation of matrix.

Although the original problem is turned into the spectral hashing problem, both of them suffer from the high time complexity of constructing the affinity matrix $\mathbf{W}$, which

[1]The preliminary version [22] has been accepted by AAAI 2017.

is $O\left(dn^2\right)$. With the increasing dimensionality d and samples n of the original data, these methods have to be faced with a huge training time. A possible way to overcome such weakness is to construct a neighborhood graph to approximate the affinity matrix $\mathbf{W}$, which can be performed in the linear time of number of samples [24]. To approximate the underlying neighborhood structure, a small set of $m \ll n$ points are first chosen as the anchors $\mathcal{U} = \left\{\mathbf{u}_j \in R^d\right\}_{j=1}^{m}$, which are usually the centers of clusters. Then a kind of local non-linear measurement between data points and anchors is conducted as $Z_{ij} = K\left(\mathbf{x}_i, \mathbf{u}_j\right)/N$, where $K(\cdot)$ is the distance function used to measure the similarity between data $\mathbf{x}_i$ and anchor $\mathbf{u}_j$, such as ℓ_2 distance in Gaussian kernel space, j is one of $s \ll m$ nearest anchors to current data $\mathbf{x}_i$. And $N = \sum_{j\in\langle i\rangle} K\left(\mathbf{x}_i, \mathbf{u}_j\right)$. Finally, the normalized matrix $\mathbf{Z} \in R^{n\times m}$ gives the approximated low-rank affinity matrix $\mathbf{A} = \mathbf{Z}\mathbf{\Lambda}^{-1}\mathbf{Z}^T$, where $\mathbf{\Lambda} = \mathrm{diag}(\mathbf{Z}^T\mathbf{1})$.

As the affinity matrix $\mathbf{A}$ has been normalized by Λ^{-1}, all summation of the columns and rows are equal to one and the Laplacian matrix becomes $\mathbf{L} = \mathbf{I} - \mathbf{A}$. Hence, Eq. 2 becomes a maximization problem, i.e.,

$$\begin{aligned} &\max_{\mathbf{B}} \ Tr\left(\mathbf{B}^T\mathbf{A}\mathbf{B}\right) \\ &s.t.\ \mathbf{B} \in \{-1, 1\}^{n\times r}, \quad \mathbf{B}^T\mathbf{B} = n\mathbf{I}, \ \mathbf{B}^T\mathbf{1} = 0. \end{aligned} \tag{3}$$

In fact, although the construction of affinity matrix has been accelerated, Eq. 3 is still hard to solve due to the binary constraint. A common strategy is to abandon the constraint and obtain the spectral solution $\mathbf{F}$ by performing the eigen-decomposition over the similarity matrix $\mathbf{A}$, where the r eigenvectors of the matrix $\mathbf{A}$ with maximal eigenvalue (excluding the eigenvector with eigenvalue one) are treated as the spectral codes $\mathbf{F}$. On the one hand, as these eigenvectors are orthogonal to each other, they satisfy the orthogonal constraint after multiplying the scale $\sqrt{n}$. On the other hand, the excluded eigenvector with eigenvalue 0 is $\mathbf{1}$ if the constructed graph L is connected. As all the other eigenvectors are orthogonal to it, the obtained spectral solutions satisfy $\mathbf{F}^T\mathbf{1} = 0$. To generate the final binary codes $\mathbf{B}$, the binarization operation with threshold "0" is performed over the solution $\mathbf{F}$. Such codes obtained by the two-stage method could lower the code efficiency, as the rounding operation may result in the improving error with the increasing code length r and even break the last two constraints.

To generate more efficient binary codes, the discrete constraint $\mathbf{B} \in \{-1, 1\}^{n\times r}$ is taken consideration again. Recently, DGH [11] proposes to relax the binary constraint and append a penalty term of the quantization error between real-valued and binary codes. Although the discrete codes can be directly generated in such framework, the codes are not efficient as expected. This is because the last two constraints are still performed on the real-valued ones. While RDSH [25] introduces an additional quantization error into the original spectral hashing objective but without the orthogonal and balanced constraint, which is therefore just comparable to the SH-methods. In this paper, to address the above weakness further, we propose to solve the original spectral hashing problem (i.e., Eq. 3) step by step, and aim to learn more efficient binary codes.

III. Spectral Rotation Hashing

A. Spectral Hashing With Spectral Rotation

As the conventional spectral hashing objective is a NP-hard problem, the practical strategy is to relax it into

$$\begin{aligned} &\max_{\mathbf{F}} \ Tr(\mathbf{F}^T\mathbf{A}\mathbf{F}) \\ &s.t.\ \mathbf{F} \in R^{n\times k}, \quad \mathbf{F}^T\mathbf{F} = n\mathbf{I}, \ \mathbf{F}^T\mathbf{1} = 0. \end{aligned} \tag{4}$$

As the discrete constraint is abandoned, the required second-stage of binarization is performed over the spectral solution $\mathbf{F}$. In fact, due to the binary property, the final coding matrix $\mathbf{B}$ obtained by the sign function over $\mathbf{F}$ is the optimal solution of minimizing the quantization error of their Euclidean distance. However, it is difficult to guarantee that such yielded spectral solution best approximates the efficient binary codes, as there are still numerous solutions fitting to Eq. 4. Hence, can we find a closer real-valued solution to the discrete one?

In fact, for any $\mathbf{F}$, $\mathbf{FQ}$ is another solution to Eq. 4, where $\mathbf{Q}$ is an arbitrary orthogonal matrix. Hence, we hope we can find a better $\mathbf{FQ}$ that is closer to the binary codes $\mathbf{B}$, which can be formulated as

$$\begin{aligned} &\min_{\mathbf{B},\mathbf{Q}} \ \|\mathbf{F}\mathbf{Q} - \mathbf{B}\|_F^2 \\ &s.t.\ \mathbf{B} \in \{-1, 1\}, \quad \mathbf{B}^T\mathbf{1} = 0, \quad \mathbf{Q}^T\mathbf{Q} = \mathbf{I}, \end{aligned} \tag{5}$$

where the Frobenius norm is employed to measure the distance from the discrete solution $\mathbf{B}$ to the revised spectral solution $\mathbf{FQ}$. Different from the constraint violation of $\mathbf{B}$ in the previous methods, Eq. 5 still preservers the balanced constraint and just relaxes the orthogonal constraint, which could make the code be more efficient.

The objective function in Eq. 5 can be iteratively optimized by solving the following problems

When $\mathbf{Q}$ is fixed, Eq. 5 w.r.t. $\mathbf{B}$ can be re-written as

$$\begin{aligned} &\min_{\mathbf{B}} \ \|\mathbf{B} - \mathbf{M}\|_F^2 = \sum_j \left\|\mathbf{b}_j - \mathbf{m}_j\right\|_2^2 \\ &s.t.\ \mathbf{B} \in \{-1, 1\}^{n\times r}, \quad \mathbf{B}^T\mathbf{1} = 0, \end{aligned} \tag{6}$$

where $\mathbf{b}_j$ and $\mathbf{m}_j$ are the $j-$column of matrix $\mathbf{B}$ and $\mathbf{M} = \mathbf{FQ}$, respectively. As the binary constraint makes the term of $\mathbf{b}_j^T\mathbf{b}_j$ equal to n, minimizing Eq. 6 is equivalent to maximizing the dot product of the column pair $\mathbf{b}_j$ and $\mathbf{m}_j$. Concretely, for each column $\mathbf{b}_j$,

$$\begin{aligned} &\max_{\mathbf{b}_j} \ \mathbf{b}_j^T\mathbf{m}_j \\ &s.t.\ \mathbf{b}_j \in \{-1, 1\}^{n\times 1}, \quad \mathbf{b}_j^T\mathbf{1} = 0. \end{aligned} \tag{7}$$

As the elements of $\mathbf{b}_j$ have to be -1 or 1 equably, the optimal solution of $\mathbf{b}_j$ can be directly obtained by sorting $\mathbf{m}_j$ in descending order and then assigning the binary value for each half part, which can be written as

$$\mathrm{b}_{ij} = \begin{cases} 1, & \mathrm{q}(\mathrm{m}_{ij}) \le n/2 \\ -1, & \text{otherwise}, \end{cases} \tag{8}$$

Algorithm 1 Spectral Hashing With Spectral Rotation

Input: Training data $\mathbf{X} \in R^{n\times d}$, affinity matrix $\mathbf{A}$.
Output: The binary codes $\mathbf{B} \in \{-1, 1\}^{n\times r}$.

1: Solve Eq. 4 and obtain the candidate real-valued solution $\mathbf{F} \in R^{n\times r}$.
2: Initial an arbitrary orthogonal matrix $\mathbf{Q} \in R^{r\times r}$.
3: Fix $\mathbf{Q}$, assign 1 or -1 to the elements of each column of $\mathbf{B}$ by Eq. 8.
4: Fix $\mathbf{B}$, update $\mathbf{Q}$ and obtain better real-valued solution by Eq. 9.
5: Repeat above 3-4 steps until convergence or N steps.

where $\mathrm{q}(\mathrm{m}_{ij})$ stands for the order of m_{ij} after sorting. Then the assigned vector $\mathbf{b}_j$ constitutes the matrix $\mathbf{B}$ as a column.

