# Hashing on Nonlinear Manifolds

Fumin Shen, Chunhua Shen, Qinfeng Shi, Anton van den Hengel, Zhenmin Tang, and Heng Tao Shen

***Abstract*— Learning-based hashing methods have attracted considerable attention due to their ability to greatly increase the scale at which existing algorithms may operate. Most of these methods are designed to generate binary codes preserving the Euclidean similarity in the original space. Manifold learning techniques, in contrast, are better able to model the intrinsic structure embedded in the original high-dimensional data. The complexities of these models, and the problems with out-of-sample data, have previously rendered them unsuitable for application to large-scale embedding, however. In this paper, how to learn compact binary embeddings on their intrinsic manifolds is considered. In order to address the above-mentioned difficulties, an efficient, inductive solution to the out-of-sample data problem, and a process by which nonparametric manifold learning may be used as the basis of a hashing method are proposed. The proposed approach thus allows the development of a range of new hashing techniques exploiting the flexibility of the wide variety of manifold learning approaches available. It is particularly shown that hashing on the basis of t-distributed stochastic neighbor embedding outperforms state-of-the-art hashing methods on large-scale benchmark data sets, and is very effective for image classification with very short code lengths. It is shown that the proposed framework can be further improved, for example, by minimizing the quantization error with learned orthogonal rotations without much computation overhead. In addition, a supervised inductive manifold hashing framework is developed by incorporating the label information, which is shown to greatly advance the semantic retrieval performance.**

***Index Terms*— Hashing, binary code learning, manifold learning, image retrieval.**

## I. Introduction

ONE key challenge in many large scale image data based applications is how to index and organize the data accurately, but also efficiently. Various hashing techniques have attracted considerable attention in computer vision, information retrieval and machine learning [11], [13], [28], [29], [43], [49], [53], [56], and seem to offer great promise towards this goal. This paper is focused on the hashing methods that aim to encode documents or images as a set of short binary codes, while maintaining aspects of the structure of the original data (*e.g.*, similarities between data points). The advantage of these compact binary representations is that pairwise comparisons may be carried out extremely efficiently in the Hamming space. This means that many algorithms which are based on such pairwise comparisons can be made more efficient, and applied to much larger datasets. Due to the flexibility of hash codes, hashing techniques can be applied in many ways. One can, for example, efficiently perform a similarity search by exploring only those data points falling into the close-by buckets to the query by the Hamming distance, or use the binary representations for other tasks like image classification.

Locality sensitive hashing (LSH) [11] is one of the most well-known *data-independent* hashing methods, and generates hash codes based on random projections. With the success of LSH, random hash functions have been extended to several similarity measures, including $p$-norm distances [7], the Mahalanobis metric [26], and kernel similarity [25], [40]. However, the methods belonging to the LSH family normally require relatively long hash codes (compared to the recently developed data-dependent hashing algorithms) and several hash tables to achieve high precision and recall. This leads to a larger storage cost than would otherwise be necessary, and thus limits the sale at which the algorithm may be applied.

*Data-dependent* or learning-based hashing methods have been developed with the goal of learning more compact hash codes. Directly learning binary embeddings typically results in an optimization problem which is very hard to solve, however. Relaxation is often used to simplify the optimization (see [3], [49]). As in LSH, the methods aim to identify a set of hyperplanes, but now these hyperplanes are learned, rather than randomly selected. For example, PCAH [49], semi-supervised hashing (SSH) [49], iterative quantization (ITQ) [13] and isotropic hashing [23], generate linear hash functions through simple principal component analysis (PCA) projections, while LDAhash [3] is based on Linear Discriminant Analysis (LDA). Extending this idea, there are also methods which learn hash functions in a kernel space, such as binary reconstructive embeddings (BRE) [24], random maximum margin hashing (RMMH) [22] and kernel-based supervised hashing (KSH) [33].

Other representative methods in the literature include the unsupervised locally linear hashing [20], discrete graph hashing (DGH) [32] and the supervised

Manuscript received August 14, 2014; revised December 6, 2014 and February 9, 2015; accepted February 10, 2015. Date of publication February 24, 2015; date of current version March 27, 2015. This work was supported in part by the Australian Research Council Future Fellowship under Grant FT120100969 and in part by the National Natural Science Foundation of China under Project 61472063 and Project 61473154. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Dacheng Tao.

F. Shen is with the School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 610051, China (e-mail: fshen@uestc.edu.cn).

C. Shen and A. van den Hengel are with the School of Computer Science, The University of Adelaide, Adelaide, SA 5005, Australia, and also with Australian Centre for Robotic Vision, Brisbane, QLD 4000, Australia (e-mail: chhshen@gmail.com; anton.vandenhengel@adelaide.edu.au).

Q. Shi is with the School of Computer Science, The University of Adelaide, Adelaide, SA 5005, Australia (e-mail: javen.shi@adelaide.edu.au).

Z. Tang is with the School of Computer Science and Technology, Nanjing University of Science and Technology, Nanjing 210094, China (e-mail: tzm.cs@mail.njust.edu.cn).

H. T. Shen is with the School of Information Technology and Electrical Engineering, The University of Queensland, Brisbane, QLD 4072, Australia (e-mail: shenht@itee.uq.edu.au).

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.2015.2405340

1057-7149 © 2015 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

**(a)** Queries **(b)** $\ell_2$ dist. on 784D **(c)** LSH with 128-bits **(d)** $\ell_2$ dist. on embeded 48D **(e)** Hamm. dist. with 48-bits

Fig. 1. Top 10 retrieved digits for 4 queries (a) on a subset of MNIST with 300 samples. Search is conducted in the original feature space (b, c) and nonlinear embedding space by t-SNE [47] (d, e) using Euclidean distance (b, d) and Hamming distance (c, e).

minimal loss hashing (MLH) [38], ranking-based supervised hashing [50], two-step hashing (TSH) [30], FastHash [29], graph cuts coding (GCC) [10], *etc*. A bilinear form of hash function is adopted in [12] and [35].

In a departure from such methods, however, spectral hashing (SH) [53], one of the most popular learning-based methods, generates hash codes by solving the relaxed mathematical program that is similar to the one in Laplacian eigenmaps [1]. Embedding the original data into a low-dimensional space while simultaneously preserving the inherent neighborhood structure is critical for learning compact, effective hash codes. In general, nonlinear manifold learning methods are more powerful than linear dimensionality reduction techniques, as they are able to more effectively preserve the *local* structure of the input data without assuming *global* linearity [44]. The geodesic distance on a manifold has been shown to outperform the Euclidean distance in the high-dimensional space for image retrieval [16], for example. Figure 1 demonstrates that searching using either the Euclidean or Hamming distance after nonlinear embedding results in more semantically accurate neighbors than the same search in the original feature space, and thus that low-dimensional embedding may actually improve retrieval or classification performance. However, the only widely used nonlinear embedding method for hashing is Laplacian eigenmaps (LE) ([34], [53], [55]). Other effective manifold learning approaches (*e.g.*, Locally Linear Embedding (LLE) [41], Elastic Embedding (EE) [4] or t-Distributed Stochastic Neighbor Embedding (t-SNE) [47]) have rarely been explored for hashing. Very recently, the authors of [20] choose to jointly minimize the LLE embedding error and the quantization loss with an orthogonal rotation.

One problem hindering the use of manifold learning for hashing is that these methods do not directly scale to large datasets. For example, to construct the neighborhood graph (or pairwise similarity matrix) in these algorithms for $n$ data points is $O(n^2)$ in time, which is intractable for large datasets. The second problem is that they are typically non-parametric and thus cannot efficiently solve the critical out-of-sample extension problem. This fundamentally limits their application to hashing, as generating codes for new samples is an essential part of the problem. One of the widely used solutions for the methods involving spectral decomposition (*e.g.*, LLE, LE and isometric feature mapping (ISOMap) [45]) is the Nyström extension [2], [44], which solves the problem by learning eigenfunctions of a kernel matrix. As mentioned in [53], however, this is impractical for large-scale hashing since the Nyström extension is as expensive as doing exhaustive nearest neighbor search ($O(n)$). A more significant problem, however, is that the Nyström extension cannot be directly applied to *non-spectral* manifold learning methods such as t-SNE.

In order to address the out-of-sample extension problem, this study proposes a new non-parametric regression approach which is both efficient and effective. This method allows rapid assignment of new codes to previously unseen data in a manner which preserves the underlying structure of the manifold. Having solved the out-of-sample extension problem, a method by which a learned manifold may serve as the basis for a binary encoding is introduced. This method is designed so as to generate encodings which reflect the geodesic distances along such manifolds. On this basis, a range of new embedding approaches based on a variety of manifold learning methods are developed. The best performing of these is based on manifolds identified through t-SNE, which has been shown to be effective in discovering semantic manifolds amongst the set of all images [47].

Given the computational complexity of many manifold learning methods, it is shown in this work that it is possible to learn the manifold on the basis of a small subset of the data $\mathbf{B}$ (with size $m \ll n$), and subsequently to inductively insert the remainder of the data, and any out-of-sample data, into the embedding in $O(m)$ time per point. This process leads to an embedding method labelled as Inductive Manifold-Hashing (IMH) which is shown to outperform state-of-the-art methods on several large scale datasets both quantitatively and qualitatively.

As an extension, this study shows that the proposed IMH framework can be improved by minimizing the quantization error of mapping real-valued data to binary codes, for example, through orthogonal rotations. Significant performance gains are achieved by this simple step, as shown in Section V. Based on supervised subspace learning, this study also presents a supervised inductive manifold hashing framework (IMHs), which is shown to significantly advance the semantic retrieval performance of IMH.

The rest of this paper is organized as follows. In Section II, some representative hashing methods related are briefly reviewed. Section III describes the proposed Inductive Manifold-Hashing framework, followed by the experimental results in Section IV. The IMH method is further shown to be improved by learned orthogonal rotations in Section V. Section VI introduces the supervised extension of the inductive manifold hashing framework based on supervised subspace learning.

This paper is an extended version of the work previously published in [42]. Major improvements over [42] include the minimization of quantization errors with learned rotations (Section V) and the extension to the

supervised case (Section VI). We made the code available at https://github.com/chhshen/Hashing-on-Nonlinear-Manifolds.

## II. Related Work

Learning based or data-dependent hashing has achieved considerable attention recently in computer vision, machine learning and information retrieval community. Many hashing methods have been proposed by applying different learning algorithms, including unsupervised methods [13], [34], [53], [55], and (semi-)supervised methods [24], [29], [33], [49]. In this section, some representative unsupervised hashing methods related to the proposed method are briefly reviewed.

