Asymmetric Binary Coding for Image Search

Fumin Shen, Yang Yang, Li Liu, Wei Liu, Dacheng Tao and Heng Tao Shen

***Abstract*—Learning to hash has attracted broad research interests in recent computer vision and machine learning studies, due to its ability to accomplish efficient approximate nearest neighbor (ANN) search. However, the closely related task, Maximum Inner Product Search (MIPS), has rarely been studied in this literature. To facilitate the MIPS study, in this work, we introduce a general binary coding framework based on asymmetric hash functions, named Asymmetric Inner-product Binary Coding (AIBC). In particular, AIBC learns two different hash functions which can reveal the inner products between original data vectors by the generated binary vectors. Although conceptually simple, the associated optimization is very challenging due to the highly nonsmooth nature of the objective that involves sign functions. We tackle the nonsmooth optimization in an alternating manner, by which each single coding function is optimized in an efficient discrete manner. We also simplify the objective by discarding the quadratic regularization term which significantly boosts the learning efficiency. Both problems are optimized in an effective discrete way without continuous relaxations, which produces high-quality hash codes. In addition, we extend the AIBC approach to the supervised hashing scenario, where the inner products of learned binary codes are forced to fit the supervised similarities. Extensive experiments on several benchmark image retrieval databases validate the superiority of the AIBC approaches over many recently proposed hashing algorithms.**

I. Introduction

In recent years, binary coding (also known as hashing) has become a very popular research subject in computer vision [19][43][46][21][13][36][22][53][16][50][26] and multimedia processing [48][42][41][23][40][39][15]. After encoding high-dimensional feature vectors of documents, images, or videos to compact binary codes, an effective binary coding or hashing method is expected to accomplish efficient similarity search while preserving the similarities among original data to some extent. As a result, using binary codes to represent and search in massive data is a promising solution to handle large-scale vision tasks, because of not only the storage efficiency (typically several hundred binary bits per data item) but also the high time efficiency of pairwise distance computations in the Hamming space.

This work was supported in part by the National Natural Science Foundation of China under Project 61502081, Project 61572108, Project 61673299 and Project 61632007, in part by the Fundamental Research Funds for the Central Universities under Project ZYGX2014Z007.

F. Shen, Y. Yang and H. T. Shen are with Center for Future Media and School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China (E-mail: fumin.shen@gmail.com, dlyyang@gmail.com,shenhengtao@hotmail.com). Corresponding author: Heng Tao Shen.

L. Liu is with Malong Technologies Co., Ltd. (E-mail: li.liu@malongtech.cn).

W. Liu is with Tencent AI Lab, Shenzhen 518057, China (E-mail: wliu@ee.columbia.edu).

D. Tao is with School of Information Technologies, The University of Sydney, Australia (E-mail: dacheng.tao@sydney.edu.au).

Almost all the previous binary coding and hashing algorithms were designed to deal with Approximate Nearest Neighbor (ANN) search. Studies have rarely been dedicated to Maximum Inner Product Search (MIPS), which actually plays a critical role in various vision and learning applications [37]. The efficacy of aforementioned methods have not been validated on the MIPS problem yet. Shrivastava and Li [37] proposed an Asymmetric Locality-Sensitive Hashing (ALSH) algorithm for searching with (un-normalized) inner product being the underlying similarity measure. ALSH converts the MIPS problem to a standard LSH ANN problem by performing a simple asymmetric transformations on data pairs. While ALSH inherits all the theoretical guarantees of LSH [37], it can hardly achieve promising search performance using short binary codes due to its independence of data.

In this work, we focus on learning data-dependent binary codes for tackling the MIPS problem. Inspired by the latest advance in asymmetric hashing [37], we propose an asymmetric binary code learning method for MIPS, thus named **Asymmetric Inner-product Binary Coding (AIBC)**. Specifically, two sets of coding functions are learned such that the inner products between their generated binary codes can reveal the inner products between original data vectors. Despite conceptually simple, the associated optimization is very challenging due to the highly nonsmooth nature of the objective that involves sign functions. To this end, we tackle the nonsmooth optimization in an alternating manner, by which a single coding function is solved with the others fixed. Through introducing auxiliary discrete variables to replace the sign functions, the optimization procedure is made efficient. As a simplified version of the proposed binary code learning framework, we propose another objective which maximizes the correlations between the inner products of the produced binary codes and raw data vectors. In both objectives, the binary codes and coding functions are simultaneously learned without continuous relaxations, which is the key to achieve high-quality binary codes.

The main contributions of our work are summarized as follows:

1) We propose a binary code learning framework for addressing the MIPS problem, which has the clear aim of preserving the inner-product similarities among raw data vectors. To this end, we design a tractable discrete optimization method, by which high-quality binary codes are iteratively generated with a closed-form solution for each bit.
2) To further speed up binary code learning for MIPS, we propose an alternative simpler objective which maximizes the correlations between the inner products of the yielded binary codes and raw data vectors. By this objective, the binary codes can be learned in a much

more efficient way and, usually, with higher quality. This is mainly because the problem can be easily solved with closed-form solutions for the two associated sub-problems.

3) Based on the AIBC framework, we propose an asymmetric supervised hashing algorithm (ASH), which learns two sets of binary codes to fit the supervised similarities instead of the inner products of original data. The effectiveness of asymmetric hashing is further validated by the supervised hashing method.
4) Our AIBC methods (with two unsupervised objectives) together with the supervised ASH are extensively evaluated on several large-scale image datasets. The experimental results demonstrate the superiority of our methods over the state-of-the-arts.

The rest of this paper is organized as follows. Section II briefly reviews the related works. Section IV-B elaborates the details of the proposed AIBC methods. In Section V, we evaluate our approaches on three real-world large-scale datasets, followed by the conclusion of this work in Section VI.

This paper is an extended version of the work previously published in [32]. The major improvements in this work over [32] include the refinement of Abstract, Introduction (Section I) and Related work (Section II); We extend the original work to a general binary coding framework for maximum inner product search, which is studied by both the unsupervised AIBC approach and the newly introduced asymmetric supervised hashing algorithm (Section IV-D); More experimental results with comparison to additional methods are provided (Section V).

II. Related Work

In this section, we briefly review related works in the hashing literature. The binary coding or hashing techniques can be roughly divided into two major categories: unsupervised and supervised methods.

A. Unsupervised hashing

Locality-Sensitive Hashing (LSH) [6] is one of the most popular data-independent unsupervised hashing methods, which generates randomized hash functions via random projections. The LSH family has been continuously developed to accommodate a variety of distance and similarity measures, Although LSH is ensured to have high collision probability for similar data items, in practice LSH usually needs long hash bits and multiple hash tables to achieve both high precision and recall. The huge storage overhead may restrict its applications.

Recent years has witnessed a rapid development of the data-dependent methods (or *learning-based* methods). The literature was comprehensive reviewed in [44] recently. Its emergence is due to the benefit that *learned* compact binary codes can effectively and efficiently index and organize massive data. Different from LSH, data-dependent binary coding methods aim to generate short binary codes using available training data. Among the learning based hashing algorithms, linear coding functions formed by a set of learned hyperplanes are mostly adopted, since they are computationally simple and easy to connect to traditional dimensionality reduction techniques. A number of algorithms in this category have been proposed, including unsupervised PCA Hashing [43], Iterative Quantization (ITQ) [7], Isotropic Hashing (IsoHash) [9], *etc*.

It has also been shown that harnessing nonlinear manifold structures will help produce neighborhood-preserving compact binary codes. Spectral Hashing (SH) [46][45] and Kernelized LSH [11] are well-known algorithms in this style. More recently, Anchor Graph Hashing (AGH) [20][18] leverages anchor graphs for making hash code training and out-of-sample extension to novel data both tractable and efficient. Shen *et al*. [35] proposed a general Inductive Manifold Hashing (IMH) scheme which generates nonlinear coding functions by exploiting the flexibility of available manifold learning approaches.