When $\mathbf{B}$ is fixed, Eq. 5 w.r.t. $\mathbf{Q}$ can be written as

$$\max_{\mathbf{Q}} \ Tr\left(\mathbf{GQ}\right) \quad s.t.\ \mathbf{Q}^T\mathbf{Q} = \mathbf{I}, \tag{9}$$

where $\mathbf{G} = \mathbf{B}^T\mathbf{F}$. The analytical solution to Eq. 9 can be directly obtained via Theorem 2.

Theorem 2: $\boldsymbol{Q} = \boldsymbol{VU}^T$ is the optimal solution to the given objective function in Eq. 9, where $\boldsymbol{U}$ and $\boldsymbol{V}$ are the left and right singular vectors of the compact Singular Value Decomposition (SVD) of $\boldsymbol{G}$.[2]

The proposed optimization of $\mathbf{B}$ and $\mathbf{Q}$ are executed iteratively until satisfying the convergence criteria, i.e. the unchanged binary code matrix $\mathbf{B}$. We summarize the proposed *Spectral Hashing with Spectral Rotation* (SHSR) in Algorithm 1. And the time complexity of spectral rotation is $O\left(2nr^2N + rnN\log_2 n\right)$, where the former part corresponds to solving $\mathbf{Q}$, the latter is for solving $\mathbf{B}$, and N is the budget iteration number and set to 20 in this paper. As the code length r is a tiny number compared with n, SHSR can be achieved in a few seconds.

B. Out-of-Sample

The above compact codes are all learned from the training data $\{\mathbf{x}_i\}_{i=1}^n$, but it is necessary to generate efficient codes $\mathbf{b}\left(\mathbf{x}^*\right)$ for the new data $\mathbf{x}^*$ beyond the dataset. As the binarization strategy could result in roughly approximated binary codes, the real-valued solution is considered in order to ensure the code efficiency instead of the discrete one,[3]

$$\min_{\mathbf{b}(\mathbf{x}^*)\in\{-1,1\}^r} \ \sum_{i=1}^{n} \mathrm{a}\left(\mathbf{x}^*, \mathbf{x}_i\right) \left\|\mathbf{b}\left(\mathbf{x}^*\right) - \mathbf{m}_i\right\|_2^2, \tag{10}$$

where $\mathbf{A}\left(\mathbf{x}^*, \mathbf{x}\right)$ is the affinity vector between $\mathbf{x}^*$ and all the data points $\{\mathbf{x}_i\}_{i=1}^n$, which can be directly obtained via $\mathbf{Z}\boldsymbol{\Lambda}^{-1}\mathbf{z}\left(\mathbf{x}^*\right)$. As the generated codes are binary, Eq. 10 is equivalent to

$$\max_{\mathbf{b}(\mathbf{x}^*)\in\{-1,1\}^r} \ \left\langle \mathbf{b}\left(\mathbf{x}^*\right), \left(\mathbf{FQ}\right)^T\mathbf{Z}\boldsymbol{\Lambda}^{-1}\mathbf{z}\left(\mathbf{x}^*\right)\right\rangle. \tag{11}$$

[2]The proof can be found in the appendix
[3]Experimental study is performed in the following sections.

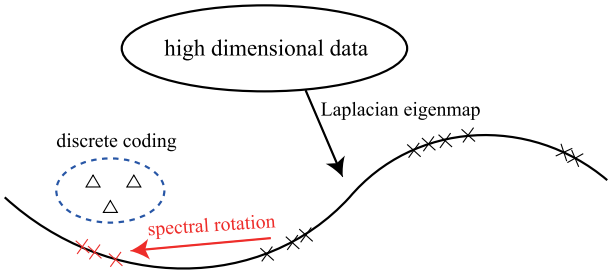

Fig. 1. The manifold learning perspective of spectral rotation. Suppose that the training data has been embedded into the low-dimensional manifold in the black cross points. The red cross points represent the solution transformed by spectral rotation, which are closer to the corresponding discrete codes represented in triangles.

As $\mathbf{b}\left(\mathbf{x}^*\right)$ is restricted to be binary value, the optimal solution to Eq. 11 becomes

$$\mathbf{b}\left(\mathbf{x}^*\right) = \mathrm{sgn}\left(\mathbf{Pz}\left(\mathbf{x}^*\right)\right), \tag{12}$$

where $\mathrm{sgn}(\cdot)$ is the sign function. Note that $\mathbf{P} = \left(\mathbf{FQ}\right)^T\mathbf{Z}\boldsymbol{\Lambda}^{-1}$ can be pre-computed, hence it is feasible to make the out-of-sample hashing more efficient.

C. Manifold Learning Perspective

The conventional spectral hashing embeds the high-dimensional features on a low-dimensional manifold via the Laplacian eigenmap, as shown in Fig. 1. The nonlinear eigenmap is expected to find the inherent manifold structure of original features, while preserving the neighborhood similarity. Unfortunately, even if the neighborhood manifold is successfully learned, the second procedure of discrete codes binarization could still mix them with other non-adjacent embedded points. In such circumstance, spectral rotation aims to derive better spectral solution to the corresponding discrete ones by taking advantage of the orthogonal matrix $\mathbf{Q}$, while preserving the similarity structure. For example, Fig. 1 shows that it transforms the original embedded points $\mathbf{F}$ into the new ones $\mathbf{FQ}$ that are closer to discrete coding (in terms of ℓ_2) via the red arrow (i.e., the orthogonal transformation $\mathbf{Q}$). Hence, it is reasonable that the transformed spectral solution becomes more efficient in generating the hashing codes. Moreover, it is also important to note that spectral rotation is obviously different from the rotation technique in ITQ. First, ITQ is just a kind of data rotation after linear PCA projection, while spectral rotation is performed with spectral embedding and aims to seek better solution while maintaining the manifold structure. Second, spectral rotation aims to shrink the distance between real-valued codes and discrete ones as introduced above, which is different from the objective of balanced variance in ITQ.

IV. Alternating Discrete Spectral Hashing

Although SHSR can transform the candidate spectral solution into proper positions via the efficient spectral rotation technique, it is essentially a kind of two-stage methods. Hence, it suffers from the same defect as the previous methods in dealing with the binary constraint. Moreover, when seeking better spectral solution by Eq. 5, the original three code constraints

are not simultaneously kept, where the orthogonal constraint is relaxed via the spectral solution. As such weaknesses may lower the code efficiency, we hope to directly solve the spectral hashing problem.

It is well known that the conventional spectral hashing objective is difficult to solve under such complex constraints. Fortunately, the re-weighted framework provides the possibility to turn it into an easier problem without any conditional hypothesis on the constraints.[4] That is, we can treat Eq. 3 as a composite function of $h\left(g\left(\mathbf{x}\right)\right)$, where $g\left(\mathbf{x}\right)=\mathbf{x}$ and $h\left(\mathbf{x}\right)=Tr\left(\mathbf{x}^T\mathbf{A}\mathbf{x}\right)$. As $h\left(\mathbf{x}\right)$ is a convex function in the domain of $g\left(\mathbf{x}\right)$, we can calculate its supergradient and multiple it by $g\left(\mathbf{x}\right)$, i.e.,

$$\begin{aligned}&\max_{\mathbf{B}}\ Tr\left(\mathbf{B}^T\mathbf{S}\right)\\&s.t.\ \mathbf{B}\in\{-1,1\}^{n\times r},\quad \mathbf{B}^T\mathbf{B}=n\mathbf{I},\ \mathbf{B}^T\mathbf{1}=0,\end{aligned}\tag{13}$$

where $\mathbf{S}=2\mathbf{A}\mathbf{B}$ is obtained by taking the derivative of Eq. 3 w.r.t $\mathbf{B}$ and viewed as the weight in Eq. 13. As the weight $\mathbf{S}$ is fixed in each iteration, Eq. 13 can be re-written as

$$\begin{aligned}&\min_{\mathbf{B}}\ \|\mathbf{B}-\mathbf{S}\|_F^2\\&s.t.\ \mathbf{B}\in\{-1,1\}^{n\times r},\quad \mathbf{B}^T\mathbf{B}=n\mathbf{I},\ \mathbf{B}^T\mathbf{1}=0.\end{aligned}\tag{14}$$

The three maintained code constraints make Eq. 14 difficult to solve, especially the binary one. To make it tractable, we propose to employ alternating projection w.r.t. these constraint sets. Specifically, the three constraints are divided into two sets, i.e., $\pi_1=\left\{\mathbf{B}|\mathbf{B}\in\{-1,1\}^{n\times r},\mathbf{B}^T\mathbf{1}=0\right\}$ and $\pi_2=\left\{\mathbf{B}|\mathbf{B}^T\mathbf{B}=n\mathbf{I}\right\}$. Then, Eq. 14 is optimized by alternately solving the following two sub-problems.