### A. Spectral Hashing

Weiss et al. [53] formulated the spectral hashing (SH) problem as

$$\min_{\mathbf{Y}} \sum_{\mathbf{x}_i,\mathbf{x}_j\in\mathbf{X}} \mathrm{w}(\mathbf{x}_i,\mathbf{x}_j)\|\mathbf{y}_i-\mathbf{y}_j\|^2$$
$$\text{s.t.}\quad \mathbf{Y}\in\{-1,1\}^{n\times r},\quad \mathbf{Y}^\top\mathbf{Y}=n\mathbf{I},\quad \mathbf{Y}^\top\mathbf{1}=0. \tag{1}$$

Here $\mathbf{y}_i \in \{-1,1\}^r$, the $i$th row in $\mathbf{Y}$, is the hash code that one wants to learn for $\mathbf{x}_i \in \mathbf{R}^d$, which is one of the $n$ data points in the training data set $\mathbf{X}$. $\mathbf{W} \in \mathbf{R}^{n\times n}$ with $\mathbf{W}_{ij} = \mathrm{w}(\mathbf{x}_i,\mathbf{x}_j) = \exp(-\|\mathbf{x}_i-\mathbf{x}_j\|^2/\sigma^2)$ is the graph affinity matrix, where $\sigma$ is the bandwidth parameter. $\mathbf{I}$ is the identity matrix. The last two constraints force the learned hash bits to be uncorrelated and balanced, respectively. By removing the first constraint (*i.e.*, *spectral relaxation* [53]), $\mathbf{Y}$ can be easily obtained by spectral decomposition on the Laplcaian matrix $\mathbf{L} = \mathbf{D} - \mathbf{W}$, where $\mathbf{D} = \mathrm{diag}(\mathbf{W1})$ and $\mathbf{1}$ is the vector of all ones. However, constructing $\mathbf{W}$ is $O(dn^2)$ (in time) and calculating the Nyström extension for a new point is $O(rn)$, which are both intractable for large datasets. It is assumed in SH [53], therefore, that the data are sampled from a uniform distribution, which leads to a simple analytical eigenfunction solution of 1-D Laplacians. However, this strong assumption is often not true in practice and the manifold structure of the original data is thus destroyed [34].

SH was extended into the tangent space in [6], however, based on the same uniform assumption. The author of [6] also proposed a non-Euclidean SH algorithm based on nonlinear clustering, which is $O(n^3)$ for training and $O(m+n/m)$ for testing. Weiss et al. [52] then improved SH by expanding the codes to include the outer-product eigenfunctions instead of only single-dimension eigenfunctions in SH, *i.e.*, multidimensional spectral hashing (MDSH).

### B. Graph Based Hashing

To efficiently solve problem (1), anchor graph hashing (AGH) [34] approximates the affinity matrix $\mathbf{W}$ by the low-rank matrix $\hat{\mathbf{W}} = \mathbf{Z}\Lambda^{-1}\mathbf{Z}$, where $\mathbf{Z} \in \mathbf{R}^{n\times m}$ is the normalized affinity matrix (with $k$ non-zeros in each row) between the training samples and $m$ *anchors* (generated by $K$-means), and $\Lambda^{-1}$ normalizes $\hat{\mathbf{W}}$ to be doubly stochastic. Then the desired hash functions may be efficiently identified by binarizing the Nyström eigenfunctions [2] with the approximated affinity matrix $\hat{\mathbf{W}}$. AGH is thus efficient, in that it has linear training time and constant search time, but as is the case for SH [53], the generalized eigenfunction is derived only for the Laplacian eigenmaps embedding.

Different from SH and AGH, Locally Linear Hashing (LLH [20]) constructs the graph affinity by locality-sensitive sparse coding to better capture the local linearity of manifolds. With the obtained affinity matrix, LLH formulates hashing as a joint optimization problem of LLE embedding error and quantization loss.

### C. Self-Taught Hashing

Self-taught Hashing (STH) [55] addressed the out-of-sample problem by a novel way: hash functions are obtained by training a support vector machine (SVM) classifier for each bit using the pre-learned binary codes as class labels. The binary codes were learned by directly solving (1) with a cosine similarity function. This process has prohibitive computational and memory costs, however, and training the SVM can be time consuming for dense data. Very recently, this idea was implemented in collaboration with graph cuts to the binary coding problems to bypass continuous relaxation [10], [29].

## III. The Proposed Method

### A. Inductive Learning for Hashing

Assuming that one has the manifold-based embedding $\mathbf{Y} := \{\mathbf{y}_1,\mathbf{y}_2,\cdots,\mathbf{y}_n\}$ for the entire training data $\mathbf{X} := \{\mathbf{x}_1, \mathbf{x}_2,\cdots,\mathbf{x}_n\}$. Given a new data point $\mathbf{x}_q$, one aims to generate an embedding $\mathbf{y}_q$ which preserves the local neighborhood relationships among its neighbors $\mathcal{N}_k(\mathbf{x}_q)$ in $\mathbf{X}$. The following simple objective is utilized:

$$\mathcal{C}(\mathbf{y}_q) = \sum_{i=1}^{n} \mathrm{w}(\mathbf{x}_q,\mathbf{x}_i)\|\mathbf{y}_q-\mathbf{y}_i\|^2, \tag{2}$$

where

$$\mathrm{w}(\mathbf{x}_q,\mathbf{x}_i) = \begin{cases}\exp(-\|\mathbf{x}_q-\mathbf{x}_i\|^2/\sigma^2), & \text{if } \mathbf{x}_i\in\mathcal{N}_k(\mathbf{x}_q),\\ 0 & \text{otherwise.}\end{cases}$$

Minimizing (2) naturally uncovers an embedding for the new point on the basis of its nearest neighbors on the low-dimensional manifold initially learned on the base set. That is, in the low-dimensional space, the new embedded location for the point should be close to those of the points close to it in the original space.

Differentiate $\mathcal{C}(\mathbf{y}_q)$ with respect to $\mathbf{y}_q$, one obtains

$$\left.\frac{\partial\mathcal{C}(\mathbf{y}_q)}{\mathbf{y}_q}\right|_{\mathbf{y}_q=\mathbf{y}_q^\star} = 2\sum_{i=1}^{n}\mathrm{w}(\mathbf{x}_q,\mathbf{x}_i)(\mathbf{y}_q^\star-\mathbf{y}_i) = 0, \tag{3}$$

which leads to the optimal solution

$$\mathbf{y}_q^\star = \frac{\sum_{i=1}^{n}\mathrm{w}(\mathbf{x}_q,\mathbf{x}_i)\mathbf{y}_i}{\sum_{i=1}^{n}\mathrm{w}(\mathbf{x}_q,\mathbf{x}_i)}. \tag{4}$$

Equation (4) provides a simple inductive formulation for the embedding: produce the embedding for a new data point by a (sparse) locally linear combination of the base embeddings.

The proposed approach here is inspired by Delalleau et al. [8], where they have focused on non-parametric graph-based learning in semi-supervised classification. The aim of this study here is completely different: the present work attempts to scale up the manifold learning process for hashing in an unsupervised manner.

The resulting solution (4) is consistent with the basic smoothness assumption in manifold learning, that close-by data points lie on or close to a locally linear manifold [1], [41], [45]. This local-linearity assumption has also been widely used in semi-supervised learning [8], [54], image coding [51], and similar. This paper proposes to apply this assumption to hash function learning.

However, as aforementioned, (4) does not scale well for both computing $\mathbf{Y}$ ($O(n^2)$ *e.g.*, for LE) and out-of-sample extension ($O(n)$), which is intractable for large scale tasks. Next, it is shown that the following prototype algorithm is able to approximate $\mathbf{y}_q$ using only a small base set well.

### B. The Prototype Algorithm

This prototype algorithm is based on entropy numbers defined below.

*Definition 1 (Entropy Numbers [18]): Given any $Y \subseteq \mathbb{R}^r$ and $m \in \mathbb{N}$, the $m$-th entropy number $\epsilon_m(Y)$ of $Y$ is defined as*

$$\epsilon_m(Y) := \inf\{\epsilon > 0 | \mathcal{N}(\epsilon, Y, \|\cdot - \cdot\|) \leq m\},$$

*where $\mathcal{N}$ is the covering number. This means $\epsilon_m(Y)$ is the smallest radius that $Y$ can be covered by less or equal to $m$ balls.*

Inspired by [18, Th. 27], a prototype algorithm is constructed below. One can use $m$ balls to cover $Y$, thus obtain $m$ disjoint nonempty subsets $Y_1, Y_2, \cdots, Y_m$ such that for any $\epsilon > \epsilon_m(Y)$, $\forall j \in \{1, \cdots, m\}, \exists \mathbf{c}_j \in \mathbb{R}^r$, s.t. $\forall \mathbf{y} \in Y_j, \|\mathbf{c}_j - \mathbf{y}\| \leq \epsilon$ and $\bigcup_{j=1}^m Y_j = Y$. One can see that each $Y_j$ naturally forms a cluster with the center $\mathbf{c}_j$ and the index set $I_j = \{i | \mathbf{y}_i \in Y_j\}$.

Let $\alpha_i = \frac{\mathrm{w}(\mathbf{x}_q, \mathbf{x}_i)}{\sum_{j=1}^n \mathrm{w}(\mathbf{x}_q, \mathbf{x}_j)}$ and $C_j = \sum_{i \in I_j} \alpha_i$. For each cluster index set $I_j$, $j = 1, \cdots, m$, $\ell_j = \lfloor mC_j + 1 \rfloor$ many indices are randomly drawn from $I_j$ proportional to their weight $\alpha_i$. That is, for $\mu \in \{1, \cdots, \ell_j\}$, the $\mu$-th randomly drawn index $u_{j,\mu}$,

$$\Pr(u_{j,\mu} = i) = \frac{\alpha_i}{C_j}, \quad \forall j \in \{1, \cdots, m\}.$$

$\hat{\mathbf{y}}_q$ is constructed as

$$\hat{\mathbf{y}}_q = \sum_{j=1}^m \frac{C_j}{\ell_j} \sum_{\mu=1}^{\ell_j} \mathbf{y}_{u_{j,\mu}}. \tag{5}$$

*Lemma 1: There is at most $2m$ many unique $\mathbf{y}_{u_{j,\mu}}$ in $\hat{\mathbf{y}}_q$.*

*Proof:* $\sum_{j=1}^m \ell_j \leq \sum_{j=1}^m (mC_j + 1) = 2m$. ■

The following lemma shows that through the prototype algorithm the mean is preserved and the variance is small.