B. Supervised hashing

By taking advantages of the manually-labeled data, the hashing performance has been significantly improved by the supervised methods. Some representative supervised hashing algorithms are supervised Minimal Loss Hashing (MLH) [30], Semi-Supervised Hashing (SSH)[43], FastHash [14], *etc*. It has also been studied to construct coding functions in a kernel space, for example, Binary Reconstructive Embedding (BRE) [10], Kernel-Based Supervised Hashing (KSH) [19], the kernel variant of ITQ [7] and Supervised Discrete Hashing (SDH) [33]. The kernel based hashing algorithms are shown to perform better than the linear based hashing ones in some applications.

Recently, a few deep learning based hashing algorithms had been proposed, where the features learned from deep neural networks were shown to achieve better performance than the hand-crafted ones [5][24][12][17]. For these deep hashing approaches, there might be concern about the binary coding time, since predicting a novel test sample needs to go through the time-consuming deep neural networks.

III. Background: Maximum Inner Product Search

We give a brief introduction of Maximum Inner Product Search (MIPS). MIPS has been playing a significant role in various applications, such as recommender systems, deformable part model, multi-class classification [37]. Given a new query $\mathbf{q}$, MIPS targets at retrieving the datum having the largest inner product with $\mathbf{q}$ from the database $\mathbf{A}$. Formally, the MIPS problem is formulated as below:

$$\mathbf{p} = \arg\max_{\mathbf{a}\in\mathbf{A}} \ \mathbf{a}^\top \mathbf{q}. \tag{1}$$

For large-scale similarity search problems, it is practical to implement MIPS by employing binary coding techniques to achieve both storage and computational efficiencies. A binary coding function $h(\mathbf{x})$ maps an original feature vector $\mathbf{x}$ to a binary code of r bits $\mathbf{b} \in \{1, -1\}^r$ in the Hamming space[1]. Then problem (1) is reformulated as follows

$$\mathbf{p} = \arg\max_{\mathbf{a}\in\mathbf{A}} \ h(\mathbf{a})^\top h(\mathbf{q}). \tag{2}$$

[1]Note that we use (1,-1) bits for mathematical derivations, and use (1,0) bits for implementations of all referred binary coding and hashing algorithms.

It is natural to apply the popular locality sensitive hashing (LSH) for this problem. The LSH algorithm is simply constructed by the random projections generated from the standard normal distribution. Despite the popularity, the direct use of LSH on MIPS does not inherit the high collision probability guarantee of that on near neighbor search problems. To solve this problem, Shrivastava and Li [37] proposed the Asymmetric Locality Sensitive Hashing (ALSH) method, which converted the MIPS problem to the standard ℓ_2 nearest neighbor search problem. ALSH adopts two sets of *different* hash functions $h(\cdot)$ and $z(\cdot)$ to compute the inner product of the query and database point. The MIPS problem is performed as

$$\mathbf{p} = \arg\max_{\mathbf{a}\in\mathbf{A}} \ h(\mathbf{a})^\top z(\mathbf{q}), \tag{3}$$

where $h(\cdot)$ and $z(\cdot)$ are both constructed directly from LSH, by simply appending a few entries to $\mathbf{a}$ and $\mathbf{q}$ with different values. ALSH was proved to share the similar collision guarantee of similar points on MIPS as LSH on nearest neighbor search problem, which is mainly benefited from the flexibility of asymmetric hash functions. An asymmetric LSH algorithm with sketches [4] was previously proposed at a early stage. Recently, [29][28][27] discussed the effectiveness of asymmetric hashes.

IV. A General Binary Coding Framework for Maximum Inner Product Search

In this section, we present our binary code learning framework for maximum inner product search (MIPS). and then propose an inner-product fitting model to learn two asymmetric coding functions. To speed up the optimization, we practically simplify the original objective by maximizing inner-product correlation.

A. The binary coding framework

While the LSH and ALSH methods guarantee search accuracy to some extent, promising performance can hardly be achieved with compact binary codes due to the independence of data. To overcome the above limitation, we propose to *learn* hash functions rather than random projection for coping with the MIPS task.

Suppose that we have two sets of points: $\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_n]$ and $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m]$, where $\mathbf{a}_i \in \mathbb{R}^{d\times 1}$ and $\mathbf{x}_i \in \mathbb{R}^{d\times 1}$. Denote the similarity matrix of $\mathbf{A}$ and $\mathbf{X}$ as $\mathbf{S} \in \mathbb{R}^{n\times m}$, of which the element $\mathbf{S}_{ij}$ defines the similarity of $\mathbf{a}_i$ and $\mathbf{x}_j$. We aim to learn two sets of binary codes for $\mathbf{A}$ and $\mathbf{X}$ respectively, the inner product of which can well approximate $\mathbf{S}$. Inspired by [37], we adopt the asymmetric form of hash functions in our binary code learning formulation. That is, we consider the following problem:

$$\min_{h,z} \sum_{i=1}^{n}\sum_{j=1}^{m} \ell(h(\mathbf{a}_i)^\top z(\mathbf{x}_j), \mathbf{S}_{ij}). \tag{4}$$

Here $\ell(\cdot,\cdot)$ can be any proper loss function that forces the two terms to be close, which we will discuss later.

Hash function The hash functions $h(\cdot)$ and $z(\cdot)$ can be a linear, non-linear (*i.e.*, kernel-based) or even a deep neural networks based model. In this work, we focus on the simple linear hash function, *i.e.*, the hash functions write

$$h(\mathbf{a}) = \mathrm{sgn}(\mathbf{W}^\top \mathbf{a}) \tag{5}$$

$$z(\mathbf{x}) = \mathrm{sgn}(\mathbf{R}^\top \mathbf{x}) \tag{6}$$

where $\mathbf{W} \in \mathbb{R}^{d\times r}$, $\mathbf{R} \in \mathbb{R}^{d\times r}$ is the projection matrix.

We name the proposed asymmetric binary coding methods for maximum inner-product search as **Asymmetric Inner-product Binary Coding (AIBC)**.

B. Unsupervised: Inner product fitting

A natural approach to model the similarity of two points is through their inner product, *i.e.*, $\mathbf{S}_{ij} = \mathbf{a}_i^\top \mathbf{x}_j$. In practice, to better fit with binary codes, we normalize $\mathbf{S}_{ij}$ to be r (or 0) by a threshold.

We consider the following problem with hash functions $h(\cdot)$ and $z(\cdot)$,

$$\min_{h,z} ||h(\mathbf{A})^\top z(\mathbf{X}) - \mathbf{S}||^2. \tag{7}$$

Here $||\cdot||$ is the Frobenius norm. Problem (7) turns out to be a problem of *inner product fitting* with binary codes. We will show next that the utilization of asymmetric hash functions can significantly facilitate the optimization of the above discrete optimization problem.

We then have

$$\min_{\mathbf{W},\mathbf{R}} ||\mathrm{sgn}(\mathbf{W}^\top \mathbf{A})^\top \mathrm{sgn}(\mathbf{R}^\top \mathbf{X}) - \mathbf{S}||^2. \tag{8}$$

It is easy to see that the above problem is highly non-convex and difficult (usually NP hard) to solve due to the discrete sign functions. A feasible solution is to relax the discrete constraint by omitting the sign function. The continuous relaxation methodology is widely applied for hash code learning. However, this approximation method may accumulate considerable quantization errors, which makes the final binary codes less effective.