The first sub-problem w.r.t. π_1 set can be re-written as

$$\begin{aligned}&\min_{\mathbf{B}}\ \|\mathbf{B}-\mathbf{S}\|_F^2\\&s.t.\ \mathbf{B}\in\{-1,1\}^{n\times r},\quad \mathbf{B}^T\mathbf{1}=0.\end{aligned}\tag{15}$$

It is obvious that Eq. 15 is the same as the first sub-problem of SHSR, so the sorting algorithm can be directly employed for solving it but based on the magnitude of $\mathbf{S}$,

$$\mathrm{b}_{ij}=\begin{cases}1, & \mathrm{q}(\mathrm{s}_{ij})\le n/2\\ -1, & \text{otherwise},\end{cases}\tag{16}$$

where $\mathrm{q}(\mathrm{s}_{ij})$ stands for the order of s_{ij} after sorting.

The second sub-problem w.r.t. π_2 set can be re-written as

$$\begin{aligned}&\max_{\mathbf{B}}\ Tr\left(\mathbf{B}^T\mathbf{S}\right)\\&s.t.\ \mathbf{B}^T\mathbf{B}=n\mathbf{I}.\end{aligned}\tag{17}$$

Different from Eq. 9, the orthogonal constraint is replaced with $\mathbf{B}^T\mathbf{B}=n\mathbf{I}$ due to the binary property. However, according to Theorem 2, we can still obtain the optimal solution of Eq. 17 by multiplying the modified singular value $\sqrt{n}$. That is, $\mathbf{B}=\sqrt{n}\mathbf{U}\mathbf{V}^T$, where $\mathbf{U}$ and $\mathbf{V}$ are the left and right singular vectors of the SVD of $\mathbf{S}$.

The optimization w.r.t. the two constraint sets $\{\pi_1,\pi_2\}$ are performed alternately, which is named as *Alternating Discrete Spectral Hashing* (ADSH) and summarized in Algorithm 2. The proposed algorithm holds the time complexity of $O\left(2nr^2N+nrN\log_2 n+nmrN\right)$, where N is the iteration number and set to 30 in the experiments. Note that, $O\left(nrN\log_2 n\right)$ corresponds to assigning the codes according to the ranking result, and $O\left(2nr^2N\right)$ is about updating the coding matrix $\mathbf{B}$ by solving Eq. 17. As the constructed affinity matrix $\mathbf{A}$ is both low-rank (at most m) and sparse, the update of weight $\mathbf{S}$ enjoys low time complexity of $O\left(nmrN\right)$.

[4]More theoretical analysis can be found in [26].

Algorithm 2 Alternating Discrete Spectral Hashing

Input: Training data $\mathbf{X}\in R^{n\times d}$, affinity matrix $\mathbf{A}$.
Output: The code matrix $\mathbf{B}\in\{-1,1\}^{n\times r}$.

1: Initial an orthogonal matrix $\mathbf{F}\in R^{n\times r}$, then rounding it into candidate code matrix $\mathbf{B}$ via zero.
2: Assign binary codes to each column of $\mathbf{B}$ by Eq. 16.
3: Update $\mathbf{B}$ by solving Eq. 17.
4: Update $\mathbf{S}=2\mathbf{A}\mathbf{B}$.
5: Repeat above 2-4 steps until convergence or N steps.

In fact, the proposed ADSH is based on alternating projection that is a simple optimization algorithm performed on convex sets [27]. Although lots of works have confirmed its effectiveness [27], including our method, it is hard to theoretically guarantee the convergence when faced with the non-convex property of the constraint sets. In the next section, we propose a simple but at the same time more efficient method to directly learn the binary codes, which avoids the weakness of alternating projection algorithm.

Similar with the strategy of SHSR in generating the hashing codes for out-of-samples, ADSH also performs the similarity measurement in the hamming space but without the rotation matrix $\mathbf{Q}$. Hence, Eq. 11 becomes

$$\max_{\mathbf{b}(\mathbf{x}^*)\in\{-1,1\}^r}\ \left\langle\mathbf{b}\left(\mathbf{x}^*\right),\mathbf{B}^T\mathbf{Z}\mathbf{\Lambda}^{-1}\mathbf{z}\left(\mathbf{x}^*\right)\right\rangle.\tag{18}$$

Then, the binary codes can be derived as $\mathbf{b}\left(\mathbf{x}^*\right)=\mathrm{sgn}\left(\mathbf{P}\mathbf{z}\left(\mathbf{x}^*\right)\right)$, where $\mathbf{P}=\mathbf{B}^T\mathbf{Z}\mathbf{\Lambda}^{-1}$ can also be pre-computed for acceleration.

V. Discrete Spectral Hashing

Although ADSH has attempted to solve the original spectral hashing objective, it has to be faced with the non-convex constraint of orthogonality. Meanwhile, the efficient spectral rotation technique is also not considered within ADSH. Hence, to simultaneously address the above problems and enjoy the bilateral merits, we propose to solve the Laplacian eigenmap problem while learning efficient binary codes via the quantization of spectral rotation. The objective is written as

$$\begin{aligned}&\min_{\mathbf{F},\mathbf{B},\mathbf{Q}}\ Tr(\mathbf{F}^T\mathbf{L}\mathbf{F})+\alpha\|\mathbf{F}\mathbf{Q}-\mathbf{B}\|_F^2\\&s.t.\ \mathbf{F}^T\mathbf{F}=n\mathbf{I},\quad \mathbf{Q}^T\mathbf{Q}=\mathbf{I},\ \mathbf{B}\in\{-1,1\}^{n\times r},\ \mathbf{B}^T\mathbf{1}=0,\end{aligned}\tag{19}$$

where α is a tuning parameter. Intuitively, Eq. 19 is the weighted combination of the two-stage method of SHSR. However, Eq. 19 actually can be viewed as an united spectral hashing framework that involves the above two proposed

methods. Concretely, when $\alpha \to 0$, to effectively generate the discrete codes, we can employ the two-stage SHSR instead. On the contrary, when $\alpha \to \infty$, Eq. 19 turns into the original spectral hashing problem, i.e., Eq. 3, where ADSH can be employed for solving it. However, when α is set to an arbitrary constant, Eq. 19 can not be directly optimized by the aforementioned algorithms. In fact, if we simply take $\widetilde{\mathbf{F}} = \mathbf{FQ}$, Eq. 19 can be equivalently transformed into

$$\begin{aligned} &\min_{\mathbf{F},\mathbf{B}} \; Tr(\mathbf{F}^T\mathbf{L}\mathbf{F}) + \alpha \left\|\mathbf{F} - \mathbf{B}\right\|_F^2 \\ &s.t.\ \mathbf{F}^T\mathbf{F} = n\mathbf{I}, \quad \mathbf{B} \in \{-1, 1\}^{n\times r},\ \mathbf{B}^T\mathbf{1} = 0. \end{aligned} \tag{20}$$

In Eq. 20, the discrete and balanced constraint are directly performed over the hashing codes, while the orthogonal constraint is relaxed to the spectral one due to the difficult optimization. By contrast, DGH relaxes and transfers all the expected constraints to the spectral solution, while the discrete codes are obtained by gradient ascent instead of the Laplacian eigenmap, hence, the generated codes cannot enjoy the same efficiency as DSH. And RDSH even does not take consideration of the constraints within its objective.

The DSH objective can be iteratively optimized w.r.t. $\mathbf{F}$ and $\mathbf{B}$, which leads to the following two sub-problems.

When $\mathbf{B}$ is fixed, then Eq. 20 becomes

$$\max_{\mathbf{F}^T\mathbf{F}=n\mathbf{I}} \; Tr(\mathbf{F}^T\mathbf{A}\mathbf{F}) + 2\alpha Tr(\mathbf{F}^T\mathbf{B}). \tag{21}$$

Obviously, Eq. 21 is a quadratic problem defined on the Stiefel manifold,[5] which aims to find the approximated binary eigenvector of the matrix $\mathbf{A}$. Although the power iteration is an efficient iterative method to compute the dominant eigenvalue and corresponding eigenvector of arbitrary symmetric matrix [28], it can not be directly used for solving Eq. 21 due to the existing quantization term. More precisely, the Lagrangian function for such quadratic problem can be written as

$$L(\mathbf{F}, \mathbf{\Lambda}) = Tr(\mathbf{F}^T\mathbf{A}\mathbf{F}) + 2\alpha Tr(\mathbf{F}^T\mathbf{B}) - Tr((\mathbf{F}^T\mathbf{F} - n\mathbf{I})\mathbf{\Lambda}). \tag{22}$$

By setting the derivative w.r.t. $\mathbf{F}$ to zero, we can get the KKT condition of Eq. 21,

$$\frac{\partial L(\mathbf{F}, \mathbf{\Lambda})}{\partial \mathbf{F}} = 2\mathbf{A}\mathbf{F} + 2\alpha\mathbf{B} - 2\mathbf{F}\mathbf{\Lambda} = 0. \tag{23}$$

However, it is difficult to directly solve $\mathbf{F}$. Recently, *Generalized Power Iteration* (GPI) [29] proposes to impose the re-weighted methods into the power iteration algorithm. Concretely, GPI takes the derivative of Eq. 21 w.r.t $\mathbf{F}$ as the weighting parameter for current spectral solution in each power iteration, then updates the solution for the next weighted trace problem. The detailed GPI algorithm is summarized in Algorithm 3.