*Lemma 2: The following holds*

$$\mathrm{E}[\hat{\mathbf{y}}_q] = \mathbf{y}_q, \quad \mathrm{Var}(\hat{\mathbf{y}}_q) \leq \frac{\epsilon^2}{m}. \tag{6}$$

*Proof:*

$$\begin{aligned}
\mathrm{E}[\hat{\mathbf{y}}_q] &= \mathrm{E}[\sum_{j=1}^m \frac{C_j}{\ell_j} \sum_{\mu=1}^{\ell_j} \mathbf{y}_{u_{j,\mu}}] = \sum_{j=1}^m \frac{C_j}{\ell_j} \sum_{\mu=1}^{\ell_j} \mathrm{E}[\mathbf{y}_{u_{j,\mu}}] \\
&= \sum_{j=1}^m \frac{C_j}{\ell_j} \sum_{\mu=1}^{\ell_j} \sum_{i \in I_j} \frac{\alpha_i}{C_j} \mathbf{y}_i = \sum_{j=1}^m \sum_{i \in I_j} \alpha_i \mathbf{y}_i = \mathbf{y}_q. \\
\mathrm{Var}(\hat{\mathbf{y}}_q) &= \sum_{j=1}^m \sum_{\mu=1}^{\ell_j} \mathrm{Var}(\frac{C_j}{\ell_j} \mathbf{y}_{u_{j,\mu}}) \leq \sum_{j=1}^m \frac{C_j^2}{\ell_j^2} \sum_{\mu=1}^{\ell_j} \epsilon^2 \\
&= \sum_{j=1}^m \frac{C_j^2}{\ell_j} \epsilon^2 \leq \sum_{j=1}^m \frac{C_j^2}{mC_j} \epsilon^2 = \frac{\sum_{j=1}^m C_j^2}{m} \epsilon^2 = \frac{\epsilon^2}{m}.
\end{aligned}$$

■

*Theorem 1: For any even number $n' \leq n$. If Prototype Algorithm uses $n'$ many non-zero $\mathbf{y} \in Y$ to express $\hat{\mathbf{y}}_q$, then*

$$\Pr[\|\hat{\mathbf{y}}_q - \mathbf{y}_q\| \geq t] < \frac{2(\epsilon_{\frac{n'}{2}}(Y))^2}{n't^2}. \tag{7}$$

*Proof:* Via Chebyshev's inequality and Lemma 2, for any $\varrho > 0$ one gets

$$\Pr\left(\|\hat{\mathbf{y}}_q - \mathbf{y}_q\| \geq \varrho \sqrt{\mathrm{Var}(\hat{\mathbf{y}}_q)}\right) \leq \frac{1}{\varrho^2}.$$

Let $t = \varrho\sqrt{\mathrm{Var}(\hat{\mathbf{y}}_q)}$ and $\epsilon \to \epsilon_{\frac{n'}{2}}(Y)$ yields the theorem. ■

*Corollary 1: For an even number $n'$, any $\epsilon > \epsilon_{\frac{n'}{2}}(Y)$, any $\delta \in (0, 1)$ and any $t > 0$, if $n' \geq \frac{2\epsilon^2}{\delta t^2}$, then with probability at least $1 - \delta$,*

$$\|\hat{\mathbf{y}}_q - \mathbf{y}_q\| < t.$$

*Proof:* Via Theorem 1, for $\epsilon > \epsilon_{\frac{n'}{2}}(Y)$, $\Pr[\|\hat{\mathbf{y}}_q - \mathbf{y}_q\| \geq t] < \frac{2\epsilon^2}{n't^2}$. Let $\delta \geq \frac{2\epsilon^2}{n't^2}$, the following holds $n' \geq \frac{2\epsilon^2}{\delta t^2}$. ■

The quality of the approximation depends on $\epsilon_{\frac{n'}{2}}(Y)$ and $n'$. If data has strong clustering pattern, *i.e.* data within each cluster are very close to cluster center, one will have small $\epsilon_{\frac{n'}{2}}(Y)$, hence better approximation. Likewise, the bigger $n'$ is, the better approximation is.

### C. Approximation of the Prototype Algorithm

For a query point $\mathbf{x}_q$, the prototype algorithm samples from clusters and then constructs $\hat{\mathbf{y}}_q$. The clusters can be obtained via clustering algorithms such as K-means. For each cluster, the higher $C_j = \sum_{i \in I_j} \alpha_i$, the more draws are made. At least one draw is made from each cluster. Since $n$ could be potentially massive, it is impractical to rank (or compute and keep a few top ones) $\alpha_i$ within each cluster. Moreover, $\mathrm{w}(\mathbf{x}_q, \mathbf{x}_j)$ depends on $\mathbf{x}_q$ — for a different query point $\mathbf{x}_{q'}$, $\mathrm{w}(\mathbf{x}_{q'}, \mathbf{x}_i)$ may be very smaller even if $\mathrm{w}(\mathbf{x}_q, \mathbf{x}_i)$ is high. Thus one needs to consider the entire $\mathbf{X}$ instead of a single $\mathbf{x}_q$.

Recall that $\alpha_i(\mathbf{x}_q) = \frac{\mathrm{w}(\mathbf{x}_q, \mathbf{x}_i)}{\sum_{j=1}^n \mathrm{w}(\mathbf{x}_q, \mathbf{x}_j)}$. Ideally, for each cluster, one wants to select the $\mathbf{y}_i$ that has high overall weight $O_i = \sum_{\mathbf{x}_q \in \mathbf{X}} \alpha_i(\mathbf{x}_q)$. For large scale $\mathbf{X}$, the reality is that one does not have access to $\mathrm{w}(\mathbf{x}, \mathbf{x}')$ for all $\mathbf{x}, \mathbf{x}' \in \mathbf{X}$. Only limited information is available such as cluster

**Algorithm 1** Inductive Manifold-Hashing (IMH)

**Input:** Training data $\mathbf{X} := \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$, code length $r$, base set size $m$, neighborhood size $k$
**Output:** Binary codes $\mathbf{Y} := \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_n\} \in \mathbb{B}^{n \times r}$
1) Generate the base set $\mathbf{B}$ by random sampling or clustering (*e.g.* K-means).
2) Embed $\mathbf{B}$ into the low-dimensional space by (11), (14) or any other appropriate manifold leaning method.
3) Obtain the low-dimensional embedding $\mathbf{Y}$ for the whole dataset inductively by Equation (10).
4) Threshold $\mathbf{Y}$ at zero.

centers $\{\mathbf{c}_j, j \in \{1, \cdots, m\}\}$ and $\mathrm{w}(\mathbf{c}_j, \mathbf{x}), \mathbf{x} \in \mathbf{X}$. Fortunately, the clustering result gives useful information about $O_i$. The cluster centers $\{\mathbf{c}_j, j \in \{1, \cdots, m\}\}$ have the largest overall weight w.r.t the points from their own cluster, *i.e.* $\sum_{i \in I_j} \mathrm{w}(\mathbf{c}_j, \mathbf{x}_i)$. This suggests one should select all cluster centers to express $\hat{\mathbf{y}}_q$. For a base set $B$, and any query point $\mathbf{x}_q$, the embedding is predicted as

$$\hat{\mathbf{y}}_q = \frac{\sum_{\mathbf{x} \in B} \mathrm{w}(\mathbf{x}_q, \mathbf{x})\mathbf{y}}{\sum_{\mathbf{x} \in B} \mathrm{w}(\mathbf{x}_q, \mathbf{x})}. \tag{8}$$

Following many methods in the area (see [34], [53]), the general inductive hash function is formulated by binarizing the low-dimensional embedding

$$h(\mathbf{x}) = \mathrm{sgn}\left(\frac{\sum_{j=1}^{m} \mathrm{w}(\mathbf{x}, \mathbf{c}_j)\mathbf{y}_j}{\sum_{j=1}^{m} \mathrm{w}(\mathbf{x}, \mathbf{c}_j)}\right), \tag{9}$$

where $\mathrm{sgn}(\cdot)$ is the sign function and $\mathbf{Y_B} := \{\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_m\}$ is the embedding for the base set $\mathbf{B} := \{\mathbf{c}_1, \mathbf{c}_2, \cdots, \mathbf{c}_m\}$, which is the cluster centers obtained by K-means. Here the embedding $\mathbf{y}_i$ are assumed to be centered on the origin. The proposed hashing method is termed *Inductive Manifold-Hashing* (IMH). The inductive hash function provides a means for generalization to new data, which has a constant $O(dm + rk)$ time. With this, the embedding for the training data becomes

$$\mathbf{Y} = \bar{\mathbf{W}}_{\mathbf{XB}}\mathbf{Y_B}, \tag{10}$$

where $\bar{\mathbf{W}}_{\mathbf{XB}}$ is defined such that $\bar{\mathbf{W}}_{ij} = \frac{\mathrm{w}(\mathbf{x}_i, \mathbf{c}_j)}{\sum_{i=1}^{m} \mathrm{w}(\mathbf{x}_i, \mathbf{c}_j)}$, for $\mathbf{x}_i \in \mathbf{X}$, $\mathbf{c}_j \in \mathbf{B}$.

Although the objective function (2) is formally related to LE, it is generally in preserving local similarity. The embeddings $\mathbf{Y_B}$ can be learned by any appropriate manifold learning method which preserves the similarity of interest in the low-dimensional space. Several other embedding methods are empirically evaluated in Section III-F. Actually, as will be shown, some manifold learning methods (*e.g.*, t-SNE described in Section III-D) can be better choices for learning binary codes, although LE has been widely used. Two methods for learning $\mathbf{Y_B}$ will be discussed in the sequel.

The IMH framework is summarized in Algorithm 1. Note that the computational cost is dominated by K-means in the first step, which is $O(dmnl)$ in time (with $l$ the number of iterations). Considering that $m$ (normally a few hundreds) is far less than $n$, and is a function of manifold complexity rather than the volume of data, the total training time is linear in the size of training set. If the embedding method is LE, for example, then using IMH to compute $\mathbf{Y_B}$ requires constructing the small affinity matrix $\mathbf{W_B}$ and solving $r$ eigenvectors of the $m \times m$ Laplacian matrix $\mathbf{L_B}$ which is $O(dm^2 + rm)$. Note that in step 3, to compute $\bar{\mathbf{W}}_{\mathbf{XB}}$, one needs to compute the distance matrix between $\mathbf{B}$ and $\mathbf{X}$, which is the output of K-means, or can be computed additionally in $O(dmn)$ time. The training process on a dataset of 70K items with 784 dimensions can thus be achieved in a few seconds on a standard desktop PC.