In order to obtain high-quality hash functions, we keep the sign function in our formulation. By taking the advantage of the flexibility of asymmetric hash functions, for this discrete optimization problem, we can naturally choose to solve $\mathbf{W}$ and $\mathbf{R}$ in an alternating fashion, that is, solve for one variable each time while keeping the other one fixed. We thus first consider the following sub-problem with variable $\mathbf{W}$ by fixing $z(\mathbf{X}) = \mathbf{Z}$ in (7),

$$\min_{\mathbf{W}} ||\mathrm{sgn}(\mathbf{W}^\top \mathbf{A})^\top \mathbf{Z} - \mathbf{S}||^2. \tag{9}$$

In practice, we initialize $\mathbf{R}$ by PCA projections. Similarly, by fixing $h(\mathbf{A}) = \mathbf{H}$, we obtain the following sub-problem with variable $\mathbf{R}$,

$$\min_{\mathbf{R}} ||\mathbf{H}^\top \mathrm{sgn}(\mathbf{R}^\top \mathbf{X}) - \mathbf{S}||^2, \tag{10}$$

Problem (9) or (10) is still with the sign function and cannot be solved trivially by an off-the-shelf solver. We now present a scalable and tractable method to (9), and (10) can be accordingly solved in the same way. To conquer the optimization involving discrete function, we introduce an auxiliary variable $\mathbf{B} \in \{-1, 1\}^{r\times n}$ to separate the optimizing

variable $\mathbf{W}$ and the sign function. We will show next the introduction of the auxiliary variable $\mathbf{B}$ (binary codes for $\mathbf{A}$) is the key to simplify the optimization. We rewrite problem (9) as

$$\begin{aligned} &\min_{\mathbf{B},\mathbf{W}} ||\mathbf{B}^\top\mathbf{Z} - \mathbf{S}||^2 + \lambda||\mathbf{B} - \mathbf{W}^\top\mathbf{A}||^2 \\ &\text{s.t.} \quad \mathbf{B} \in \{-1, 1\}^{r\times n}. \end{aligned} \tag{11}$$

The objective of (11) has a clear explanation: the first term minimizes the inner products fitting error by the learned binary codes; while the second term ensures the hash function $h(\mathbf{x})$ can well predict the target binary codes $\mathbf{B}$ with minimum quantization loss. The parameter λ serve a trade-off between these two loss terms.

In problem (11), given $\mathbf{B}$, it is easy to compute $\mathbf{W}$

$$\mathbf{W} = (\mathbf{A}\mathbf{A}^\top)^{-1}\mathbf{A}\mathbf{B}^\top. \tag{12}$$

We then have the following problem w.r.t. $\mathbf{B}$,

$$\begin{aligned} &\min_{\mathbf{B}} ||\mathbf{B}^\top\mathbf{Z}||^2 - 2\text{trace}(\mathbf{B}^\top\mathbf{D}) \\ &\text{s.t.} \quad \mathbf{B} \in \{-1, 1\}^{r\times n}, \end{aligned} \tag{13}$$

where $\mathbf{D} = \mathbf{Z}\mathbf{S}^\top + \lambda\mathbf{W}^\top\mathbf{A}$. Note that $||\mathbf{B}||^2 = nr$.

For this key sub-problem, inspired by the recent a *discrete cyclic coordinate descent (DCC)* method [33], we optimize one row of $\mathbf{B}$ each time while fixing all other rows, *i.e.*, we compute one-bit for all n samples each time. Let $\mathbf{b}$ be the l^{th} row of $\mathbf{B}$, $l = 1, \cdots, r$, and $\tilde{\mathbf{B}}$ the remaining rows of $\mathbf{B}$. Then $\mathbf{b}$ contains one bit for each of n samples. Similarly, let $\mathbf{d}$ be the l^{th} row of $\mathbf{D}$, $\tilde{\mathbf{D}}$ the matrix of $\mathbf{D}$ excluding $\mathbf{d}$, $\mathbf{z}$ the l^{th} row of $\mathbf{Z}$ and $\tilde{\mathbf{Z}}$ the matrix of $\mathbf{Z}$ excluding $\mathbf{z}$. With these notations and a few simple matrix manipulations, problem (13) can be written as w.r.t. $\mathbf{b}$

$$\begin{aligned} &\min_{\mathbf{b}} \mathbf{b}(\tilde{\mathbf{B}}^\top\tilde{\mathbf{Z}}\mathbf{z}^\top - \mathbf{d}^\top) \\ &\text{s.t.} \quad \mathbf{b} \in \{-1, 1\}^n. \end{aligned} \tag{14}$$

Thus, we obtain the optimal solution for the l^{th} row of $\mathbf{B}$,

$$\mathbf{b} = \text{sgn}(\mathbf{z}\tilde{\mathbf{Z}}^\top\tilde{\mathbf{B}} - \mathbf{d}). \tag{15}$$

By this method, each bit is iteratively updated with the pre-learned $r-1$ bits till the procedure converges with a set of better codes. In our experiments, the procedure usually converges with less than 10 iterations. With the analytical solution for each bit, the whole optimization is very efficient and thus can easily scale to massive data. Till now, we are ready to solve problem (11) (therefor also (9)) iteratively with the solution of $\mathbf{W}$ and $\mathbf{B}$ provided above. By iteratively solve (9) and (10), we finally obtain a pair of hash function of $h(\cdot)$ and $z(\cdot)$. Although this method can hardly achieve the globally optimal solution for the discrete optimization problem, at each step, the local optimal solution for each variable (*e.g.*, $\mathbf{W}, \mathbf{b}$) is obtained in a closed form. We will show in the experiments that this optimization strategy works very well. Since the objective (7) of this method involves a quadratic term with the hash functions, we denote this method as AIBC-Q.

C. Unsupervised: Inner product correlation maximization

In this part, we propose an alternative model to further speed up the learning procedure of AIBC. We optimize the following problem:

$$\max_{h,z} \text{trace}(h(\mathbf{A})\mathbf{S}z(\mathbf{X})^\top). \tag{16}$$

Problem (16) has a clear intuition: it maximizes the correlation between the similarity matrix $\mathbf{S}$ and the inner products between learned binary codes. Note that the objective of (16) can be easily obtained by discarding the quadratic term $||h(\mathbf{A})^\top z(\mathbf{X})||^2$ in (4). However, the quadratic term does not leverage the groundtruth similarity and can be seen as a regularization with the magnitude of the learned inner product. We show next it provides a much more efficient mean of solving the asymmetric hash function learning problem.

Using the same strategy described in the previous section, for (16) we obtain the following two sub-problems (17) and (18) w.r.t. $\mathbf{W}$ and $\mathbf{R}$, respectively.

$$\max_{\mathbf{W}} \text{trace}(\text{sgn}(\mathbf{W}^\top\mathbf{A})\mathbf{S}\mathbf{Z}^\top), \tag{17}$$

$$\max_{\mathbf{R}} \text{trace}(\mathbf{H}\mathbf{S}\text{sgn}(\mathbf{R}^\top\mathbf{X})^\top). \tag{18}$$

By introducing the auxiliary variable $\mathbf{B}$, problem (17) is formulated as

$$\begin{aligned} &\max_{\mathbf{B},\mathbf{W}} \text{trace}(\mathbf{B}\mathbf{S}\mathbf{Z}^\top) - \lambda||\mathbf{B} - \mathbf{W}^\top\mathbf{A}||^2 \\ &\text{s.t.} \quad \mathbf{B} \in \{-1, 1\}^{r\times n}. \end{aligned} \tag{19}$$

Same as problem (11), we solve problem (19) iteratively with $\mathbf{B}$ and $\mathbf{W}$. The difference is, problem (19) benefits from the advantage that it has a optimal analytical solution for $\mathbf{B}$ with a given $\mathbf{W}$,

$$\mathbf{B} = \text{sgn}\big(\mathbf{Z}\mathbf{S}^\top + 2\lambda\mathbf{W}^\top\mathbf{A}\big). \tag{20}$$

Compared to problem (4)*, optimizing* (16) *does not only make* (19) *much more efficient to solve but also provide more accurate solution for the whole optimization.* This explains why this method performs slightly better than AIBC-Q in our experiments. The optimization of (19) can be easily solved by iteratively updating $\mathbf{B}$ and $\mathbf{W}$ (by (12)). With the above closed-form solutions, the training of the proposed method can be easily performed on large-scale data with very high efficiency. Since the objective (16) of this method only involves a linear term with each hash function, we denote this method as AIBC-L.

D. Supervised: Supervised asymmetric hashing

In this section, we extend our asymmetric inner-product binary coding (AIBC) approach to cope with the supervised hashing problem. In the supervised scenario, a simple modification of AIBC is to compute the similarity matrix (denoted as $\mathbf{S}^l$) by the provided semantic labels. That is, we compute $\mathbf{S}^l$ by

$$\mathbf{S}^l_{ij} = \begin{cases} r, & \text{if } \mathbf{a}_i \text{ and } \mathbf{x}_j \text{ share the same label(s)}, \\ 0, & \text{otherwise.} \end{cases} \tag{21}$$

Algorithm 1 Asymmetric Inner-product Binary Coding (AIBC)

Input: Data $\mathbf{A}$ and $\mathbf{X}$; code length r; maximum iteration number t; parameters λ.