When $\mathbf{F}$ is fixed, then Eq. 20 becomes

$$\begin{aligned} &\min_{\mathbf{B}} \; \left\|\mathbf{F} - \mathbf{B}\right\|_F^2 \\ &s.t.\ \mathbf{B} \in \{-1, 1\}^{n\times r}, \quad \mathbf{B}^T\mathbf{1} = 0. \end{aligned} \tag{24}$$

[5]The orthonormal column in the Stiefel manifold, $\mathbf{F}^T\mathbf{F} = \mathbf{I}$, is replaced by $\mathbf{F}^T\mathbf{F} = n\mathbf{I}$ due to the binary property.

Algorithm 3 Generalized Power Iteration

Input: The affinity matrix $\mathbf{A} \in R^{n\times n}$, the matrix $\mathbf{B} \in \{-1, 1\}^{n\times r}$

Output: The matrix $\mathbf{F} \in R^{n\times r}$.

1: Initial an orthogonal matrix $\mathbf{F} \in R^{n\times r}$
2: Update $\mathbf{M} = 2\mathbf{A}\mathbf{F} + 2\alpha\mathbf{B}$.
3: Perform the compact SVD over $\mathbf{M}$, i.e., $\mathbf{M} = \mathbf{U}\mathbf{S}\mathbf{V}^T$
4: Update $\mathbf{F} = \sqrt{n}\mathbf{U}\mathbf{V}^T$.
5: Repeat above 2-4 steps until convergence or N_G steps.

Algorithm 4 Discrete Spectral Hashing

Input: Training data $\mathbf{X} \in R^{n\times d}$, affinity matrix $\mathbf{A}$.

Output: The code matrix $\mathbf{B} \in \{-1, 1\}^{n\times r}$.

1: Initial an orthogonal matrix $\mathbf{F} \in R^{n\times r}$
2: Update $\mathbf{F}$ via the GPI in Algorithm 3.
3: Assign binary codes to each column of $\mathbf{B}$ according to the magnitude of columns of $\mathbf{F}$.
4: Repeat above 2-3 steps until convergence or N steps.

Similar with the aforementioned two spectral-based methods, Eq. 24 can be effectively solved by the sorting algorithm. Then the whole optimization w.r.t. $\mathbf{F}$ and $\mathbf{B}$ are executed iteratively and summarized in Algorithm 4, which is named as *Discrete Spectral Hashing* (DSH). As GPI and the sorting algorithm can find proper local and global optimum, the bounded objective of Eq. 20 can monotonically converge to a stationary point.

Although DSH employs an iteration manner to learn the hashing codes, it enjoys comparable time complexity of $O\left(nmrN_GN + nrN\log_2 n\right)$, where N_G and N are the budget iteration numbers of GPI and the whole DSH method, respectively. The former part relates to solving the spectral solution via the GPI algorithm, while the latter one corresponds to the quantization term in Eq. 20. Similar with the above ADSH, in view of the low-rank (at most m) and sparse matrix $\mathbf{A}$, multiplying it with $\mathbf{F}$ can be efficiently performed. As for the hashing codes generation for the out-of-samples, the same strategy as ADSH is adopted, i.e., Eq. 18.

Table I shows the comparison of time complexity between conventional spectral-based methods and our proposed methods, where the complexity of constructing the affinity matrix $\mathbf{A}$ is not included. It is obviously that AGH enjoys the lowest complexity, as it directly solves the spectral solution via eigen-decomposition. By contrast, RDSH has to solve a standard Sylvester equation when seeking the spectral solution $\mathbf{F}$, hence, it takes the largest one of $O\left(n^3\right)$. And the rest four methods are all based on the iterative optimization for the spectral-based objectives. As the hyper-parameter of m, r and N are independent of the dataset size, they enjoy linear training time with n. Note that, the term of nr^2N in DGH is much larger than the term of $nrN\log_2 n$ in our proposed methods, as the hashing codes r usually takes dozens of bits to encode features, which is greater than $\log_2 n$. Meanwhile, as our methods need less iterations to converge, they take lower complexity compared with DGH.

TABLE I
THE TIME COMPLEXITY COMPARISON AMONG EXISTING SPECTRAL HASHING METHODS

Methods	Time complexity
AGH [19]	$O\left(m^2n + (s+1)\,rn\right)$
DGH [11]	$O\left(nmrN_xN + r^2nN\right)$
RDSH [25]	$O\left(n^3\right)$
SHSR	$O\left(m^2n + (s+1)\,rn + 2nr^2N + nrN\log_2 n\right)$
ADSH	$O\left(2nr^2N + nrN\log_2 n + nmrN\right)$
DSH	$O\left(nmrN_GN + nrN\log_2 n\right)$

VI. EXPERIMENTS

A. Dataset

Four benchmark datasets are chosen for evaluation, including MNIST[6] [31], CIFAR-10[7] [32], YouTube Faces[8] [33], and NUS-WIDE[9] [34].

MNIST dataset consists of 70,000 images of handwritten digits from "0" to "9". These images are all 28×28 pixels, which result in the 784D feature vectors for representation. The testing set consists of 100 samples of each digit, which are randomly sampled. The remaining samples constitute the training set.

CIFAR-10 is a collection of 60,000 tiny images, which consists of ten object categories (6000 images per category). And 512D GIST vector [35] are pre-extracted from these images and used as the image feature for evaluation. This dataset is split into a training set of 59,000 samples and a testing set of 1,000 samples (100 samples per object).

YouTube Faces is a database of 3,425 face videos captured from 1,595 different people. Similar with the setting of [11], a new subset is constructed by selecting the person who has at least 500 face images, which results in 370,319 samples. And 1,770D LBP vector [36] is also pre-extracted for representing images. For this subset, the testing set consists of 3,800 images from 38 people who have more than 2,000 images, and we uniformly sample 100 images from each people. The remaining samples constitute the training set.

NUS-WIDE consists of 269,648 multi-label images. Different from the above three datasets whose samples with the same label are usually considered as the true neighbors, the true neighbors in the NUS-WIDE are defined as whether two images share at least one label. The common 21 labels are considered in our experiments and the images are represented into 500D bag-of-words based on SIFT. 100 images are uniformly sampled from each label and constitute the testing set, and the rest images are served as the training set.

B. Metric

The conventional evaluation metrics, Hamming ranking and hash lookup, are both adopted. Specifically, Hamming ranking focuses on the quality of whole retrieved items, while hash lookup just deals with the top retrieved items. Hence, Hamming ranking computes the Mean Average Precision (MAP) based on the Hamming distance to a query, and hash lookup computes the $\{Recall, F-measure\}$ score according to the retrieved items within a Hamming ball of radius 2 to the query.

As there is no label information considered within the proposed hashing methods, it is more sensible to directly evaluate the quality of preserved nearest neighbors after hashing projection. Hence, we propose to compute the average distance (in the Euclidean space) of the original data that corresponds to the top-k retrieved items. The better preserved neighbors will take a smaller distance value.

C. Comparison Experiments

The proposed three methods aim to deal with different problems of spectral hashing, hence it is expected to evaluate the improved performance by comparing them with specific methods accordingly. Firstly, as the spectral hashing methods (i.e., SH, AGH-1, and AGH-2) just treat the candidate spectral solution as the final ones, we compare them with SHSR to validate the effectiveness of spectral rotation. Then, the two-stage hashing methods of ITQ, *Scalable Graph Hashing* (SGH) [30], and IMH (IMH-tSNE) are chosen for comparing with the proposed one-stage method of ADSH. As for the united framework of DSH, the state-of-the-art spectral hashing methods of DGH and RDSH are included, and the deep unsupervised hashing of BA is also considered. Note that, these methods not only cover the spectral-based approaches, but also other learning-to-hash algorithms. We follow the common parameter setting of $m = 300$ and $s = 3$ [11], [20].We find that α is not sensitive to different datasets and keep it at 0.1 in all the experiments.