**Connection to the Nyström Method:** As Equation (4), the Nyström eigenfunction by Bengio et al. [2] also generalizes to a new point by a linear combination of a set of low-dimensional embeddings:

$$\phi(\mathbf{x}) = \sqrt{n} \sum_{j=1}^{n} \tilde{\mathrm{K}}(\mathbf{x}, \mathbf{x}_j)\mathbf{V}_r^j \Sigma_r^{-1}.$$

For LE, $\mathbf{V}_r$ and $\Sigma_r$ correspond to the top $r$ eigenvectors and eigenvalues of a normalized kernel matrix $\tilde{K}$ with

$$\tilde{K}_{ij} = \tilde{k}(\mathbf{x}_i, \mathbf{x}_j) = \frac{1}{n} \frac{\mathrm{w}(\mathbf{x}_i, \mathbf{x}_j)}{\sqrt{E_{\mathbf{x}}[\mathrm{w}(\mathbf{x}_i, \mathbf{x})]E_{\mathbf{x}}[\mathrm{w}(\mathbf{x}, \mathbf{x}_j)]}}.$$

In AGH [34], the formulated hash function was proved to be the corresponding Nyström eigenfunction with the approximate low-rank affinity matrix. The Laplacian eigenmaps latent variable model (LELVM) [5] also formulated the out-of-sample mappings for LE in a manner similar to (4) by combining latent variable models. Both of these methods, and the proposed one, can thus be seen as applications of the Nyström method. *Note, however, that the suggested method differs in that it is not restricted to spectral methods such as LE, and that the present study aims to learn binary hash functions for similarity-based search rather than dimensionality reduction.* LELVM [5] cannot be applied to other embedding methods other than LE.

### *D. Stochastic Neighborhood Preserving Hashing*

In order to demonstrate the effectiveness of the proposed approach, a hashing method based on t-SNE [47], a non-spectral embedding method, is derived below. T-SNE is a modification of stochastic neighborhood embedding (SNE) [19] which aims to overcome the tendency of that method to crowd points together in one location. t-SNE provides an effective technique for visualizing data and dimensionality reduction, which is capable of preserving local structures in the high-dimensional data while retaining some global structures [47]. These properties make t-SNE a good choice for nearest neighbor search. Moreover, as stated in [48], the cost function of t-SNE in fact maximizes the *smoothed recall* [48] of query points and their neighbors.

The original t-SNE does not scale well, as it has a time complexity which is quadratic in $n$. More significantly, however, it has a non-parametric form, which means that there is no simple function which may be applied to out-of-sample data in order to calculate their coordinates in the embedded space. As was proposed in the previous subsection, one first applies t-SNE [47] to the base set $\mathbf{B}$,

$$\min_{\mathbf{Y_B}} \sum_{\mathbf{x}_i \in \mathbf{B}} \sum_{\mathbf{x}_j \in \mathbf{B}} p_{ij} \log\left(\frac{p_{ij}}{q_{ij}}\right). \tag{11}$$

Here $p_{ij}$ is the symmetrized conditional probability in the high-dimensional space, and $q_{ij}$ is the joint probability defined using the t-distribution in the low-dimensional embedding space. The optimization problem (11) is easily solved by a gradient descent procedure.[1] After obtaining the embeddings $\mathbf{Y_B}$ of samples $\mathbf{x}_i \in \mathbf{B}$, the hash codes for the entire dataset can be easily computed using (9). This method is labelled IMH-tSNE.

### *E. Hashing With Relaxed Similarity Preservation*

As in the last subsection, one can compute $\mathbf{Y_B}$ considering local smoothness only within $\mathbf{B}$. Based on equation (4), in this subsection, $\mathbf{Y_B}$ is alternatively computed by considering the smoothness both within $\mathbf{B}$ and between $\mathbf{B}$ and $\mathbf{X}$. As in [8], the objective can be easily obtained by modifying (1) as:

$$\begin{aligned}\mathcal{C}(\mathbf{Y_B}) = &\sum_{\mathbf{x}_i,\mathbf{x}_j\in\mathbf{B}} \mathrm{w}(\mathbf{x}_i,\mathbf{x}_j)\|\mathbf{y}_i-\mathbf{y}_j\|^2 \quad (\mathcal{C}_{\mathbf{BB}})\\ &+\lambda\sum_{\mathbf{x}_i\in\mathbf{B},\mathbf{x}_j\in\mathbf{X}} \mathrm{w}(\mathbf{x}_i,\mathbf{x}_j)\|\mathbf{y}_i-\mathbf{y}_j\|^2 \quad (\mathcal{C}_{\mathbf{BX}}) \quad (12)\end{aligned}$$

where $\lambda$ is the trade-off parameter. $\mathcal{C}_{\mathbf{BB}}$ enforces smoothness of the learned embeddings within $\mathbf{B}$ while $\mathcal{C}_{\mathbf{BX}}$ ensures the smoothness between $\mathbf{B}$ and $\mathbf{X}$. This formulation is actually a relaxation of (1), by discarding the part which minimizes the dissimilarity within $\mathbf{X}$ (denoted as $\mathcal{C}_{\mathbf{XX}}$). $\mathcal{C}_{\mathbf{XX}}$ is ignored since computing the similarity matrix within $\mathbf{X}$ costs $O(n^2)$ time. The smoothness between points in $\mathbf{X}$ is implicitly ensured by (10).

Applying equation (10) for $\mathbf{y}_j$, $j \in \mathbf{X}$ to (12), one obtains the following problem

$$\begin{aligned}\min\ &\mathrm{trace}(\mathbf{Y_B^\top}(\mathbf{D_B}-\mathbf{W_B})\mathbf{Y_B})\\ &+\lambda\,\mathrm{trace}(\mathbf{Y_B^\top}(\mathbf{D_{BX}}-\bar{\mathbf{W}}_{\mathbf{XB}}^\top\mathbf{W_{XB}})\mathbf{Y_B}), \quad (13)\end{aligned}$$

where $\mathbf{D_B} = \mathrm{diag}(\mathbf{W_B1})$ and $\mathbf{D_{BX}} = \mathrm{diag}(\mathbf{W_{BX}1})$ are both $m \times m$ diagonal matrices. Taking the constraint in (1), one obtains

$$\begin{aligned}\min_{\mathbf{Y_B}}\ &\mathrm{trace}(\mathbf{Y_B^\top}(\mathbf{M}+\lambda\mathbf{T})\mathbf{Y_B})\\ \text{s.t.}\ &\mathbf{Y_B^\top Y_B} = m\mathbf{I} \quad (14)\end{aligned}$$

where $\mathbf{M} = \mathbf{D_B} - \mathbf{W_B}$, $\mathbf{T} = \mathbf{D_{BX}} - \bar{\mathbf{W}}_{\mathbf{XB}}^\top\mathbf{W_{XB}}$. The optimal solution $\mathbf{Y_B}$ of the above problem is easily obtained by identifying the $r$ eigenvectors of $\mathbf{M}+\lambda\mathbf{T}$ corresponding to the smallest eigenvalues (excluding the eigenvalue 0 with respect to the trivial eigenvector $\mathbf{1}$).[2] This method is named IMH-LE in the following text.

### *F. Manifold Learning Methods for Hashing*

In this section, different manifold learning methods are compared for hashing within the proposed IMH framework. Figure 2 reports the comparative results in mean of average precision (MAP). For comparison, linear PCA is also evaluated within the framework (IMH-PCA in the figure).

[1]See details in [47]. A Matlab implementation of t-SNE is provided by the authors of [47] at http://homepage.tudelft.nl/19j49/t-SNE.html.

[2]The parameter $\lambda$ is set to 2 in all experiments.

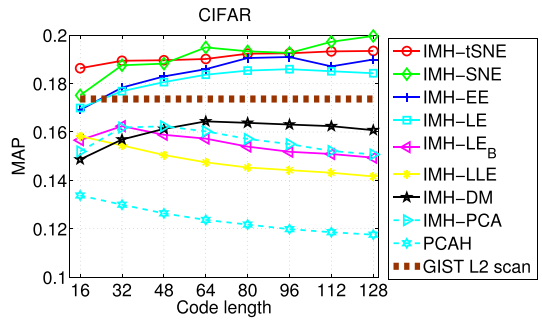



Fig. 2. Comparison among different manifold learning methods within the proposed IMH hashing framework on CIFAR-10. IMH with linear PCA (IMH-PCA) and PCAH [49] are also evaluated for comparison. For clarity, for IMH-LE in Section III-E, IMH with the original LE algorithm on the base set $\mathbf{B}$ is termed as IMH-LE$_\mathbf{B}$. IMH-DM is IMH with the diffusion maps of [27]. The base set size is set to 400.

As can be clearly seen that, IMH-tSNE, IMH-SNE and IMH-EE perform slightly better than IMH-LE (Section III-E). This is mainly because these three methods are able to preserve local neighborhood structure while, to some extent, preventing data points from crowding together. It is promising that all of these methods perform better than an exhaustive $\ell_2$ scan using the uncompressed GIST features.

Figure 2 shows that LE (IMH-LE$_\mathbf{B}$ in the figure), the most widely used embedding method in hashing, does not perform as well as a variety of other methods (*e.g.*, t-SNE), and in fact performs worse than PCA, which is a linear technique. This is not surprising because LE (and similarly LLE) tends to collapse large portions of the data (and not only nearby samples in the original space) close together in the low-dimensional space. The results are consistent with the analysis in [4] and [47]. Based on the above observations, we argue that manifold learning methods (*e.g.* t-SNE, EE), which not only preserve local similarity but also force dissimilar data apart in the low-dimensional space, are more effective than the popular LE for hashing. Preserving the global structures of the data is critical for the manifold learning algorithms like t-SNE used in the proposed IMH framework. The previous work called spline regression hashing (SRH) [36] also exploited both the local and global data similarity structures of data via a Laplacian matrix, which can decrease over-fitting, as discussed in [36]. SRH captures the global similarity structure of data through constructing global non-linear hash functions, while, in the proposed method, capturing the global structures (*i.e.*, by t-SNE) is only applied on the small base set and the hash function is formulated by a locally linear regression model.

It is interesting to see that IMH-PCA outperforms PCAH [49] by a large margin, despite the fact that PCA is performed on the whole training data set by PCAH. This shows that the generalization capability of IMH based on a very small set of data points also works for linear dimensionality methods.