Output: Hash function $h(\mathbf{x})$ and $z(\mathbf{x})$.

1) Compute similarity matrix $\mathbf{S}$ by $\mathbf{A}^\top\mathbf{X}$ for unsupervised hashing or by Eq. (21) for supervised hashing.
2) Initialize $\mathbf{R}$ by PCA projections.
3) Loop until converge or reach maximum t iterations:
   - h-step: Compute $\mathbf{W}$ by solving problem (9) or (17).
   - z-step: Compute $\mathbf{R}$ by solving problem (10) or (18).

Particularly, for multiple labelled data, $\mathbf{S}_{ij} = r$ if $\mathbf{a}_i$ and $\mathbf{x}_j$ share at least one labels. Similar settings has been used in many supervised hashing methods [19], [43]. With the ℓ_2 loss as in (7), we write our supervised hashing as

$$\min_{h,z} ||h(\mathbf{A})^\top z(\mathbf{X}) - \mathbf{S}^l||^2. \tag{22}$$

Similarly, with the linear loss as in (16), we have

$$\max_{h,z} \operatorname{trace}(h(\mathbf{A})\mathbf{S}^l z(\mathbf{X})^\top). \tag{23}$$

Both the above two problems can be solved exactly the same as the unsupervised algorithm as discussed before. For efficiency, in practice we adopt the supervised hashing method in (23). The optimization follows the same procedure in Section IV-C with PCA as initialization. For the supervised hashing method, other supervised dimensionality reduction algorithms (*e.g.*, LDA) can potentially improve the performance, which is however not the focus of this work. We denote this method as asymmetric supervised hashing (ASH) in the following text.

Connection to KSH The kernel-based supervised hashing (KSH) [19] uses a similar objective as in (22):

$$\min_{h} ||h(\mathbf{A})h(\mathbf{A})^\top - \mathbf{S}^l||^2. \tag{24}$$

However, our proposed AIBC mainly differs from KSH in the following aspects: 1) KSH learns a single coding function while AIBC learns two coding functions in an asymmetric fashion. 2) KSH applies a greedy optimization procedure to solve a relaxed problem by using a sigmoid function to replace the sign function. In contrast, AIBC directly optimizes the binary codes without resorting to any continuous relaxations. In the experiments, we compare our methods with KSH and its unsupervised version (implemented by ourselves) and the comparative results clearly show the advantage of our AIBC techniques. We summarize the proposed Asymmetric Inner-product Binary Coding (AIBC) algorithm in Algorithm 1.

V. Experiments

In this section, we first compare the proposed AIBC to the state-of-the-arts in both supervised and unsupervised scenarios to validate its effectiveness. The AIBC model is then evaluated in terms of time efficiency, training convergence and parameter sensitiveness.

TABLE I: Results in terms of mAP and Precision of top 500 samples of the compared methods on **SUN397** with 32, 64 and 128 bits, respectively.

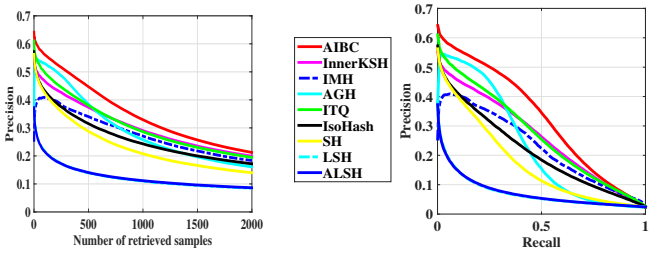
Method	mAP			Precision@500		
	32-bit	64-bit	128-bit	32-bit	64-bit	128-bit
LSH	0.0605	0.0920	0.1428	0.0921	0.1397	0.2036
ALSH	0.0592	0.0913	0.1425	0.0918	0.1406	0.2033
SH	0.2103	0.2191	0.1991	0.2737	0.2877	0.2735
IsoHash	0.2414	0.2668	0.2850	0.2912	0.3198	0.3391
ITQ	0.3014	0.3336	0.3456	0.3448	0.3774	0.3887
AGH	0.3629	0.3026	0.2383	**0.4238**	0.3806	0.3353
IMH	0.3043	0.3374	0.3631	0.3363	0.3690	0.3925
InnerKSH	0.2900	0.3312	0.3558	0.3367	0.3734	0.3923
AIBC-Q	0.3549	0.3925	0.4299	0.3996	0.4263	0.4409
AIBC-L	**0.3639**	**0.4042**	**0.4348**	0.4078	**0.4428**	**0.4642**

A. Evaluation on unsupervised hashing

In this section, we evaluate the proposed two unsupervised hashing algorihtms (*i.e.*, AIBC-Q and AIBC-L) by comparing them to several representative unuspervised hashing methods in the literature. Three large-scale datasets: YouTube Faces [47], SUN397 [49] and ImageNet [3] are used. The compared unsupervised hashing methods including LSH (implemented by signed random projections), ALSH [37], SH [46], ITQ [7], IsoHash [9], AGH [20] and IMH [34]. We also compare the unsupervised version of KSH [19] implemented by ourselves with the similarity computed by inner product (thus named InnerKSH). We use the public codes and suggested parameters of these methods from the authors. For InnerKSH, we use 4,000 training samples to form the pairwise similarity matrix and randomly choose 1,000 samples as anchor points. Since our methods are unsupervised, we do not compare them with the supervised methods. For our AIBC, we empirically set the parameter λ to 100 and the maximum iteration number $t = 2$; the data matrix $\mathbf{A}$ is set as the whole training data; $\mathbf{X}$ is formed by the 10,000 randomly selected points from training data. We binarize each column of $\mathbf{S}$ with a threshold computed by the k_{th} largest one among the n inner products, where k is set to 500 for SUN397, 1000 for ImageNet and 2000 for YouTube Faces. For evaluation, database data and queries are compressed by $h(\mathbf{x})$ and $z(\mathbf{x})$, respectively.

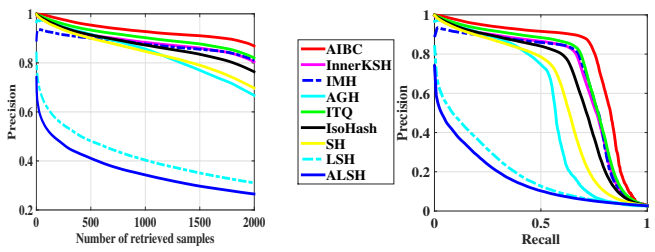
We report the compared results in terms of both hash lookup: precision of Hamming distance 2 (HD2 Precision) and Hamming ranking: mean of average precision (mAP) and mean precision of the top 500 retrieved neighbors (Precision@500). We also present the detailed results by precision-recall and the precision of top 2000 curves. Note that we treat a query as a false case if no point is returned when calculating precisions. Ground truths are defined by the category information from the datasets.

1) SUN397: retrieval with scene images: SUN397 [49] contains about 108K images from 397 scene categories, where each image is represented by a 1,600-dimensional feature vector extracted by PCA from 12,288-dimensional Deep Convolutional Activation Features [8]. We use a subset of this dataset including 42 categories with each containing more than



(a) Precision (b) Precision-recall

Fig. 1: Precision curves of up to top 2000 retrieved samples and precision-recall curves on SUN397. We only report AIBC-L for AIBC for clarity. 64 bits are used.



(a) Precision (b) Precision-recall

Fig. 2: Precision curves of up to top 2000 retrieved samples and precision-recall curves on Youtube Faces. We only report AIBC-L for AIBC for clarity. 64 bits are used.

TABLE II: Results in terms of mAP, Precision of top 500 samples of the compared methods on the **YouTube Faces** database with 32, 64 and 128 bits, respectively.