1) Spectral Rotation: First of all, it is expected to verify the effectiveness of *Spectral Rotation* (SR). Hence, we simply perform the rotation technique with the spectral solution of AGH [19]. Concretely, as AGH directly binarizes the real-valued solution via zero, it is fair to perform the same operation within SHSR. To this end, the second constraint in Eq. 5 is not considered any more and the sign function is directly performed over $\mathbf{FQ}$ instead of Eq. 8 for updating $\mathbf{B}$, which is named as AGH + SR. Table IV shows the comparison results in MAP, where multiple code lengths are considered. It is obvious that AGH + SR enjoys a noticeable improvement over AGH on both datasets. This is because SR transforms the embedded points into proper positions where the spectral solutions better approximate the discrete codes, as shown in Fig. 1. To make the codes more efficient [18], the balanced constraint is reconsidered, which actually leads to the SHSR objective, i.e., Eq. 5. In Table IV, SHSR shows better results than AGH but just comparable to AGH + SR after introducing the constraint. In some conditions, SHSR performs even worse than AGH + SR, such as SHSR @ 32 bits and 64 bits on CIFAR-10. As the only difference between AGH + SR and SHSR is the balanced constraint, the reason is probably that the revised spectral solution via SR has provided efficient manifold representation, so the balanced constraint may be not necessary for spectral hashing in such cases.

[6] http://yann.lecun.com/exdb/mnist/
[7] https://www.cs.toronto.edu/ kriz/cifar.html
[8] https://www.cs.tau.ac.il/ wolf/ytfaces/
[9] http://lms.comp.nus.edu.sg/research/NUS-WIDE.htm

TABLE II
HAMMING RANKING PERFORMANCE (IN MAP) ON MNIST AND CIFAR-10 DATASET WITH VARYING CODE LENGTHS

Dataset	MNIST						CIFAR-10					
Code Length	8 *bits*	16 *bits*	32 *bits*	64 *bits*	96 *bits*	128 *bits*	8 *bits*	16 *bits*	32 *bits*	64 *bits*	96 *bits*	128 *bits*
SH [18]	0.2929	0.2730	0.2607	0.2476	0.2498	0.2497	0.1438	0.1359	0.1324	0.1326	0.1298	0.1313
AGH-1 [19]	0.5205	0.5494	0.4431	0.3567	0.3229	0.3008	0.1966	0.1897	0.1927	0.1918	0.1882	0.1893
AGH-2 [19]	0.5601	0.6058	0.6447	0.6122	0.5846	0.5681	0.2103	0.1966	0.1897	0.1927	0.1900	0.1918
SHSR	**0.5862**	**0.6418**	**0.6891**	**0.6611**	**0.6456**	**0.6401**	**0.2121**	**0.2094**	**0.2010**	**0.2029**	**0.1962**	**0.1988**
ITQ [14]	0.3854	0.4025	0.4325	0.4451	0.4580	0.4598	0.1680	0.1709	0.1753	0.1813	0.1847	0.1864
SGH [30]	0.1075	0.1058	0.1067	0.1142	0.1250	0.1356	0.1411	0.1477	0.1457	0.1429	0.1434	0.1454
IMH [20]	0.5389	0.5309	0.4771	0.4390	0.4066	0.3853	0.1862	0.1743	0.1821	0.1859	0.1819	0.1869
ADSH	**0.5896**	**0.6085**	**0.6484**	**0.6751**	**0.6483**	**0.6597**	**0.2012**	**0.1934**	**0.1896**	**0.1957**	**0.1983**	**0.1987**
BA [16]	0.5871	0.6500	**0.7031**	**0.7354**	**0.7585**	**0.7597**	0.1558	0.1610	0.1712	0.1749	0.1830	0.1847
RDSH [25]	0.3917	0.4047	0.4407	0.4522	0.4583	0.4660	0.1693	0.1711	0.1754	0.1830	0.1873	0.1889
DGH [11]	0.5998	0.6100	0.5785	0.5414	0.5544	0.5555	0.2027	0.2000	0.2024	0.2001	0.1964	0.1918
DSH	**0.6011**	**0.6521**	0.6704	0.6294	0.6605	0.6185	**0.2122**	**0.2142**	**0.2116**	**0.2063**	**0.1971**	**0.2030**

TABLE III
HAMMING RANKING PERFORMANCE (IN MAP) ON YOUTUBE FACES AND NUS-WIDE DATASET WITH VARYING CODE LENGTHS

Dataset	YouTube Faces						NUS-WIDE					
Code Length	8 *bits*	16 *bits*	32 *bits*	64 *bits*	96 *bits*	128 *bits*	8 *bits*	16 *bits*	32 *bits*	64 *bits*	96 *bits*	128 *bits*
SH [18]	0.1355	**0.3131**	**0.5030**	0.5951	0.6136	0.6127	0.2505	0.2505	0.2509	0.2521	0.2570	0.2574
AGH-1 [19]	0.1273	0.2248	0.4498	0.5656	0.5902	0.5932	0.2503	0.2515	0.2574	0.2827	0.2802	0.2776
AGH-2 [19]	0.1304	0.1662	0.3046	0.5200	0.5654	0.5922	0.2520	0.2544	0.2559	0.2713	0.2765	0.2784
SHSR	**0.1601**	0.2963	0.4634	**0.6163**	**0.6501**	**0.6551**	**0.2584**	**0.2603**	**0.2788**	**0.2838**	**0.2861**	**0.2847**
ITQ [14]	**0.1972**	0.4402	0.6048	0.6647	0.6799	0.6911	0.2580	0.2727	0.2791	0.2704	0.2714	0.2721
SGH [30]	0.0877	0.2602	0.4101	0.5217	0.5661	0.5698	0.2590	0.2583	0.2589	0.2600	0.2613	0.2617
IMH [20]	0.0764	0.1259	0.1733	0.2553	0.2739	0.2934	0.2509	0.2511	0.2515	0.2525	0.2553	0.2553
ADSH	0.1224	**0.4427**	**0.6328**	**0.7366**	**0.7477**	**0.7604**	**0.2775**	**0.2819**	**0.2839**	**0.2848**	**0.2876**	**0.2857**
BA [16]	0.2313	**0.4918**	0.6166	0.6713	0.6833	0.6894	0.2797	0.2883	0.2930	**0.2952**	0.2949	0.2928
RDSH [25]	0.2005	0.4468	0.6142	0.6691	0.6842	0.6994	0.2587	0.2771	0.2823	0.2753	0.2735	0.2759
DGH [11]	0.2131	0.4591	0.6302	0.6517	0.6672	0.6805	0.2672	0.2731	0.2750	0.2754	0.2759	0.2747
DSH	**0.2334**	0.4465	**0.6848**	**0.7207**	**0.7528**	**0.7605**	**0.2834**	**0.2921**	**0.2954**	0.2938	**0.2962**	**0.3045**

TABLE IV
COMPARISON AMONG AGH, AGH WITH SR, AND SHSR ON CIFAR-10 AND NUS-WIDE DATASET

Dataset	Methods	12	16	24	32	64
CIFAR-10	AGH	18.63	18.97	19.28	19.27	19.18
	AGH+SR	20.99	20.92	20.53	20.84	20.80
	SHSR	20.99	20.94	20.54	20.10	20.29
NUS-WIDE	AGH	25.04	25.15	25.47	25.74	28.29
	AGH+SR	25.48	25.57	25.99	27.64	28.38
	SHSR	25.61	26.03	27.02	27.88	28.38

The front part of Table II and Table III show the detailed comparison results for SR in Hamming ranking. Notice that AGH-2 is a hierarchical hashing scheme based on the graph Laplacian eigenvectors of AGH-1. Obviously, SHSR has a remarkable improvement of 4 and 7 points on 8 and 16 bits, and $\sim$30 points on other bits over the baseline method of AGH-1, and 2$\sim$8 points over AGH-2 on MNIST. By comparing AGH with SH, it is easy to find that the neighborhood graph not only accelerates the graph building, but also improves the retrieval performance, as these graph-based methods focus on the nearest anchors. And the special cases on YouTube Faces are because of the out-of-sample strategy, which will be discussed in the following section.

The hash lookup results in F-meassure and Recall are provided in Fig. 2 and Fig. 3. We can see that SHSR outperforms the other methods with almost all the code lengths, especially on the F-meassure metric. When the code becomes longer, all the methods do not hold the high performance, including SHSR. Such phenomenon could come from two reasons. First, Shi and Malik [37] consider that the discrete solution is obtained by converting the real-valued spectral solution, so that the quantization error could accumulate with the increasing codes. Although AGH-2 proposes to give more priority to the lower eigenvectors, it also badly suffers from such problem. Second, the longer codes result in sparser hamming space. Hence, there will be less embedding points falling into the hamming ball when the number of samples is fixed. In other words, the hamming distance between nearby embedding points are enlarged relatively, which results in the declining hash lookup performance. Even so, SHSR still performs better than the other ones, especially on the YouTube Faces dataset.