## IV. Experimental Results

IMH is evaluated on four large scale image datasets: CIFAR-10,[3] MNIST, SIFT1M [49] and GIST1M.[4]

[3]http://www.cs.toronto.edu/~kriz/cifar.html

[4]http://corpus-texmex.irisa.fr/

TABLE I
MAP (%) OF DIFFERENT BASE GENERATING METHODS: RANDOM SAMPLING VS. K-MEANS. THE COMPARISON IS PERFORMED ON THE CIFAR-10 DATASET WITH CODE LENGTHS FROM 32 TO 96 AND BASE SET SIZE 400. AVERAGE RESULTS (STANDARD DEVIATION) ARE GIVEN BASED ON 10 RUNS

| Bits | Method | IMH-LE$_\mathbf{B}$ | IMH-LE | IMH-tSNE | AGH |
|---|---|---|---|---|---|
| 32 | Random | 14.07 (0.27) | 16.20 (0.19) | 17.26 (0.43) | - |
| | K-medians | 15.14 (0.15) | 17.08 (0.21) | 17.32 (0.27) | - |
| | K-means | 16.05 (0.16) | 17.48 (0.44) | **18.38 (0.36)** | 15.76 (0.17) |
| 64 | Random | 14.64 (0.17) | 16.98 (0.21) | 16.93 (0.49) | - |
| | K-medians | 15.43 (0.12) | 17.11 (0.17) | 17.62 (0.31) | - |
| | K-means | 15.90 (0.11) | 18.20 (0.37) | **19.04 (0.27)** | 14.55 (0.13) |
| 96 | Random | 14.76 (0.20) | 17.02 (0.23) | 17.21(0.48) | - |
| | K-medians | 15.42 (0.19) | 17.21 (0.28) | 17.73 (0.28) | - |
| | K-means | 15.46 (0.09) | 18.56 (0.34) | **19.41 (0.23)** | 13.98 (0.10) |

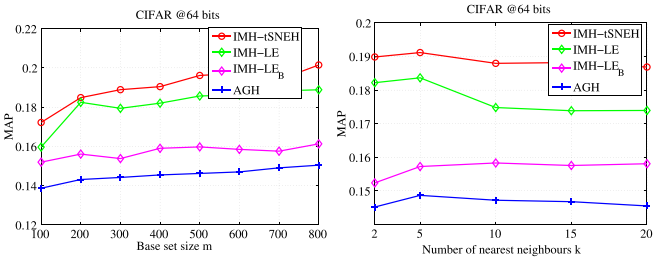



Fig. 3. MAP results versus varying base set size $m$ (left, fixing $k = 5$) and number of nearest base points $k$ (right, fixing $m = 400$) for the proposed methods and AGH. The comparison is conducted on the CIFAR-10 dataset using 64-bits.

The MNIST dataset consists of 70,000 images, each of 784 dimensions, of handwritten digits from '0' to '9'. As a subset of the well-known 80M tiny image collection [46], CIFAR-10 consists of 60,000 images which are manually labelled as 10 classes with 6,000 samples for each class. Each image in this dataset is represented by a GIST feature vector [39] of dimension 512. For MNIST and CIFAR-10, the whole dataset is split into a test set with 1,000 samples and a training set with all remaining samples.

Nine hashing algorithms are compared, including the proposed IMH-tSNE, IMH-LE and seven other unsupervised state-of-the-art methods: PCAH [49], SH [53], AGH [34] and STH [55], BRE [24], ITQ [13], Spherical Hashing (SpH) [17]. The provided codes and suggested parameters according to the authors of these methods are used. Due to the high computational cost of BRE and high memory cost of STH, 1,000 and 5,000 training points are sampled for these two methods respectively. The performance is measured by MAP or precision and recall curves for *Hamming ranking* using 16 to 128 hash bits. The results for *hash lookup* using a Hamming radius within 2 by F1 score [37]: $F_1 = 2(precision \cdot recall)/(precision + recall)$ are also reported. Ground truths are defined by the category information for the labeled datasets MNIST and CIFAR-10, and by Euclidean neighbors for SIFT1M and GIST1M.

### A. Base Selection

In this section, the CIFAR-10 dataset is taken as an example to compare different base generation methods and different base sizes for the proposed methods. AGH is also evaluated here for comparison. Table I compares three methods for generating base point sets: random sampling, K-medians and K-means on the training data. One can easily see that the performance of the proposed methods using K-means is better at all code lengths than that using other two methods. It is also clear from Table I that the K-medians algorithm achieves better performance than random sampling, however still inferior to K-means.

Different from K-means that constructs each base point by averaging the data points in the corresponding cluster, random sampling and K-medians generate the base set with real data samples in the training set. However, this property does not help K-medians obtain better hash codes than K-means. This is possibly because K-means produces the cluster centers with minimum quantization distortions at each group of the data points. Another advantage of K-means is it is much more efficient than K-medians.

As can also be seen, even with a base set by random sampling, the proposed methods outperform AGH in all cases but one. Due to the superior results and high efficiency in practice, the base set is generated by K-means in the following experiments.

From Figure 3, it is clear that the performance of the proposed methods is consistently improved with increasing base set size $m$, which is consistent with the analysis of the prototype algorithm. One can observe that, the performance does not change significantly when varying the number of nearest base points $k$. It is also clear that IMH-LE$_\mathbf{B}$, which only enforces smoothness in the base set, does not perform as well as IMH-LE, which also enforces smoothness between the base set and training set.

To further investigate the impact of the base set size $m$, in both performance and efficiency, the proposed methods are performed with a large variance of $m$ on CIFAR-10 and MNIST. The results for the proposed IMH-tSNE are shown in Table II. As can be clearly seen, for the dataset of CIFAR-10, the MAP score by IMH-tSNE improves consistently with the increasing base set size $m$ when $m \leq 400$, while does not change dramatically with larger $m$. On MNIST, the performance of IMH-tSNE is also consistently improved when $m$ increases. Same as on CIFAR-10, the MAP score does not significantly change with larger $m$.

In terms of computational efficiency, on both datasets, the training time cost is considerably increased with larger $m$. For example, IMH-tSNE costs about 4.3 seconds with 400 base samples, while costs more than 21 seconds with 1,000 base samples on the CIFAR-10 dataset. For the retrieval task, the testing time is more crucial. As shown in Table II, the testing time is in general linearly increased with $m$. With a small $m$, the testing is very efficient. When with a large $m$ (*e.g.*, $m \geq 3000$), IMH-tSNE needs more than one milliseconds to compute the binary codes of the query, which is not scalable for large-scale tasks. Take both performance and computational efficiency into account, for the remainder

TABLE II

IMPACT OF THE BASE SET SIZE $m$ OF IMH-tSNE ON THE RETRIEVAL PERFORMANCE AND COMPUTATIONAL EFFICIENCY. THE BASE SET IS GENERATED BY K-MEANS. THE RESULTS WITH 64 BITS ARE REPORTED, BASED ON 10 INDEPENDENT RUNS. THE EXPERIMENTS ARE CONDUCTED ON A DESKTOP PC WITH A 4-CORE 3.40GHz CPU AND 32G RAM

| | $m$ | 100 | 200 | 400 | 800 | 1000 | 1500 | 2000 | 3000 | 4000 | 5000 | 10000 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CIFAR | MAP (%) | 17.55 | 18.19 | 19.52 | 19.99 | 20.11 | 20.13 | 20.05 | 20.23 | 20.37 | 20.45 | 20.85 |
| | Training Time (s) | 0.7754 | 1.3913 | 4.2850 | 13.5576 | 21.3487 | 53.5668 | 97.6969 | 229.6100 | 419.4027 | 664.0867 | 2701.8001 |
| | Testing Time (ms) | 0.0057 | 0.0095 | 0.0157 | 0.0254 | 0.0308 | 0.0475 | 0.0621 | 0.1045 | 0.1388 | 0.1981 | 0.3634 |
| MNIST | MAP (%) | 57.01 | 67.72 | 77.78 | 79.71 | 80.40 | 82.52 | 82.10 | 86.47 | 85.66 | 87.37 | 87.74 |
| | Training Time (s) | 0.9546 | 1.6804 | 4.6302 | 814.7084 | 22.7297 | 54.7796 | 102.1419 | 232.8753 | 423.6395 | 669.6443 | 2707.8012 |
| | Testing Time (ms) | 0.0074 | 0.0122 | 0.0182 | 0.0330 | 0.0394 | 0.0579 | 0.0984 | 0.1222 | 0.2032 | 0.2293 | 0.4357 |

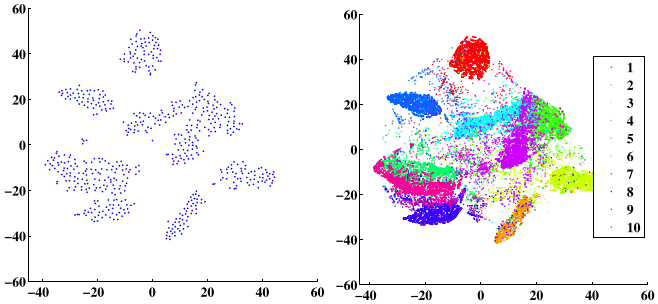


Fig. 4. Visualization of the digits in MNIST in 2D by t-SNE: (left) t-SNE embeddings of the base set (400 data points generated by K-means); (right) embeddings of all the 60,000 samples computed by the proposed inductive method. In the right, the labels of the digits are indicated with 10 different colors.

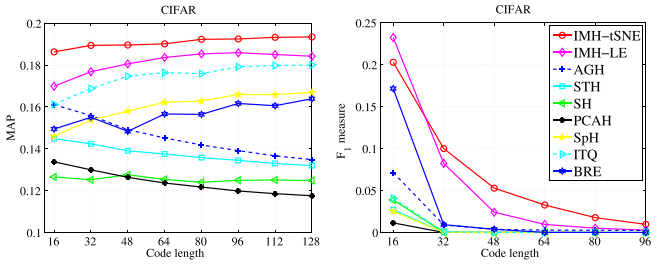



Fig. 5. Comparison of different methods on CIFAR-10 based on MAP (left) and $F_1$ (right) for varying code lengths.

of this paper, the settings $m = 400$ and $k = 5$ are used for the proposed methods, unless otherwise specified.

Figure 4 shows the t-SNE embeddings of the base set (400 points generated by K-means) and the embeddings of the whole MNIST dataset computed by the proposed inductive method (without binarization). As can be seen from the right figure that most of the points are close to their corresponding clusters (with respect to 10 digits). This observation shows that, in the low-dimensional embedding space, the proposed method can well preserve the local manifold structure of the entire dataset based on a relatively small base set.

### B. Results on CIFAR-10 Dataset

The comparative results based on MAP for Hamming ranking with code lengths from 16 to 128 bits are reported in Figure 5. It can be seen that the proposed IMH-LE and IMH-tSNE perform best in all cases. Among the proposed

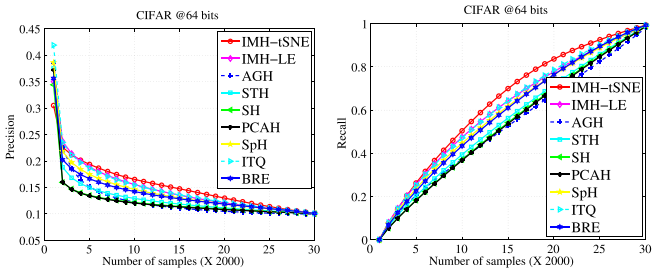



Fig. 6. Comparison of different methods on CIFAR-10 based on precision (left) and recall (right) using 64-bits. Please refer to the complementary for complete results for other code lengths.

algorithms, the LE based IMH-LE is inferior to the t-SNE based IMH-tSNE. IMH-LE is still much better than AGH and STH, however. ITQ performs better than SpH and BRE on this dataset, but is still inferior to IMH. SH and PCAH perform worst in this case. This is because SH relies upon its uniform data assumption while PCAH simply generates the hash hyperplanes by PCA directions, which does not explicitly capture the similarity information. The results are consistent with the complete precision and recall curves shown in the supplementary material. The $F_1$ results for hash lookup with Hamming radius 2 are also reported. It is can be seen that IMH-LE and IMH-tSNE also outperform all other methods by large margins. BRE and AGH obtain better results than the remaining methods, although the performance of all methods drop as code length grows.