Method	mAP			Precision@500		
	32-bit	64-bit	128-bit	32-bit	64-bit	128-bit
LSH	0.1341	0.2513	0.4092	0.2710	0.4811	0.7004
ALSH	0.1124	0.2104	0.3565	0.2210	0.4106	0.6355
SH	0.6543	0.6395	0.5677	0.8556	0.9001	0.9047
IsoHash	0.6756	0.7204	0.7274	0.8698	0.9150	0.9274
ITQ	0.7443	0.7775	0.7767	0.8979	0.9321	0.9416
AGH	0.7054	0.6236	0.4610	0.8843	0.9233	0.9171
IMH	0.7467	0.7916	0.7916	0.8600	0.9228	0.9343
InnerKSH	0.7512	0.7634	0.7757	0.8989	0.9138	0.9239
AIBC-Q	0.7913	0.8175	0.8293	0.9393	0.9502	**0.9625**
AIBC-L	**0.8152**	**0.8233**	**0.8400**	**0.9477**	**0.9551**	0.9590

TABLE III: Results in terms of mAP and Precision of top 500 samples of the compared methods on the **ImageNet** database with 32, 64 and 128 bits, respectively.

Method	mAP			Precision@500		
	32-bit	64-bit	128-bit	32-bit	64-bit	128-bit
LSH	0.0496	0.0974	0.1743	0.1036	0.1963	0.3133
ALSH	0.0495	0.0907	0.1694	0.1043	0.1847	0.3074
SH	0.2418	0.3066	0.3309	0.3647	0.4531	0.4956
IsoHash	0.2521	0.3326	0.3847	0.3673	0.4649	0.5231
ITQ	0.3231	0.4127	0.4621	0.4304	0.5313	0.5882
AGH	0.4545	0.5095	0.4541	0.5346	0.6086	0.6036
IMH	0.2805	0.3554	0.3985	0.3197	0.4453	0.4998
InnerKSH	0.4073	0.4651	0.4850	0.5080	0.5693	0.5893
AIBC-Q	0.4421	0.5280	**0.5756**	0.5702	0.6180	**0.6658**
AIBC-L	**0.4771**	**0.5402**	0.5753	**0.5796**	**0.6375**	0.6643

500 images (with total 33K images); 100 images are sampled uniformly randomly from each category to form a test set of 4,200 images.

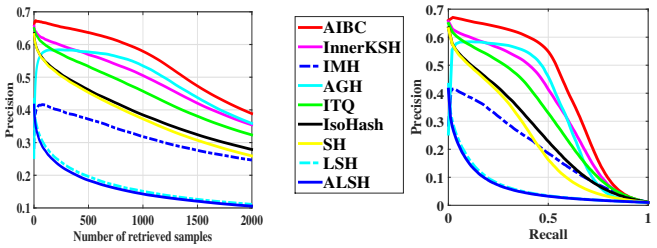
The comparative results are shown in Table I. First we can see that the proposed AIBC-L significantly outperforms all other algorithms in mAP with all code lengths. In terms of precision500, AGH achieves the best result at 32-bit. However, we observe that it suffers from a severe performance drop with long code length for both MAP and precision. For example, AGH obtains much lower results than ITQ, IMH and AIBC at 128-bit. Our AIBC-Q performs slight worse than AIBC-L on this dataset, although it still outperforms all other methods in most situations. It is not surprising that the data-independent algorithm LSH and ALSH do not perform as well as other learning based methods. Interestingly we also observe that the asymmetric ALSH algorithm achieves close results with the original signed random projection based LSH method on this dataset. The detailed precision curves of up to top 2000 retrieved samples and the precision-recall curves using 64 bits are shown in Figure 1. We can easily see that the performance rank of the precision curves of the compared methods is consistent with the above analysis. The proposed AIBC still performs the best among the compared algorithms.

2) YouTube Faces: retrieval with face images: YouTube Faces dataset contains 1,595 different people, from which we choose 340 people such that each one has at least 500 images to form a subset of 370,319 face images, and represent each face image as a 1,770-dimensional LBP feature vector [1]. We use a subset of YouTube Faces with 38 people each containing more than 2,000 faces (about 100K images in total). The test set includes 3,800 face images which are evenly sampled from each of the 38 classes.

We report the results on YouTube Faces in Table II. Again, the proposed AIBC-L and AIBC-Q achieve the best results on this dataset in all situations. For instance, with 64 bits the proposed AIBC-L obtains 82.33% MAP which is higher than the second best 79.16% (by IMH) by 3.17%. SH and IsoHash do not perform as well as other data-dependent methods on this dataset. The proposed AIBC is further compared to these methods in precision of top 2000 samples and precision-recall. As can be clearly observed in Figure 2, AIBC ranks the first on both of the two evaluation curves.

3) ImageNet: retrieval with large dataset : As a subset of ImageNet [3], the large dataset ILSVRC 2012 contains over 1.2 million images of totally 1,000 categories. We form the retrieval database by the 100 largest classes with total 128K images from the provided training set, and 50,000 images from the validation set as the query set. As in [14], we use the 4096-dimensional features extracted by the convolution neural networks (CNN) model.

The results are shown in Table III. The superiority of the AIBC is further demonstrated on this large-scale database. For



(a) Precision (b) Precision-recall

Fig. 3: Precision curves of up to top 2000 retrieved samples and precision-recall curves on ImageNet. We only report AIBC-L for AIBC for clarity. 64 bits are used.

TABLE IV: Comparison in HD2 precision with code length 32.

Method	SUN 397	Youtube Faces	ImageNet
LSH	0.0304	0.4444	0.0708
ALSH	0.0352	0.3262	0.0644
SH	0.2479	0.9596	0.4029
IsoHash	0.3101	0.9561	0.4194
ITQ	0.3815	0.9646	0.4619
AGH	**0.5195**	0.8561	0.4903
IMH	0.3107	0.7467	0.2493
InnerKSH	0.3735	0.9697	0.4942
AIBC-Q	0.4684	0.9859	0.5280
AIBC-L	0.4704	**0.9931**	**0.5595**

example, with 64-bit AIBC-L achieves 54.02% mAP while the best results of other methods is 50.95% obtained by AGH. Between the proposed methods, AIBC-L tends to achieve better MAP and precision than AIBC-Q with short binary codes (*i.e.*, 32-bit and 64-bit), while these two methods obtain close results at 128-bit. Figure 3 illustrates the precision and precision-recall curves, which clearly show AIBC performs better than other methods on this large dataset.

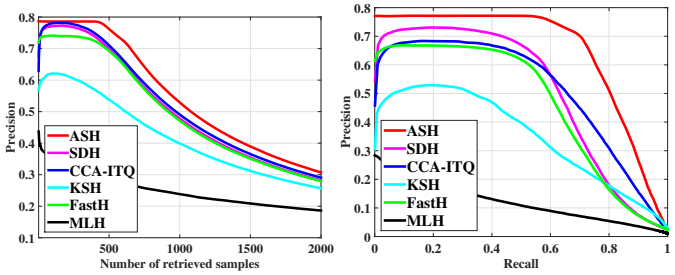
At last, we evaluate the proposed AIBC by hash lookup in the metric of Hamming distance 2 precision. The compared results of various methods with 32 bits are shown in Table IV. AGH obtains the best results (51.95%) on the SUN 397 dataset. However, AGH is surpassed by AIBC by large gaps, *i.e.*, 13.7% and 6.92% on Youtube faces and ImageNet, respectively. The proposed AIBC methods (AIBC-Q and AIBC-L) also obtain similar results under this metric.

B. Evaluation on supervised hashing

We next validate the effectiveness of the proposed asymmetric hashing algorithm ASH by comparing with several recently proposed supervised hashing algorithm. These methods include Minimal Loss Hashing (MLH) [30], Kernel-Based Supervised Hashing (KSH) [19], CCA-ITQ [7], FastH [14] and Supervised Discrete Hashing (SDH) [33]. Due to the large memory overhead and computational costs, we sample 5K and 20K training samples for MLH, KSH and FastH, respectively. For the efficient CCA-ITQ, all available training data is used. As in Section V-A, we use all training data for $\mathbf{A}$ and 10,000 samples for $\mathbf{X}$ for our method.