2) Alternating Discrete Spectral Hashing: The hashing ranking performance in MAP on the four datasets are shown

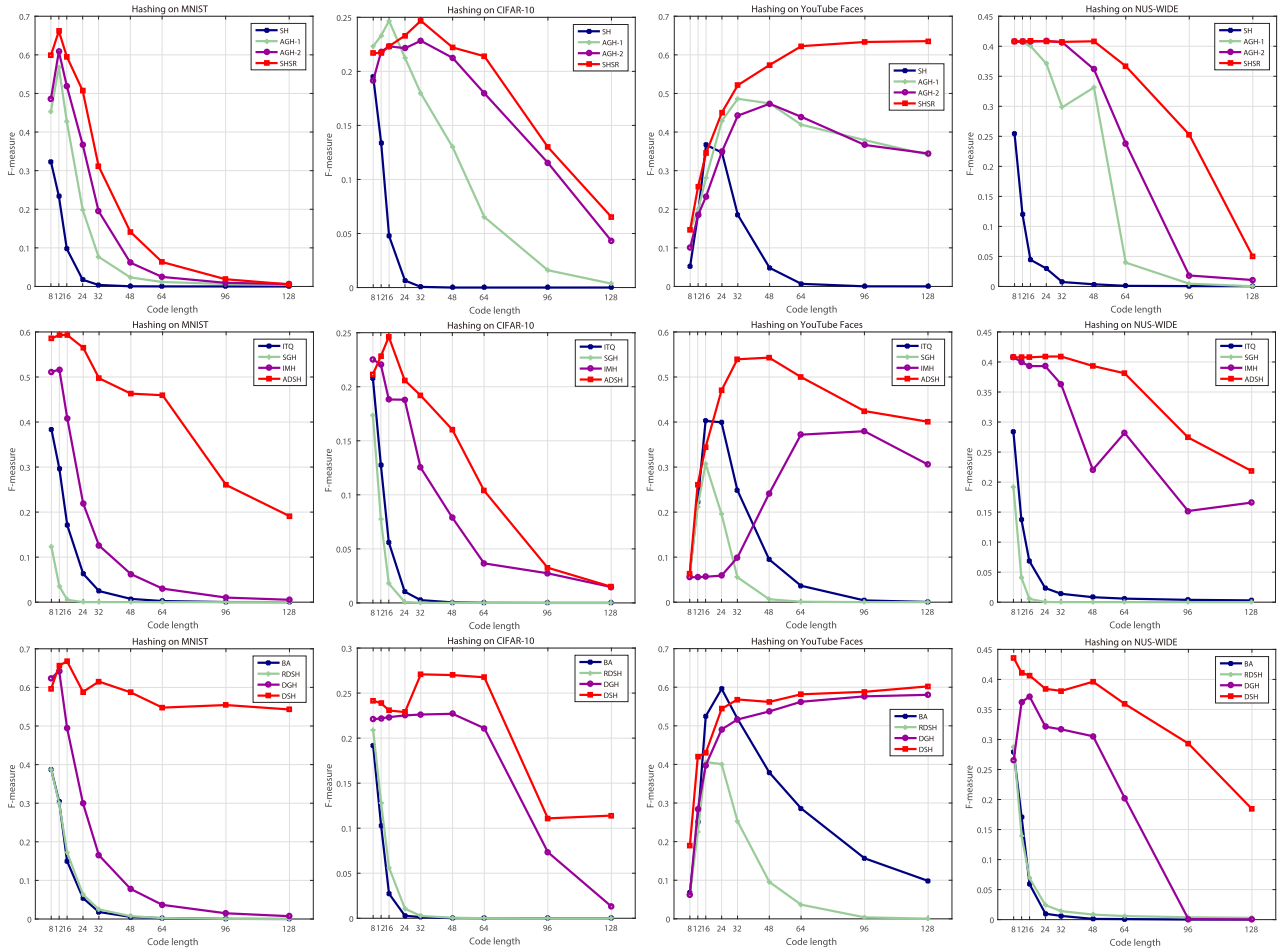

Fig. 2. Hash lookup performance (in F-measure) on MNIST, CIFAR-10, YouTube Faces and NUS-WIDE with varying code lengths.

in the middle part of Table II and Table III. It is clear to find that our method outperforms almost all the other three methods. And there are three points we should pay attention to. First, ITQ with the relaxation strategy almost always performs better than SGH and IMH which adopt the binarization approach. It confirms the necessity of directly solving the binary codes to some extent. Second, the spectral-based method of IMH decreases sharply with the increasing code length on MNIST, which results from the increasing error caused by the rounding operation and sparser hamming space. By contrast, ADSH shows an increasing MAP score. Third, although it is hard to theoretically guarantee the convergence of ADSH, the binary codes generated by ADSH better preserve the neighborhood structure compared with other methods. Such performance benefits from the effective code learning in the discrete space and confirms the ability of ADSH in practice.

Fig. 2 and Fig. 3 show the comparison results in F-measure and Recall, and ADSH shows noticeable performance compared with other methods on the four datasets. IMH achieves nice result on NUS-WIDE, which could benefit from the advantages of t-SNE in preserving local structures of multi-label data, but it is lower than ADSH when the codes become longer. Besides, ITQ and SGH have an significant decrease on the YouTube Faces dataset when faced with longer codes, while ADSH shows a rapid growth and then remains a high performance that is superior over the others, which also confirms the effectiveness of ADSH in practice.

3) Discrete Spectral Hashing: As the most similar works to DSH are DGH and RDSH, they are chosen for comparison, and the deep hashing of BA is also considered. The iteration numbers of GPI and the DSH, i.e., N_G and N, are set to 30. The Hamming ranking performance is shown in Table II and Table III. We can find that DSH shows the best performance in the most conditions. More precisely, DSH achieves noticeable improvements over the state-of-art spectral method of RDSH and DGH on all the datasets, but worse than BA on the MNIST dataset. This is because DSH directly executes the code constraints on the hashing codes, which are actually performed in the discrete space, while RDSH neglects these constraints and DGH chooses to perform the constraints over the real-valued solution instead of the discrete codes. Hence, DSH could learn more efficient codes. Different from these spectral-based methods, BA utilizes the ability of deep networks in the nonlinear modeling to encode the features, which aims to seek the semantic embedding of hashing codes [17]. Hence, BA can effectively encode the original features into efficient binary codes under the simple data distribution of MNIST. While for

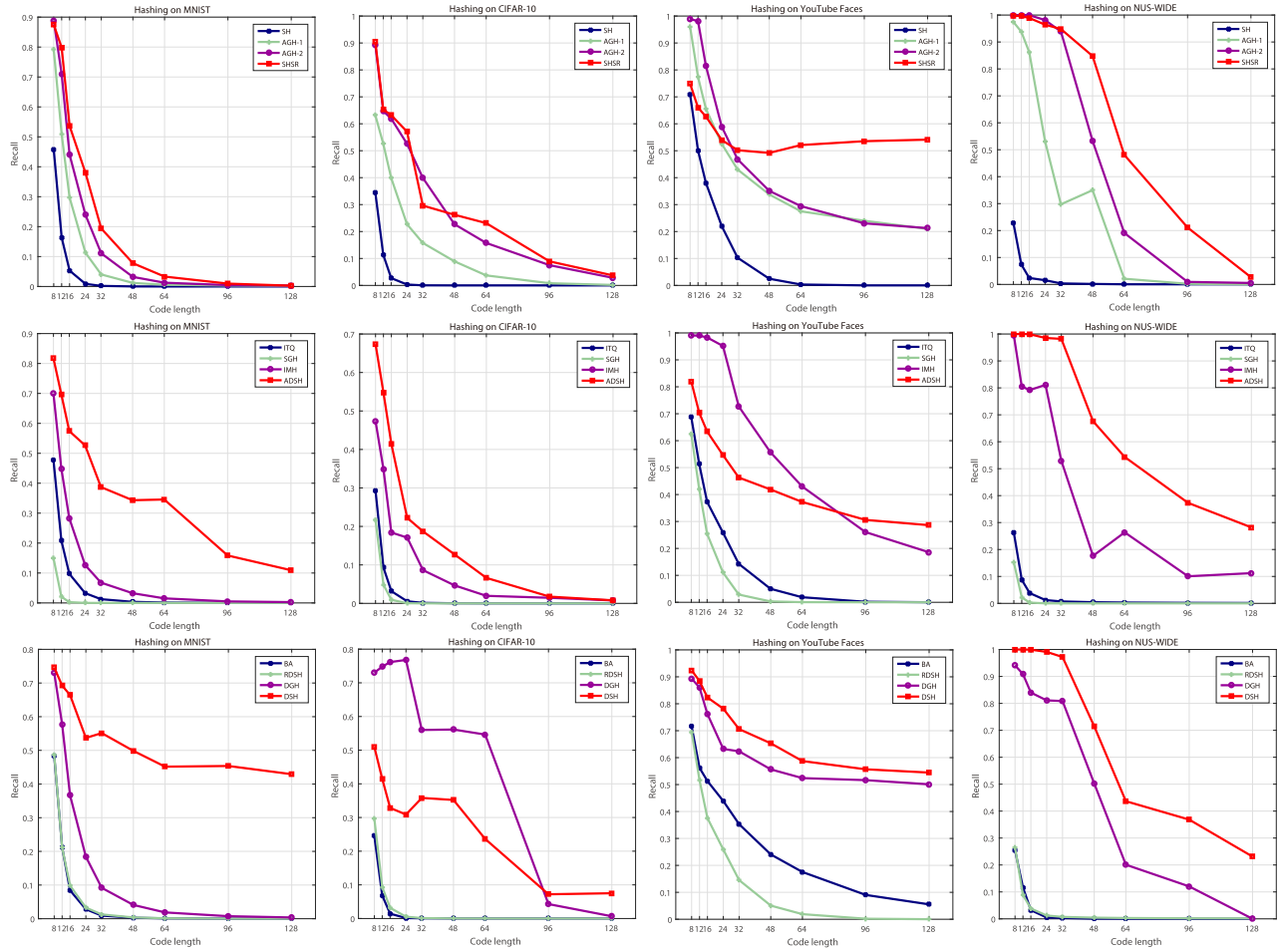

Fig. 3. Hash lookup performance (in Recall) on MNIST, CIFAR-10, YouTube Faces and NUS-WIDE with varying code lengths.