Figure 6 shows the precision and recall curves of Hamming ranking for the compared methods. STH and AGH obtain relatively high precisions when a small number of samples are returned, however precision drops significantly as the number of retrieved samples increases. In contrast, IMH-tSNE, IMH-LE and ITQ achieve higher precisions with relatively larger numbers of retrieved points.

### C. Results on MNIST Dataset

The MAP and $F_1$ scores for these compared methods are reported in Figure 7. As in Figure 5, IMH-tSNE achieves the best results. It is clear that, on this dataset IMH-tSNE outperforms IMH-LE by a large margin, which increases as the code length increases. This further demonstrates the advantage of t-SNE as a tool for hashing by embedding high-dimensional data into a low-dimensional space. The dimensionality reduction procedure not only preserves the local neighborhood

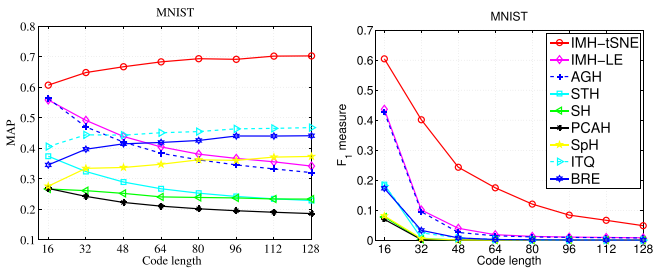



Fig. 7. Comparison of different methods on the MNIST dataset using MAP (left) and $F_1$ (right) for varying code lengths.

TABLE III
COMPARISON OF TRAINING AND TESTING TIMES (IN SECONDS) ON MNIST WITH 70K 784D FEATURE POINTS. THE REPORTED RESULTS FOR AGH AND IMH INCLUDE THE TIME TAKEN BY THE DOMINATING K-MEANS (8.9 SECONDS), WHICH CAN BE CONDUCTED IN ADVANCE IN PRACTICE THOUGH. THE EXPERIMENTS ARE BASED ON A DESKTOP PC WITH A 4-CORE 3.07GHz CPU AND 8G RAM

| Method | Train time | | Test time | |
|---|---|---|---|---|
| | 64-bits | 128-bits | 64-bits | 128-bits |
| IMH-LE | 9.9 | 9.9 | $5.1 \times 10^{-5}$ | $3.8 \times 10^{-5}$ |
| IMH-tSNE | 16.7 | 20.2 | $2.8 \times 10^{-5}$ | $3.1 \times 10^{-5}$ |
| SH | 6.8 | 16.2 | $5.8 \times 10^{-5}$ | $1.8 \times 10^{-4}$ |
| STH | 266.1 | 485.4 | $1.8 \times 10^{-3}$ | $3.6 \times 10^{-3}$ |
| AGH | 9.5 | 9.5 | $4.7 \times 10^{-5}$ | $5.5 \times 10^{-5}$ |
| PCAH | 3.8 | 4.1 | $5.7 \times 10^{-6}$ | $1.2 \times 10^{-5}$ |
| SpH | 19.7 | 41.0 | $1.3 \times 10^{-5}$ | $2.0 \times 10^{-5}$ |
| ITQ | 10.4 | 20.3 | $6.9 \times 10^{-6}$ | $1.1 \times 10^{-5}$ |
| BRE | 418.9 | 1731.9 | $1.2 \times 10^{-5}$ | $2.4 \times 10^{-5}$ |

structure, but also reveals important global structure (such as clusters) [47]. Among the four LE-based methods, while IMH-LE shows a small advantage over AGH, both methods achieve much better results than STH and SH. ITQ and BRE obtain high MAPs with longer bit lengths, but they still perform less well for the hash look up $F_1$. PCAH performs worst in terms of both MAP and the $F_1$ measure. Refer to the supplementary material for the complete precision and recall curves which validate the observations here.

*Efficiency* Table III shows training and testing time on the MNIST dataset for various methods, and shows that the linear method, PCAH, is fastest. IMH-tSNE is slower than IMH-LE, AGH and SH in terms of training time, however all of these methods have relatively low execution times and are much faster than STH and BRE. In terms of test time, both IMH algorithms are comparable to other methods, except STH which takes much more time to predict the binary codes by SVM on this non-sparse dataset.

### *D. Results on SIFT1M and GIST1M*

SIFT1M contains one million local SIFT descriptors extracted from a large set of images [49], each of which is represented by a 128D vector of histograms of gradient orientations. GIST1M contains one million GIST features and each feature is represented by a 960D vector. For both of these datasets, one million samples are used as training set and additional 10K are used for testing. As in [49], ground truth is defined as the closest 2 percent of points as measured by the Euclidean distance. For these two large datasets, 1, 000 points are generated by K-means and $k$ is set as 2 for both IMH and AGH. The comparative results on SIFT1M and GIST1M are summarized in Figure 8 and Figure 9, respectively. Again, IMH consistently achieves superior results in terms of both $F_1$ score and recall with Hamming radius 2. As can be seen that, the performance of most of these methods decreases dramatically with increasing code length as the Hamming spaces become more sparse, which makes the hash lookup fail more often. However, IMH-tSNE still achieves relatively high scores with large code lengths. If one looks at Figure 8 (left), ITQ obtains the highest $F_1$ with 16-bits, however it decreases to near zero at 64-bits. In contrast, IMH-tSNE still manages an $F_1$ of 0.2. Similar results are observed in the recall curves.

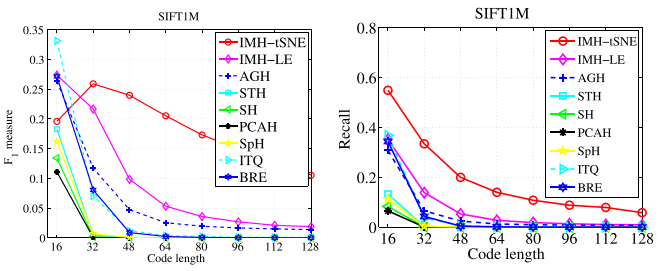



Fig. 8. Comparative results results on SIFT1M for $F_1$ (left) and recall (right) with Hamming radius 2. Ground truth is defined to be the closest 2 percent of points as measured by the Euclidean distance.

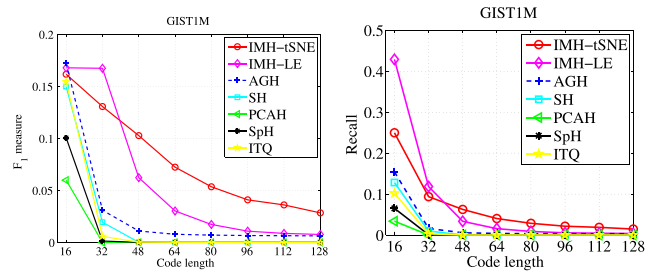



Fig. 9. Comparative results results on GIST1M by $F_1$ (left) and recall (right) with Hamming radius 2. Ground truth is defined to be the closest 2 percent of points as measured by the Euclidean distance.

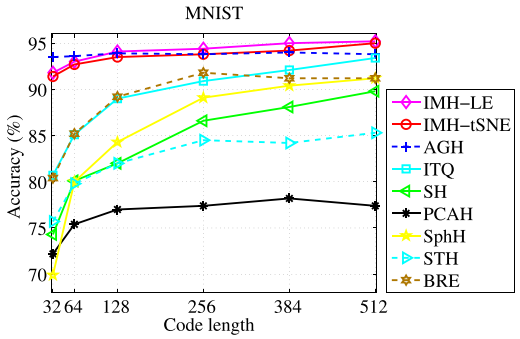



Fig. 10. Classification accuracy (%) on MNIST with binary codes of various hashing methods by linear SVM.

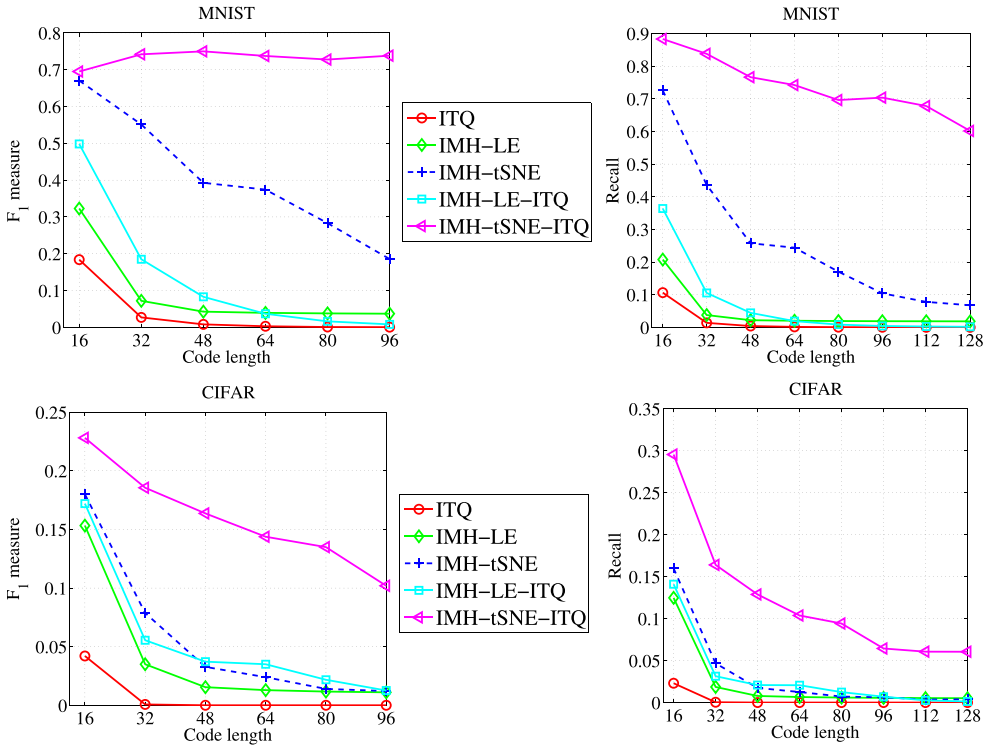



Fig. 11. Evaluation of IMH with ITQ rotations on the MNIST and CIFAR dataset using $F_1$ and *recall* with Hamming radius 2 for varying code lengths.