TABLE V: MAP and Precision500 of the compared supervised hashing methods on SUN 397 and ImageNet.

Method	mAP			Precision@500		
	32-bit	64-bit	128-bit	32-bit	64-bit	128-bit
SUN 397						
MLH	0.1558	0.2040	0.2170	0.2200	0.2824	0.3091
CCA-ITQ	0.6291	0.7241	0.7390	0.6291	0.7099	0.7170
KSH	0.6291	0.7099	0.7170	0.4815	0.5376	0.5833
FastH	0.6204	0.7043	0.7526	0.6219	0.6925	0.7295
SDH	0.6317	0.7007	0.7310	0.6526	0.6992	0.7181
ASH	**0.8087**	**0.8237**	**0.8271**	**0.7480**	**0.7617**	**0.7650**
ImageNet						
MLH	0.4106	0.4889	0.5509	0.5136	0.5939	0.6533
CCA-ITQ	0.4286	0.5959	0.7199	0.4730	0.6401	0.7601
KSH	0.4624	0.5553	0.6228	0.5488	0.6431	0.6963
FastH	0.4721	0.5791	0.6495	0.5942	0.6846	0.7339
SDH	0.5919	0.6771	0.7208	0.6867	0.7432	0.7667
ASH	**0.7501**	**0.7854**	**0.7961**	**0.7128**	**0.7480**	**0.7688**

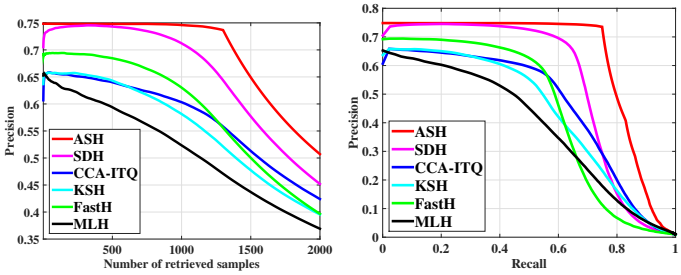


(a) Precision (b) Precision-recall

Fig. 4: Precision top 2000 curves and Precision-recall curves on SUN397 with 64 bits.

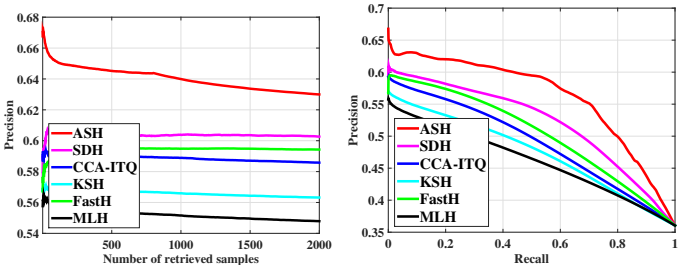
1) Retrieval with single-labled images: In this part, we use the single-labeled datasets SUN397 and ImageNet for evaluation. The comparative results in MAP and Precision@500 are reported in Table V. It can be clearly seen that on these two datasets our ASH significantly outperforms other methods in most cases. On the SUN 397 dataset, CCA-ITQ, KSH, FastH and SDH achieve on par MAPs with short bit length, while FastH perform best at 128-bit (75.26%), which is however still much lower than ASH (82.71%). In terms of precision500, ASH also outperform all other methods by over 4% accuracy. On the ImageNet dataset, SDH performs the second best and achieve close precision at 128-bit with ASH. The detailed comparison of these methods are furthered shown in Precision of top 2000 curves Precision-recall curves in Figure 4 and 5 on the two datasets. Our method shows consistent improvements over other state-of-the-art approaches. These results clearly demonstrate the effectiveness of asymmetric hashing for image search in the supervised setting.

2) Retrieval with multi-labled images: In this part, we compare these supervised hashing methods on the NUS-WIDE dataset containing images annoted by multiple lables. The NUS-WIDE database [2] contains about 270,000 images collected from Flickr. The images in NUS-WIDE are associ-



(a) Precision (b) Precision-recall

Fig. 5: Precision top 2000 curves and Precision-recall curves on ImageNet with 64 bits.



(a) Precision (b) Precision-recall

Fig. 6: Precision top 2000 curves and Precision-recall curves on **NUS-WIDE** with 64 bits.

TABLE VI: MAP (%) and Precision500 (%) of the compared supervised hashing methods on the multi-labled dataset NUS-WIDE.

Method	mAP			Precision@500		
	32-bit	64-bit	128-bit	32-bit	64-bit	128-bit
MLH	0.4633	0.4670	0.4733	0.5413	0.5587	0.5676
CCA-ITQ	0.4544	0.4627	0.4690	0.5354	0.5440	0.5533
KSH	0.4899	0.5085	0.5095	0.4066	0.5292	0.5752
FastH	0.5146	0.5239	0.5327	0.5584	0.5885	0.6140
SDH	0.5141	0.5214	0.5283	0.5399	0.5739	0.5883
ASH	**0.6307**	**0.6363**	**0.6371**	**0.6387**	**0.6452**	**0.6651**

ated with 81 concepts, with each image containing multiple semantic labels. We define the true neighbors of a query as the images sharing at least one labels with the query image. The provided 500-dimensional Bag-of-Words features are used. As suggested in [33], we collect the 21 most frequent label for test. For each label, 100 images are uniformly sampled for the query set and the remaining images are for the training set. The results in terms of retrieval performance (mAP and Precision@500) are reported in Table VI. We observe that ASH achieves promising results on this multi-labeled dataset. The precision and precision-recall curves in Figure 6 validate the advantage of the proposed ASH on this dataset.

Through the above experiments, we conclude that the proposed AIBC can achieve promising retrieval results compared to both supervised and unsupervised state-of-the-art hashing methods. We next analyze the AIBC model by the computational efficiency, parameter sensitiveness and convergence.

C. Algorithm analysis

Efficiency We take the SUN397 dataset as an example to evaluate the computational efficiency of these compared algorithms. The model training time and the testing time (of compressing one query into binary codes with the trained model) are shown in Table VII. First, we are not surprised to see that the proposed AIBC-L has a significant computational advantage over AIBC-Q, due to the simplified objective. Compared to SH, IsoHash and AGH, AIBC-L consumes more time to train the hash functions, which is mainly occupied by the inner product matrix calculation. However, the training of AIBC-L is still sufficiently efficient to scale to large-scale data:

TABLE VII: Comparison of the the computational efficiency (in consumed seconds) on SUN397. Both the training and testing time are compared with two different code lengths.

Method	Training time		Testing time	
	64-bit	128-bit	64-bit	128-bit
LSH	0.0846	0.1459	3.136e-6	3.629e-6
ALSH	0.5460	0.5976	1.296e-5	1.486e-5
SH	2.377	5.223	2.750e-5	1.060e-4
IsoHash	1.825	2.311	3.664e-6	6.335e-6
ITQ	3.326	5.296	3.527e-6	6.409e-6
AGH	0.623	0.656	1.905e-05	1.925e-05
IMH	2.427	2.964	2.402e-05	2.468e-05
InnerKSH	1933.1	4259.1	5.533e-5	6.1706e-5
AIBC-Q	53.29	173.9	3.179e-6	4.240e-6
AIBC-L	15.19	15.61	3.272e-6	3.853e-6

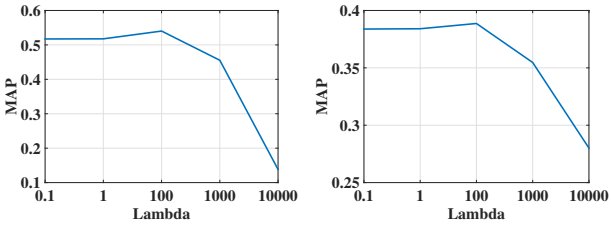
it runs only about 15 seconds on a standard PC for training with the whole 33K images of the SUN397 database. As can be seen, InnerKSH suffers from a huge computational overhead with the greedy optimization procedure, while AIBC-L can be trained much more efficiently with the closed-form solution of each sub-problem.