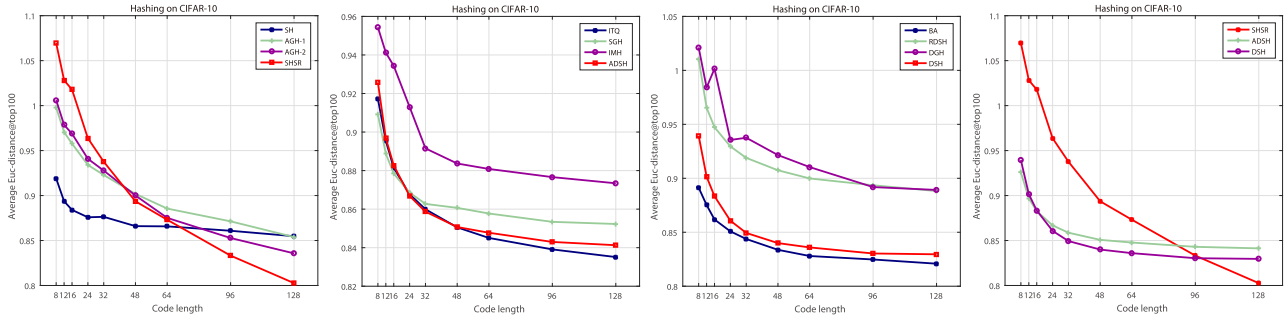

Fig. 4. The average Euclidean distance (in the original feature space) of the top-100 retrieved items with different code lengths.

the other three datasets, our proposed DSH shows stronger ability in encoding data under more complex distribution.

The hash lookup results in F-measure and Recall are plotted in the bottom of Fig. 2 and Fig. 3, respectively. It is obviously that DSH shows the best performance over the other three methods, especially BA. To be specific, there are two points we should pay attention to. First, different from the situation in Hamming ranking, DSH shows remarkable improvement over BA on the lookup metric, which indicates that DSH can provide more exact results within the top retrieved items. Second, different from the previous results, DSH avoids the situation of decreasing performance to some extent, such as the F-measure score on MNIST. This is because DSH contains both the advantages of SHSR and ADSH, that is to say, DSH can directly seek for the discrete spectral solution under efficient constraints in the hamming space.

D. Neighborhood Evaluation

To evaluate the quality of generated codes in preserving the neighborhood structure precisely, Fig. 4 shows the average Euclidean distance of the top-100 retrieved items in the original feature space (lower is better). It is obvious that these results are slightly different from F-measure and MAP.

TABLE V
THE PARAMETER SENSITIVITY ANALYSIS ABOUT α OF DSH

Dataset	0.001	0.01	0.1	1	10	100
MNIST	59.52	63.70	62.94	65.62	58.31	57.28
CIFAR-10	20.96	21.39	20.63	19.81	19.74	19.79

Specifically, AGH enjoys a good performance on the conventional metric while suffers from poor results in Fig. 4, similar for IMH and DGH. This is because the average distance is computed from only one hundred nearest items, instead of thousands of images with the same label. Under such evaluation metrics, our proposed methods still take the top two best performance, and only a litter worse than the deep method of BA. In the right-most of Fig.4, DSH almost always shows the best performance among the three proposed methods, which actually confirms its effectiveness in preserving the neighborhood structure when performing hashing projection. Note that, similar with other methods (i.e., SH, IMH, and AGH), our method assumes that there is a low-dimensional manifold structure among the training data. If the data structure does not meet the manifold assumption, our method may be inferior to other non-spectral methods, such as BA.

E. Parameter Sensitivity

As mentioned in Sec.V, DSH is an integrated framework of the spectral rotation in SHSR and discrete code learning in ADSH, hence it involves an hyper-parameter of α who controls the importance of the spectral rotation and quantization term. In fact, SHSR and ADSH are the special cases of DSH when α approaches zero or positive infinity, respectively. As the performance of these two proposed methods are comparable to DSH as shown above, the value of α has limited effects on the final hashing codes. Even so, we still perform a parameter analysis about it on the MNIST and CIFAR-10 dataset. Table. V shows the Hamming ranking performance @64 bits codes. Obviously, DSH is not very sensitive to the value of α on both datasets. Specifically, When α becomes smaller (e.g., 1 to 0.001 on MNIST), DSH tends to solve the relaxed spectral objective but pay few attention on the quantization with rigorous constraints, hence the generated codes become low efficient. When α becomes larger (e.g., 1 to 100 on MNIST), DSH emphasizes more on the discrete codes learning and the spectral solution could be quickly transformed into binary value. However, such practice could result in poor spectral solution, which finally leads to the declining performance. Even so, DSH still enjoys the bilateral merits from SHSR and ADSH and has superiorities over other methods.

F. Efficient Out-of-Sample Methods

By comparing the out-of-sample methods of SHSR, ADSH, and DSH, we can find that SHSR takes the distinctive real-valued solution of the training set instead of the binary ones. This is because the direct binarization could result in roughly approximated hashing codes within the conventional two-stage hashing method, i.e., AGH, IMH. Hence, they usually choose the real-valued spectral solution, and it is fair to employ the same strategy for SHSR. However, to further evaluate the performance of SHSR when employing the discrete codes instead, we perform the following comparison. The experiments are conducted on CIFAR-10 and YouTube Faces dataset. The results in MAP are shown in Table. VI, where SHSR stands for the original real-valued method and SHSR-B represents the binary strategy. It is clear that SHSR-B is comparable with SHSR on CIFAR-10, but much better on YouTube Faces. Such phenomenon indicates that the minimized quantization term with spectral rotation (i.e., Eq. 8) indeed helps to learn more efficient discrete codes, especially when faced with more categories and more data. Hence, it is recommended to use the discrete solution of Eq. 8 for the hashing codes generation of testing data in SHSR.

TABLE VI
THE COMPARISON RESULTS ABOUT THE OUT-OF-SAMPLE STRATEGIES OF THE SHSR METHOD

Dataset	Methods	8	24	32	48	64
CIFAR-10	SHSR	21.22	20.54	20.10	20.24	20.29
	SHSR-B	21.23	20.51	19.89	20.33	20.19
YouTube-Faces	SHSR	16.01	38.96	46.34	56.14	61.63
	SHSR-B	16.42	43.89	52.36	59.80	63.20

VII. CONCLUSION

In this paper, we mainly focus on two conventional problems of spectral hashing, one is the poor spectral candidate, and the other is the intractable binary constraint. To facilitate the spectral hashing to overcome the above problems, we provide two specific solutions for them. First, we propose to seek better spectral solution by introducing the spectral rotation technique into the quantization error, which constitutes a two-stage spectral-based hashing method, called SHSR. By comparing with the original spectral solution, SHSR shows noticeable improvement on different evaluation metrics. Second, we propose to directly optimize the original objective function under the binary constraint. To do so, two constraint sets are constructed and a sequence of projections are alternately performed on them within the reweighted framework, which is named as ADSH. The experimental results show that ADSH can learn more efficient codes than the two-stage methods. Third, to integrate the merits of above methods, we finally propose DSH that considers both of spectral rotation and discrete codes learning and can be efficiently optimized by existing methods. Extensive experiments on four large-scale datasets demonstrate that DSH can generate efficient compact codes within comparable time complexity.

APPENDIX

Theorem 2: $\boldsymbol{Q} = \boldsymbol{V}\boldsymbol{U}^T$ *is the optimal solution to the given objective function in Eq. 9, where* $\boldsymbol{U}$ *and* $\boldsymbol{V}$ *are the left and right singular vectors of the compact Singular Value Decomposition (SVD) of* $\boldsymbol{G}$.