### *E. Classification on Binary Codes*

In order to demonstrate classification performance a linear SVM is trained on the binary codes generate by IMH for the MNIST data set. In order to learn codes with higher bit lengths for IMH and AGH, the base set size is set to $1,000$. Accuracies of different binary encodings are shown in Figure 10. Both IMH and AGH achieve high accuracies on this dataset, although IMH performs better with higher code lengths. In contrast, the best results of all other methods, obtained by ITQ, are consistently worse than those for IMH, especially for short code lengths. Note that even with only 128-bit binary features IMH obtains a high 94.1%. Interestingly, the same classification rate of 94.1% is obtained by applying the linear SVM to the uncompressed 784D features, which occupy several hundreds times as much space as the learned hash codes.

## V. Minimizing the Quantization Distortion by Learned Rotations

In the above sections, the binary codes are obtained by directly thresholding the learned embeddings at zeros. The simple binarization may cause large quantization loss. In this section, learned rotations are applied to minimize the quantization errors. That is, normalize the data points such that they are zero-centred in the embedded space and then orthogonally rotate the normalized embeddings for binarization. Orthogonal rotation has been adopted by various methods (see [9], [13], [15], [21]) to minimize the quantization error. The simple algorithm in ITQ [13] is used for the proposed method.

The impact of the rotations applied on the learned embeddings is evaluated on MNIST and CIFAR. Figure 11 clearly shows that the orthogonal rotations achieve significantly performance improvements (in terms of both $F_1$ and recall) for IMH-LE and IMH-tSNE. In conjunctions with the rotations, the proposed IMH-LE and IMH-tSNE methods perform much better than the PCA based PCA-ITQ. Again this result demonstrates the advantages of the proposed manifold hashing method.

**Algorithm 2** Supervised Inductive Manifold-Hashing (IMHs)

**Input:** Unlabelled data $\mathbf{X} := \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$, labelled data $\mathbf{Xs}$ of $t$ classes, code length $r$, base set size $m$, neighborhood size $k$

**Output:** Binary codes $\mathbf{Y} := \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_n\} \in \mathbb{B}^{n \times r}$

1) Generate the base set $\mathbf{B} := \{\mathbf{c}_{1,1}, \cdots, \mathbf{c}_{m_1,1}, \cdots, \mathbf{c}_{1,t} \cdots, \mathbf{c}_{m_t,t}\}$ by K-means on data points from each class of $\mathbf{Xs}$.

2) Embed $\mathbf{B}$ into the low-dimensional space by (11), (14) or any other appropriate manifold leaning method and get $\mathbf{Y_B}$.

3) Apply supervised subspace learning algorithms such as LDA on $\mathbf{Y_B}$.

4) Obtain the low-dimensional embedding $\mathbf{Y}$ for the whole dataset inductively by Equation (10).

5) Threshold $\mathbf{Y}$ at zero.

## VI. Semantic Hashing With Supervised Manifold Learning

The proposed inductive manifold hashing algorithm has shown to work well on preserving the semantic neighborhood relationships without using label information. It is expected that the performance can be improved by applying supervised learning methods instead of the unsupervised ones to learn the nonlinear embeddings. A straightforward supervised extension to the proposed IMH algorithm is proposed in this study. First, the base set $\mathbf{B} := \{\mathbf{c}_{1,1}, \cdots, \mathbf{c}_{m_1,1}, \cdots, \mathbf{c}_{1,t} \cdots, \mathbf{c}_{m_t,t}\}$ is generated by applying K-means on data from each of

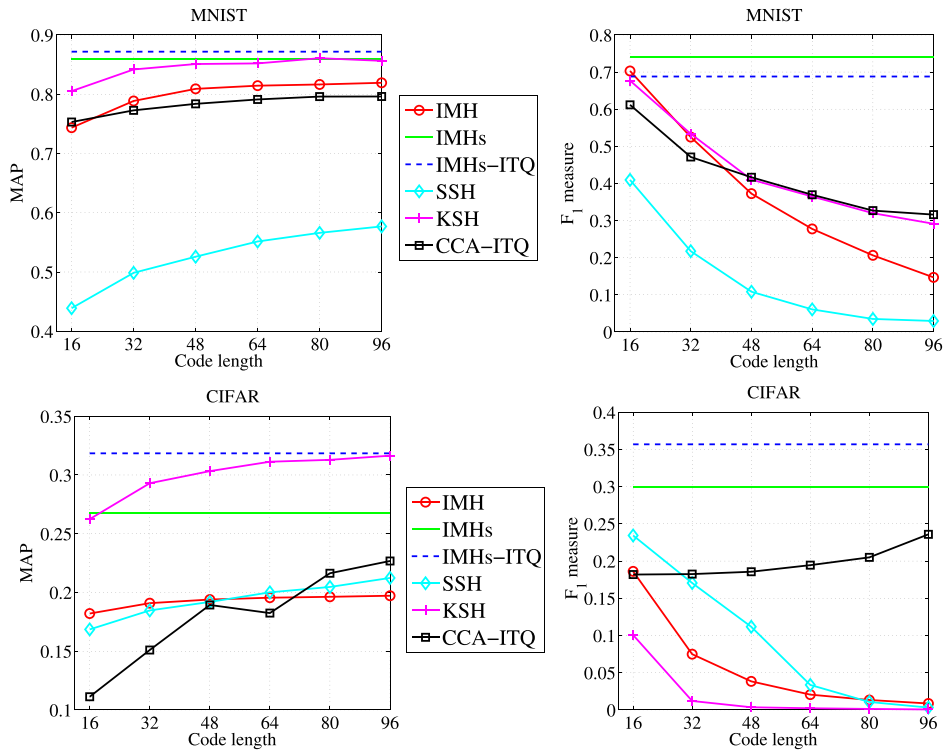



Fig. 12. Evaluation of IMH with supervised learning by LDA on the MNIST and CIFAR datasets. Non-linear embeddings of IMH are obtained by t-SNE. Since there are only 10 classes with both these two datasets, the reduced dimensionality by LDA (thereby the binary code lenght) is set to 9.

the $t$ classes. After the nonlinear embeddings $\mathbf{Y_B}$ of $\mathbf{B}$ are obtained, the supervised subspace learning algorithms are simply conducted on $\mathbf{Y_B}$. For a new data point, its binary codes are then obtained by (9). The supervised manifold hashing method is summarized in Algorithm 2.

In this section, linear discriminant analysis (LDA) is taken as an example to verify the efficacy of the proposed supervised manifold hashing algorithm. The proposed method is also compared with several recently proposed supervised hashing approaches such as semi-supervised hashing (SSH [49]) with sequential projection learning, kernel-based supervised hashing (KSH [33]) and ITQ with Canonical Correlation Analysis (CCA-ITQ [14]).

Experiments are performed on MNIST and CIFAR. In this experiment, 2,000 labelled samples are randomly selected for supervised learning for SSH and KSH, and 1,000 labelled samples are sampled for the base set for IMHs. All labelled training data are used for the linear CCA-ITQ. Since there are only 10 classes with both these two datasets, the reduced dimensionality by LDA (thereby the binary code length) in the proposed IMHs is fixed at 9. The results are reported in Figure 12. It is clearly that the proposed supervised inductive manifold hashing algorithm IMHs significantly improve the original IMH and other compared supervised methods in both MAP and $F_1$ measure. The ITQ rotations (IMHs-ITQ in the figure) further improves IMHs with considerable gains, especially on the CIFAR dataset of natural images. Among other supervised hashing methods, KSH obtains the highest MAPs on MNIST and CIFAR with IMHs or IMHs-ITQ. However, it needs much larger binary code lengths to achieve comparable performance with the proposed methods. In terms of $F_1$, CCA-ITQ is identified to have the best results with long codes.

From these results, it is clear that label information is very useful to achieve semantically effective hashing codes. Further more, the proposed simple supervised hashing framework can effectively leverage the supervised information in the proposed manifold hashing algorithms. Also note that the proposed method does not assume a specific algorithm like LDA, any other supervised subspace learning or metric learning algorithms may further improve the performance.

## VII. Conclusion and Discussion

We have proposed a simple yet effective hashing framework, which provides a practical connection between manifold learning methods (typically non-parametric and with high computational cost) and hash function learning (requiring high efficiency). By preserving the underlying manifold structure with several non-parametric dimensionality reduction methods, the proposed hashing methods outperform several state-of-the-art methods in terms of both hash lookup and Hamming ranking on several large-scale retrieval-datasets. The proposed inductive hashing methods require only linear time ($O(n)$) for indexing all of the training data and a constant search time for a novel query. Experiments showed that the hash codes can also achieve promising results on a classification problem even with very short code lengths. The proposed inductive manifold hashing method was then extended by applying orthogonal rotations on the learned nonlinear embeddings to minimize the quantization errors,

which was shown to achieve significant performance improvements. In addition, this work further extended IMH by adopting supervised subspace learning on the data manifolds, which provides an effective supervised manifold hashing framework.

The proposed hashing methods have been shown to work well on image retrieval and classification tasks. As an efficient and effective nonlinear feature extraction method, this algorithm can also be applied to other applications, especially the ones need to deal with large dataset. For example, the introduced hashing techniques can be applied to large-scale mobile video retrieval [31]. Another useful application of the hashing methods would be compressing the high-dimensional features into short binary codes, which could significantly speed up the potential tasks, such as the large scale ImageNet image classification [12], [29].

In this work, the base set size $m$ was set empirically, which was not an optimal choice apparently. How to automatically set this parameter according to the size and distribution of a specific dataset deserves a future study.

## REFERENCES


[1] M. Belkin and P. Niyogi, "Laplacian eigenmaps and spectral techniques for embedding and clustering," in *Advances in Neural Information Processing Systems*. Cambridge, MA, USA: MIT Press, 2001, pp. 585–591.

[2] Y. Bengio, O. Delalleau, N. Le Roux, J.-F. Paiement, P. Vincent, and M. Ouimet, "Learning eigenfunctions links spectral embedding and kernel PCA," *Neural Comput.*, vol. 16, no. 10, pp. 2197–2219, 2004.

[3] C. Strecha, A. M. Bronstein, M. M. Bronstein, and P. Fua, "LDAHash: Improved matching with smaller descriptors," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 1, pp. 66–78, Jan. 2012.

[4] M. A. Carreira-Perpinán, "The elastic embedding algorithm for dimensionality reduction," in *Proc. Int. Conf. Mach. Learn.*, 2010, pp. 167–174.

[5] M. Carreira-Perpinán and Z. Lu, "The Laplacian eigenmaps latent variable model," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2007, pp. 59–66.

[6] R. Chaudhry and Y. Ivanov, "Fast approximate nearest neighbor methods for non-Euclidean manifolds with applications to human activity analysis in videos," in *Proc. 11th Eur. Conf. Comput. Vis.*, 2010, pp. 735–748.