Parameter sensitive study In this part, we empirically evaluate the proposed algorithm (AIBC-L) on ImageNet and SUN397 with parameter λ varied from 0.1 to 10,000. We observe from Figure 7 that the performance reaches the peak at $\lambda = 100$, and drops dramatically with larger λs. In Eq. 11, the value of $\mathbf{W}^\top \mathbf{A}$ will be exactly equal to $\mathbf{B}$ with arbitrary large λ. This result demonstrate that allowing a certain discrepancy between $\mathbf{B}$ and the magnitude $\mathbf{W}^\top \mathbf{A}$ of the sign function is preferred for better performance.

Convergence study We also show in Figure 8 the objective values of our algorithm with increasing number of iterations. As can be seen, the algorithm converges within about 10 iterations. We conclude the fast convergence is due to the optimal analytical solution obtained in each sub-problem. The analytical solution not only provides fast training but also achieves high-quality codes as in the previous experiments.

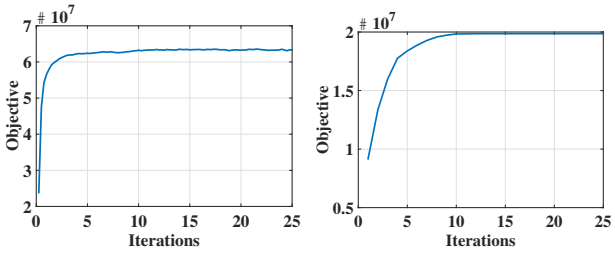
VI. Conclusion and Future Work

This paper focused on binary code learning for the maximum Inner product search (MIPS) problem through proposing an inner-product fitting framework. In the framework, two



(a) ImageNet (b) SUN397

Fig. 7: MAPs with varying λs.



(a) ImageNet (b) SUN397

Fig. 8: Objective value with iterations.

asymmetric coding functions are learned such that the inner products between original data pairs are approximated by the produced binary code vectors. Benefiting from the flexibility of the asymmetric coding mechanism, we solved the optimization in an alternating fashion involving two sub-problems. We also proposed an alternative objective that maximizes the correlations between the inner products of the pursed binary codes and raw data vectors, which enjoys a closed-form solution for each sub-problem, thus making the whole optimization more efficient. The effectiveness of the proposed Asymmetric Inner-product Binary Coding (AIBC) technique with both quadratic (AIBC-Q) and linear objectives (AIBC-L) were validated on several large-scale image datasets. As an extension of AIBC, we proposed an asymmetric supervised hashing (ASH) algorithm, which maximized the correlations between the supervised similarity matrix and code inner products.

Despite visual retrieval, the proposed AIBC can be potentially used in recommendation systems [52], visual recognition [51][25][38][31] and related tasks.

REFERENCES

[1] T. Ahonen, A. Hadid, and M. Pietikainen. Face description with local binary patterns: Application to face recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(12):2037–2041, 2006.

[2] T.-S. Chua, J. Tang, R. Hong, H. Li, Z. Luo, and Y.-T. Zheng. NUS-WIDE: A real-world web image database from national university of singapore. In *Proc. of ACM Conf. on Image and Video Retrieval*, number 48, 2009.

[3] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proc. CVPR*, pages 248–255, 2009.

[4] W. Dong, M. Charikar, and K. Li. Asymmetric distance estimation with sketches for similarity search in high-dimensional spaces. In *Proc. SIGIR*, pages 123–130, 2008.

[5] V. Erin Liong, J. Lu, G. Wang, P. Moulin, and J. Zhou. Deep hashing for compact binary codes learning. In *Proc. CVPR*, pages 2475–2483, 2015.

[6] A. Gionis, P. Indyk, and R. Motwani. Similarity search in high dimensions via hashing. In *Proc. VLDB*, pages 518–529, 1999.

[7] 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.*, 35(12):2916–2929, 2013.

[8] Y. Gong, L. Wang, R. Guo, and S. Lazebnik. Multi-scale orderless pooling of deep convolutional activation features. In *Proc. ECCV*, pages 392–407. Springer, 2014.

[9] W. Kong and W.-J. Li. Isotropic hashing. In *Proc. NIPS*, pages 1655–1663, 2012.

[10] B. Kulis and T. Darrell. Learning to hash with binary reconstructive embeddings. In *Proc. NIPS*, pages 1042–1050, 2009.

[11] B. Kulis and K. Grauman. Kernelized locality-sensitive hashing for scalable image search. In *Proc. ICCV*, pages 1092–104, 2009.

[12] H. Lai, Y. Pan, Y. Liu, and S. Yan. Simultaneous feature learning and hash coding with deep neural networks. In *Proc. CVPR*, pages 3270–3278, 2015.

[13] Z. Li, X. Liu, J. Wu, and H. Su. Adaptive binary quantization for fast nearest neighbor search. In *ECAI*, pages 64–72, 2016.

[14] 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. CVPR*, pages 1971–1978, 2014.

[15] L. Liu, Z. Lin, L. Shao, F. Shen, G. Ding, and J. Han. Sequential discrete hashing for scalable cross-modality similarity retrieval. *IEEE Trans. Image Proc.*, 26(1):107–118, 2017.

[16] L. Liu, F. Shen, Y. Shen, X. Liu, and L. Shao. Deep sketch hashing: Fast free-hand sketch-based image retrieval. In *Proc. CVPR*, 2017.

[17] L. Liu, M. Yu, F. Shen, and L. Shao. Discretely coding semantic rank orders for image hashing. In *Proc. CVPR*, 2017.

[18] W. Liu, C. Mu, S. Kumar, and S.-F. Chang. Discrete graph hashing. In *Proc. NIPS*, pages 3419–3427, 2014.

[19] W. Liu, J. Wang, R. Ji, Y.-G. Jiang, and S.-F. Chang. Supervised hashing with kernels. In *Proc. CVPR*, pages 2074–2081, 2012.

[20] W. Liu, J. Wang, S. Kumar, and S.-F. Chang. Hashing with graphs. In *Proc. ICML*, pages 1–8, 2011.

[21] X. Liu, C. Deng, B. Lang, D. Tao, and X. Li. Query-adaptive reciprocal hash tables for nearest neighbor search. *IEEE Trans. Image Proc.*, 25(2):907–919, 2015.

[22] X. Liu, B. Du, C. Deng, M. Liu, and B. Lang. Structure sensitive hashing with adaptive product quantization. *IEEE trans. Cybernetics*, 46(10):2252–2264, 2016.

[23] X. Liu, J. He, and B. Lang. Multiple feature kernel hashing for large-scale visual search. *Pattern Recognition*, 47(2):748–757, 2014.

[24] J. Lu, V. E. Liong, and J. Zhou. Deep hashing for scalable image search. *IEEE Transactions on Image Processing*, pages 2352–2367, 2017.

[25] J. Lu, V. E. Liong, X. Zhou, and J. Zhou. Learning compact binary face descriptor for face recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 37(10):2041–2056, 2015.

[26] Y. Luo, Y. Yang, F. Shen, Z. Huang, P. Zhou, and H. T. Shen. Robust discrete code modeling for supervised hashing. *Pattern Recognition*, 2017. doi:10.1016/j.patcog.2017.02.034.

[27] Y. Lv, W. W. Y. Ng, Z. Zeng, D. S. Yeung, and P. P. K. Chan. Asymmetric cyclical hashing for large scale image retrieval. *IEEE Trans. Multimedia*, 17(8):1225–1235, 2015.

[28] B. Neyshabur and N. Srebro. On symmetric and asymmetric lshs for inner product search. In *Proc. ICML*, pages 1926–1934, 2015.

[29] B. Neyshabur, N. Srebro, R. R. Salakhutdinov, Y. Makarychev, and P. Yadollahpour. The power of asymmetry in binary hashing. In *NIPS*, pages 2823–2831, 2013.

[30] M. Norouzi and D. M. Blei. Minimal loss hashing for compact binary codes. In *Proc. ICML*, pages 353–360, 2011.

[31] J. Qin, L. Liu, L. Shao, B. Ni, C. Chen, F. Shen, and Y. Wang. Binary coding for partial action analysis with limited observation ratios. In *Proc. CVPR*, 2017.