Proof: Suppose that the SVD of $\mathbf{G}$ is $\mathbf{G} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$ for arbitrary $\mathbf{B}$, then Eq. 9 can be re-written as

$$Tr\left(\mathbf{G}\mathbf{Q}\right) = Tr\left(\mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T\mathbf{Q}\right) \tag{25}$$

$$= Tr\left(\mathbf{V}^T\mathbf{Q}\mathbf{U}\boldsymbol{\Sigma}\right) \tag{26}$$

$$= Tr\left(\boldsymbol{\Gamma}\boldsymbol{\Sigma}\right), \tag{27}$$

where $\mathbf{\Gamma} = \mathbf{V}^T\mathbf{Q}\mathbf{U}$. Suppose that $\{\tau_i\}_{i=1}^n$ and $\{\sigma_i\}_{i=1}^n$ are the singular values of $\mathbf{\Gamma}$ and $\mathbf{\Sigma}$, respectively. Meanwhile, due to $\mathbf{\Gamma}^T\mathbf{\Gamma} = \mathbf{I}$, the singular value $\tau_i = 1$. Then, by using the von Neumann's trace inequality [38], Eq. 25 becomes

$$Tr\left(\mathbf{\Gamma}\mathbf{\Sigma}\right) \leq \sum_{i=1}^{r} \sigma_i. \tag{28}$$

The equality holds when $\mathbf{\Gamma} = \mathbf{I}$, which derives

$$\mathbf{Q} = \mathbf{V}\mathbf{U}^T. \tag{29}$$

This completes our proof. ■

REFERENCES

[1] R. Yang, Y. Shi, and X.-S. Xu, "Discrete multi-view hashing for effective image retrieval," in *Proc. ACM Int. Conf. Multimedia Retr.*, 2017, pp. 175–183.

[2] K. Weinberger, A. Dasgupta, J. Langford, A. Smola, and J. Attenberg, "Feature hashing for large scale multitask learning," in *Proc. 26th Annu. Int. Conf. Mach. Learn.*, 2009, pp. 1113–1120.

[3] Y. Luo, Y. Yang, F. Shen, Z. Huang, P. Zhou, and H. T. Shen, "Robust discrete code modeling for supervised hashing," *Pattern Recognit.*, vol. 75, pp. 128–135, Mar. 2018.

[4] Y. Gong, S. Kumar, H. A. Rowley, and S. Lazebnik, "Learning binary codes for high-dimensional data using bilinear projections," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 484–491.

[5] S. Korman and S. Avidan, "Coherency sensitive hashing," in *Proc. Int. Conf. Comput. Vis.*, Nov. 2011, pp. 1607–1614.

[6] J. Wang, W. Liu, S. Kumar, and S.-F. Chang, "Learning to hash for indexing big data—A survey," *Proc. IEEE*, vol. 104, no. 1, pp. 34–57, Jan. 2016.

[7] A. Andoni and P. Indyk, "Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions," in *Proc. 47th Annu. IEEE Symp. Found. Comput. Sci. (FOCS)*, Oct. 2006, pp. 459–468.

[8] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni, "Locality-sensitive hashing scheme based on p-stable distributions," in *Proc. 29th Annu. Symp. Comput. Geometry*, 2004, pp. 253–262.

[9] M. S. Charikar, "Similarity estimation techniques from rounding algorithms," in *Proc. 34th Annu. ACM Symp. Theory Comput.*, 2002, pp. 380–388.

[10] B. Kulis and K. Grauman, "Kernelized locality-sensitive hashing," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 6, pp. 1092–1104, Jun. 2012.

[11] W. Liu, C. Mu, S. Kumar, and S.-F. Chang, "Discrete graph hashing," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 3419–3427.

[12] J. Wang, T. Zhang, J. Song, N. Sebe, and H. T. Shen, "A survey on learning to hash," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 4, pp. 769–790, Apr. 2018.

[13] A. Andoni and I. Razenshteyn, "Optimal data-dependent hashing for approximate near neighbors," in *Proc. 47th Annu. ACM Symp. Theory Comput.*, 2015, pp. 793–801.

[14] Y. Gong, S. Lazebnik, A. Gordo, and F. Perronnin, "Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 12, pp. 2916–2929, Dec. 2013.

[15] W. Kong and W.-J. Li, "Isotropic hashing," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1646–1654.

[16] M. A. Carreira-Perpinan and R. Raziperchikolaei, "Hashing with binary autoencoders," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 557–566.

[17] R. Salakhutdinov and G. Hinton, "Semantic hashing," *Int. J. Approx. Reasoning*, vol. 50, no. 7, pp. 969–978, Jul. 2009.

[18] Y. Weiss, A. Torralba, and R. Fergus, "Spectral hashing," in *Proc. Adv. Neural Inf. Process. Syst.*, 2009, pp. 1753–1760.

[19] W. Liu, J. Wang, S. Kumar, and S.-F. Chang, "Hashing with graphs," in *Proc. 28th Int. Conf. Mach. Learn. (ICML)*, 2011, pp. 1–8.

[20] F. Shen, C. Shen, Q. Shi, A. van den Hengel, and Z. Tang, "Inductive hashing on manifolds," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 1562–1569.

[21] M. Hu, Y. Yang, F. Shen, N. Xie, and H. T. Shen, "Hashing with angular reconstructive embeddings," *IEEE Trans. Image Process.*, vol. 27, no. 2, pp. 545–555, Feb. 2018.

[22] X. Li, D. Hu, and F. Nie, "Large graph hashing with spectral rotation," in *Proc. AAAI Conf. Artif. Intell.*, 2017, pp. 2203–2209.

[23] F. R. K. Chung, *Spectral Graph Theory*, vol. 92. Providence, RI, USA: AMS, 1997.

[24] W. Liu, J. He, and S.-F. Chang, "Large graph construction for scalable semi-supervised learning," in *Proc. 27th Int. Conf. Mach. Learn. (ICML)*, 2010, pp. 679–686.

[25] Y. Yang, F. Shen, H. T. Shen, H. Li, and X. Li, "Robust discrete spectral hashing for large-scale image semantic indexing," *IEEE Trans. Big Data*, vol. 1, no. 4, pp. 162–171, Dec. 2015.

[26] F. Nie, X. Wang, and H. Huang, "Multiclass capped ℓ_p-Norm SVM for robust classifications," in *Proc. 31st AAAI Conf. Artif. Intell. (AAAI)*, San Francisco, CA, USA, 2017, pp. 2415–2421.

[27] R. Escalante and M. Raydan, *Alternating Projection Methods*. Philadelphia, PA, USA: SIAM, 2011.

[28] T. E. Booth, "Power iteration method for the several largest eigenvalues and eigenfunctions," *Nucl. Sci. Eng.*, vol. 154, no. 1, pp. 48–62, 2006.

[29] F. Nie, R. Zhang, and X. Li. (2017). "A generalized power iteration method for solving quadratic problem on the Stiefel manifold." [Online]. Available: https://arxiv.org/abs/1701.00381

[30] Q.-Y. Jiang and W.-J. Li, "Scalable graph hashing with feature transformation," in *Proc. IJCAI*, 2015, pp. 2248–2254.

[31] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[32] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Univ. Toronto, Toronto, Canada, Tech. Rep., 2009, vol. 1, no. 4, p. 7.

[33] L. Wolf, T. Hassner, and I. Maoz, "Face recognition in unconstrained videos with matched background similarity," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2011, pp. 529–534.

[34] T.-S. Chua, J. Tang, R. Hong, H. Li, Z. Luo, and Y. Zheng, "NUS-WIDE: A real-world Web image database from National University of Singapore," in *Proc. ACM Conf. Image Video Retr. (CIVR)*, Santorini, Greece, 2009, Art. no. 48.

[35] A. Oliva and A. Torralba, "Modeling the shape of the scene: A holistic representation of the spatial envelope," *Int. J. Comput. Vis.*, vol. 42, no. 3, pp. 145–175, 2001.

[36] T. Ahonen, A. Hadid, and M. Pietikäinen, "Face description with local binary patterns: Application to face recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 12, pp. 2037–2041, Dec. 2006.

[37] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 888–905, Aug. 2000.

[38] L. Mirsky, "A trace inequality of John von Neumann," *Monatshefte furMath.*, vol. 79, no. 4, pp. 303–306, 1975.

Di Hu is currently pursuing the Ph.D. degree with Northwestern Polytechnical University, under the supervision of F. Nie and X. Li. He mainly focuses on multimodal machine learning and relevant applications, such as audiovisual understanding, cross-modal retrieval, and hashing. He has published several papers in top conferences, such as the CVPR, ICCV, AAAI, and ACM MM. He has served as a PC Member for the AAAI, CVPR, and ACCV.

Feiping Nie received the Ph.D. degree in computer science from Tsinghua University, China, in 2009. He is currently a Full Professor with Northwestern Polytechnical University, China. His research interests are machine learning and its applications, such as pattern recognition, data mining, computer vision, image processing, and information retrieval. He has published over 100 papers in journals and conferences such as the TPAMI, IJCV, TIP, TNNLS, TKDE, ICML, NIPS, KDD, IJCAI, AAAI, ICCV, CVPR, and ACM MM. His papers have been cited over 10 000 times and the H-index is 53. He is currently serving as an associate editor or PC member for several prestigious journals and conferences in the related fields.

Xuelong Li (M'02–SM'07–F'12) is currently a Full Professor with the Xi'an Institute of Optics and Precision Mechanics, Chinese Academy of Sciences, Xi'an, China.