[7] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni, "Locality-sensitive hashing scheme based on p-stable distributions," in *Proc. 20th Annu. Symp. Comput. Geometry*, 2004, pp. 253–262.

[8] O. Delalleau, Y. Bengio, and N. Le Roux, "Efficient non-parametric function induction in semi-supervised learning," in *Proc. 10th Int. Workshops Artif. Intell. Statist.*, 2005, pp. 96–103.

[9] T. Ge, K. He, Q. Ke, and J. Sun, "Optimized product quantization for approximate nearest neighbor search," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 2946–2953.

[10] T. Ge, K. He, and J. Sun, "Graph cuts for supervised binary coding," in *Proc. 13th Eur. Conf. Comput. Vis.*, 2014, pp. 250–264.

[11] A. Gionis, P. Indyk, and R. Motwani, "Similarity search in high dimensions via hashing," in *Proc. 25th Int. Conf. Very Large Data Bases*, 1999, pp. 518–529.

[12] 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.

[13] Y. Gong and S. Lazebnik, "Iterative quantization: A procrustean approach to learning binary codes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2011, pp. 817–824.

[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] K. He, F. Wen, and J. Sun, "K-means hashing: An affinity-preserving quantization method for learning binary compact codes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 2938–2945.

[16] X. He, W.-Y. Ma, and H.-J. Zhang, "Learning an image manifold for retrieval," in *Proc. 12th Annu. ACM Int. Conf. Multimedia*, 2004, pp. 17–23.

[17] J.-P. Heo, Y. Lee, J. He, S.-F. Chang, and S.-E. Yoon, "Spherical hashing," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 2957–2964.

[18] R. Herbrich and R. C. Williamson, "Algorithmic luckiness," *J. Mach. Learn. Res.*, vol. 3, pp. 175–212, Sep. 2002.

[19] G. E. Hinton and S. T. Roweis, "Stochastic neighbor embedding," in *Advances in Neural Information Processing Systems*. Cambridge, MA, USA: MIT Press, 2002, pp. 833–840.

[20] G. Irie, Z. Li, X.-M. Wu, and S.-F. Chang, "Locally linear hashing for extracting non-linear manifolds," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 2123–2130.

[21] H. Jégou, M. Douze, C. Schmid, and P. Pérez, "Aggregating local descriptors into a compact image representation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2010, pp. 3304–3311.

[22] A. Joly and O. Buisson, "Random maximum margin hashing," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2011, pp. 873–880.

[23] W. Kong and W.-J. Li, "Isotropic hashing," in *Advances in Neural Information Processing Systems*. Red Hook, NY, USA: Curran & Associates Inc., 2012, pp. 1646–1654.

[24] B. Kulis and T. Darrell, "Learning to hash with binary reconstructive embeddings," in *Advances in Neural Information Processing Systems*. Red Hook, NY, USA: Curran & Associates Inc., 2009.

[25] B. Kulis and K. Grauman, "Kernelized locality-sensitive hashing for scalable image search," in *Proc. IEEE 12th Int. Conf. Comput. Vis.*, Sep./Oct. 2009, pp. 2130–2137.

[26] B. Kulis, P. Jain, and K. Grauman, "Fast similarity search for learned metrics," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 12, pp. 2143–2157, Dec. 2009.

[27] S. Lafon and A. B. Lee, "Diffusion maps and coarse-graining: A unified framework for dimensionality reduction, graph partitioning, and data set parameterization," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 9, pp. 1393–1403, Sep. 2006.

[28] X. Li, G. Lin, C. Shen, A. van den Hengel, and A. Dick, "Learning hash functions using column generation," in *Proc. 30th Int. Conf. Mach. Learn.*, 2013, pp. 142–150.

[29] G. Lin, C. Shen, Q. Shi, A. van den Hengel, and D. Suter, "Fast supervised hashing with decision trees for high-dimensional data," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 1971–1978.

[30] G. Lin, C. Shen, D. Suter, and A. van den Hengel, "A general two-step approach to learning-based hashing," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 2552–2559.

[31] J. Liu, Z. Huang, H. Cai, H. T. Shen, C. W. Ngo, and W. Wang, "Near-duplicate video retrieval: Current research and future trends," *ACM Comput. Surv.*, vol. 45, no. 4, 2013, Art. ID 44.

[32] W. Liu, C. Mu, S. Kumar, and S.-F. Chang, "Discrete graph hashing," in *Advances in Neural Information Processing Systems*. Red Hook, NY, USA: Curran & Associates Inc., 2014, pp. 3419–3427.

[33] W. Liu, J. Wang, R. Ji, Y.-G. Jiang, and S.-F. Chang, "Supervised hashing with kernels," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 2074–2081.

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

[35] W. Liu, J. Wang, Y. Mu, S. Kumar, and S.-F. Chang, "Compact hyperplane hashing with bilinear functions," in *Proc. 29th Int. Conf. Mach. Learn.*, 2012, pp. 17–24.

[36] Y. Liu, F. Wu, Y. Yang, Y. Zhuang, and A. G. Hauptmann, "Spline regression hashing for fast image search," *IEEE Trans. Image Process.*, vol. 21, no. 10, pp. 4480–4491, Oct. 2012.

[37] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*. New York, NY, USA: Cambridge Univ. Press, 2008.

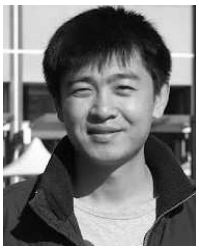
[38] M. Norouzi and D. J. Fleet, "Minimal loss hashing for compact binary codes," in *Proc. 28th Int. Conf. Mach. Learn.*, 2011, pp. 353–360.

[39] 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.

[40] M. Raginsky and S. Lazebnik, "Locality-sensitive binary codes from shift-invariant kernels," in *Advances in Neural Information Processing Systems*. Red Hook, NY, USA: Curran & Associates Inc., 2009.

[41] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, no. 5500, pp. 2323–2326, 2000.

[42] 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.
[43] J. Song, Y. Yang, Y. Yang, Z. Huang, and H. T. Shen, "Inter-media hashing for large-scale retrieval from heterogeneous data sources," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2013, pp. 785–796.
[44] A. Talwalkar, S. Kumar, and H. A. Rowley, "Large-scale manifold learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2008, pp. 1–8.
[45] J. B. Tenenbaum, V. de Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, no. 5500, pp. 2319–2323, 2000.
[46] A. Torralba, R. Fergus, and W. T. Freeman, "80 million tiny images: A large data set for nonparametric object and scene recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 11, pp. 1958–1970, Nov. 2008.
[47] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, pp. 2579–2605, Nov. 2008.
[48] J. Venna, J. Peltonen, K. Nybo, H. Aidos, and S. Kaski, "Information retrieval perspective to nonlinear dimensionality reduction for data visualization," *J. Mach. Learn. Res.*, vol. 11, pp. 451–490, Feb. 2010.
[49] J. Wang, S. Kumar, and S.-F. Chang, "Semi-supervised hashing for large-scale search," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 12, pp. 2393–2406, Dec. 2012.
[50] J. Wang, W. Liu, A. X. Sun, and Y.-G. Jiang, "Learning hash codes with listwise supervision," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 3032–3039.
[51] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong, "Locality-constrained linear coding for image classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2010, pp. 3360–3367.
[52] Y. Weiss, R. Fergus, and A. Torralba, "Multidimensional spectral hashing," in *Proc. 12th Eur. Conf. Comput. Vis.*, 2012, pp. 340–353.
[53] Y. Weiss, A. Torralba, and R. Fergus, "Spectral hashing," in *Advances in Neural Information Processing Systems*. Red Hook, NY, USA: Curran & Associates Inc., 2008.
[54] K. Yu, T. Zhang, and Y. Gong, "Nonlinear learning using local coordinate coding," in *Advances in Neural Information Processing Systems*. Red Hook, NY, USA: Curran & Associates Inc., 2009.
[55] D. Zhang, J. Wang, D. Cai, and J. Lu, "Self-taught hashing for fast similarity search," in *Proc. 33rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, 2010, pp. 18–25.
[56] X. Zhu, Z. Huang, H. Cheng, J. Cui, and H. T. Shen, "Sparse hashing for fast multimedia search," *ACM Trans. Inf. Syst.*, vol. 31, no. 2, 2013, Art. ID 9.



**Fumin Shen** received the B.S. degree from Shandong University, in 2007, and the Ph.D. degree from the Nanjing University of Science and Technology, China, in 2014. He is currently a Lecturer with the School of Computer Science and Engineering, University of Electronic of Science and Technology of China, China. His major research interests include computer vision and machine learning, including face recognition, image analysis, hashing methods, and robust statistics with its applications in computer vision.

**Chunhua Shen** is currently a Professor with the School of Computer Science, The University of Adelaide. His current research interests include intersection of computer vision and statistical machine learning. He received the Australian Research Council Future Fellowship in 2012. He is an Associate Editor of the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS.

**Qinfeng Shi** received the bachelor's and master's degrees in computer science and technology from Northwestern Polytechnical University, in 2003 and 2006, respectively, and the Ph.D. degree in computer science from Australian National University, in 2011. He is currently a Senior Lecturer with the Australian Centre for Visual Technologies and the School of Computer Science, The University of Adelaide.

**Anton van den Hengel** received the B.Sc. and L.L.B. degrees, the master's degree in computer science, and the Ph.D. degree in computer vision from The University of Adelaide, in 1991, 1993, 1994, and 2000, respectively. He is currently the Founding Director of the Australian Centre for Visual Technologies. He is currently a Professor with the School of Computer Science, The University of Adelaide.

**Zhenmin Tang** received the Ph.D. degree from the Nanjing University of Science and Technology, Nanjing, China. He is currently a Professor and the Head of the School of Computer Science with the Nanjing University of Science and Technology. He has authored over 80 papers. His major research area includes intelligent system, pattern recognition, image processing, and embedded system. He is also the Leader of several key programs of the National Nature Science Foundation of China.

**Heng Tao Shen** received the B.Sc. (Hons.) and Ph.D. degrees from the Department of Computer Science, National University of Singapore, in 2000 and 2004, respectively. He joined The University of Queensland as a Lecturer, Senior Lecturer, and Reader, where he became a Professor in 2011. He is currently a Professor of Computer Science and an ARC Future Fellow with the School of Information Technology and Electrical Engineering, The University of Queensland. He is a Visiting Professor with Nagoya University and the National University of Singapore. His research interests mainly include multimedia/mobile/web search, and big data management on spatial, temporal, multimedia, and social media databases. He received the Chris Wallace Award for outstanding research contribution by the Computing Research and Education Association, Australasia, in 2010. He has extensively published and served on program committees in most prestigious international publication venues of interests. He is an Associate Editor of the IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, and a PC Co-Chair of ACM Multimedia 2015.