[32] F. Shen, W. Liu, S. Zhang, Y. Yang, and H. Tao Shen. Learning binary codes for maximum inner product search. In *Proc. ICCV*, pages 4148–4156, December 2015.

[33] F. Shen, C. Shen, W. Liu, and H. T. Shen. Supervised discrete hashing. In *Proc. CVPR*, pages 37–45, 2015.

[34] F. Shen, C. Shen, Q. Shi, A. van den Hengel, and Z. Tang. Inductive hashing on manifolds. In *Proc. CVPR*, pages 1562–1569, 2013.

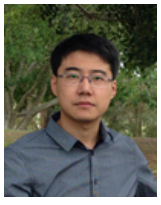
[35] F. Shen, C. Shen, Q. Shi, A. van den Hengel, Z. Tang, and H. T. Shen. Hashing on nonlinear manifolds. *IEEE Trans. Image Proc.*, 24(6):1839–1851, 2015.

[36] F. Shen, X. Zhou, Y. Yang, J. Song, H. T. Shen, and D. Tao. A fast optimization method for general binary code learning. *IEEE Trans. Image Proc.*, 25(12):5610–5621, 2016.

[37] A. Shrivastava and P. Li. Asymmetric lsh (ALSH) for sublinear time maximum inner product search (MIPS). In *Proc. NIPS*, pages 2321–

2329. 2014.
[38] J. Song, L. Gao, F. Nie, H. T. Shen, Y. Yan, and N. Sebe. Optimized graph learning using partial tags and multiple features for image and video annotation. *IEEE Trans. Image Proc.*, 25(11):4999–5011, 2016.
[39] J. Song, L. Gao, Y. Yan, D. Zhang, and N. Sebe. Supervised hashing with pseudo labels for scalable multimedia retrieval. In *ACM Multimedia*, pages 827–830, 2015.
[40] J. Song, H. T. Shen, J. Wang, Z. Huang, N. Sebe, and J. Wang. A distance-computation-free search scheme for binary code databases. *IEEE Trans. Multimedia*, 18(3):484–495, 2016.
[41] J. Song, Y. Yang, Z. Huang, H. Shen, and J. Luo. Effective multiple feature hashing for large-scale near-duplicate video retrieval. *IEEE Trans. Multimedia*, 15:1997 – 2008, 2013.
[42] J. Song, Y. Yang, Y. Yang, Z. Huang, and H. T. Shen. Inter-media hashing for large-scale retrieval from heterogeneous data sources. In *SIGMOD*, pages 785–796, 2013.
[43] J. Wang, S. Kumar, and S.-F. Chang. Semi-supervised hashing for large scale search. *IEEE Trans. Pattern Anal. Mach. Intell.*, 34(12):2393–2406, 2012.
[44] J. Wang, T. Zhang, J. Song, N. Sebe, and H. T. Shen. A survey on learning to hash. *arXiv preprint arXiv:1606.00185*, 2016.
[45] Y. Weiss, R. Fergus, and A. Torralba. Multidimensional spectral hashing. In *Proc. ECCV*, pages 340–353, 2012.
[46] Y. Weiss, A. Torralba, and R. Fergus. Spectral hashing. In *Proc. NIPS*, pages 1753–1760, 2008.
[47] L. Wolf, T. Hassner, and I. Maoz. Face recognition in unconstrained videos with matched background similarity. In *Proc. CVPR*, pages 529–534, 2011.
[48] F. Wu, Z. Yu, Y. Yang, S. Tang, Y. Zhang, and Y. Zhuang. Sparse multi-modal hashing. *IEEE Trans. Multimedia*, 16(2):427–439, 2014.
[49] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *Proc. CVPR*, pages 3485–3492, 2010.
[50] 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*, 1(4):162–171, 2015.
[51] Y. Yang, Z.-J. Zha, Y. Gao, X. Zhu, and T.-S. Chua. Exploiting web images for semantic video indexing via robust sample-specific loss. *IEEE Trans. Multimedia*, 16(6):1677–1689, 2014.
[52] H. Zhang, F. Shen, W. Liu, X. He, H. Luan, and T.-S. Chua. Discrete collaborative filtering. In *SIGIR*, pages 325–334, 2016.
[53] H. Zhang, M. Wang, R. Hong, and T.-S. Chua. Play and rewind: Optimizing binary representations of videos by self-supervised temporal hashing. In *ACM Multimedia*, pages 781–790, 2016.

Fumin Shen received his Bachelor degree at 2007 and PhD degree at 2014 from Shandong University and Nanjing University of Science and Technology, China, respectively. Now he is an Associate Professor with University of Electronic Science and Technology of China. His major research interests include computer vision and machine learning. He was the recipient of the Best Paper Award Honorable Mention at ACM SIGIR 2016 and the World's FIRST 10K Best Paper Award - Platinum Award at IEEE ICME 2017.



Yang Yang is currently with University of Electronic Science and Technology of China. He was a Research Fellow under the supervision of Prof. Tat-Seng Chua in National University of Singapore during 2012-2014. He was conferred his Ph.D. Degree (2012) from The University of Queensland, Australia. During the PhD study, Yang Yang was supervised by Prof. Heng Tao Shen and Prof. Xiaofang Zhou. He obtained Master Degree (2009) and Bachelor Degree (2006) from Peking University and Jilin University, respectively.

Li Liu received the B.Eng. degree in electronic information engineering from Xian Jiaotong University, Xian, China, in 2011, and the Ph.D. degree from the Department of Electronic and Electrical Engineering, University of Sheffield, Sheffield, U.K., in 2014. He is currently a Senior Consultant with Malong Technologies Co., Ltd, China. His current research interests include computer vision, machine learning, and data mining.

Wei Liu received the M.Phil. and Ph.D. degrees in electrical engineering from Columbia University, New York, NY, USA, in 2012. He is currently a Research Scientist with Tencent AI Lab, and holds an adjunct faculty position with Rensselaer Polytechnic Institute, Troy, NY, USA. In 2012, he was the Josef Raviv Memorial Post-Doctoral Fellow with the IBM T. J. Watson Research Center, for one year. His current research interests include machine learning, data mining, computer vision, pattern recognition, and information retrieval.

Dacheng Tao (F'15) is currently a Professor with the The University of Sydney. He mainly applies statistics and mathematics to Artificial Intelligence and Data Science. His research interests spread across computer vision, data science, image processing, machine learning, and video surveillance. His research results have expounded in one monograph and over 200 publications at prestigious journals and prominent conferences, such as the IEEE T-PAMI, T-NNLS, T-IP, JMLR, IJCV, NIPS, ICML, CVPR, ICCV, ECCV, AISTATS, ICDM, and the ACM SIGKDD, with several best paper awards, such as the Best Theory/Algorithm Paper Runner Up Award in the IEEE ICDM07, the Best Student Paper Award in the IEEE ICDM13, and the 2014 ICDM 10-Year Highest Impact Paper Award. He received the 2015 Australian Scopus-Eureka Prize, the 2015 ACS Gold Disruptor Award, and the 2015 UTS Vice-Chancellors Medal for Exceptional Research. He is a fellow of the OSA, IAPR, SPIE and IEEE.

Heng Tao Shen is a Professor of National "Thousand Talents Plan" and the director of Center for Future Media at the University of Electronic Science and Technology of China (UESTC). He obtained his BSc with 1st class Honours and PhD from Department of Computer Science, National University of Singapore (NUS) in 2000 and 2004 respectively. He then joined the University of Queensland (UQ) as a Lecturer, Senior Lecturer, Reader, and became a Professor in late 2011. His research interests mainly include Multimedia Search, Computer Vision, and Big Data Management on spatial, temporal, and multimedia databases. He has published over 150 peer-reviewed papers, most of which are in prestigious international venues of interests. For his outstanding research contributions, he received the Chris Wallace Award in 2010 conferred by Computing Research and Education Association, Australasia, and the Future Fellowship from Australia Research Council in 2012. He is an Associate Editor of IEEE Transactions on Knowledge and Data Engineering, and has organized ICDE 2013 as Local Organization Co-Chair, and ACM Multimedia 2015 as Program Committee Co-Chair.