Asymmetric Transfer Hashing With Adaptive Bipartite Graph Learning

Jianglin Lu, Jie Zhou, Member, IEEE, Yudong Chen, Witold Pedrycz, Life Fellow, IEEE, and Kwok-Wai Hung

Abstract—Thanks to the efficient retrieval speed and low storage consumption, learning to hash has been widely used in visual retrieval tasks. However, the known hashing methods assume that the query and retrieval samples lie in homogeneous feature space within the same domain. As a result, they cannot be directly applied to heterogeneous cross-domain retrieval. In this article, we propose a generalized image transfer retrieval (GITR) problem, which encounters two crucial bottlenecks: 1) the query and retrieval samples may come from different domains, leading to an inevitable domain distribution gap and 2) the features of the two domains may be heterogeneous or misaligned, bringing up an additional feature gap. To address the GITR problem, we propose an asymmetric transfer hashing (ATH) framework with its unsupervised/semi-supervised/supervised realizations. Specifically, ATH characterizes the domain distribution gap by the discrepancy between two asymmetric hash functions, and minimizes the feature gap with the help of a novel adaptive bipartite graph constructed on cross-domain data. By jointly optimizing asymmetric hash functions and the bipartite graph, not only can knowledge transfer be achieved but information loss caused by feature alignment can also be avoided. Meanwhile, to alleviate negative transfer, the intrinsic geometrical structure of single-domain data is preserved by involving a domain affinity graph. Extensive experiments on both single-domain and cross-domain benchmarks under different GITR subtasks indicate the superiority of our ATH method in comparison with the state-of-the-art hashing methods.

I. INTRODUCTION

DUE TO the fast retrieval speed and low memory footprint, learning to hash [1], [2], [3], [4], [5] has attracted extensive attention in different visual tasks over the past decades. The hashing approach aims to encode the data into compact binary codes so as to conduct fast retrieval based on hardware-level XOR operation. Recently, a large number of hashing methods have been proposed from different perspectives [6], [7], [8], [9], [10]. However, the known hashing approaches mainly rely on two assumptions: 1) sufficient training samples are required to learn discriminative hash functions and 2) query and retrieval samples come from the same domain with the same data distribution. In real-world scenarios, these assumptions are often easily violated. Specifically, the data available in the domain of interest may be inadequate for training, leading to data sparsity problem [11]. Moreover, there may exist an inevitable data distribution discrepancy between the query and retrieval domains. To tackle these drawbacks, a feasible solution is to integrate hashing approaches with transfer learning (TL) [12]. TL aims to enhance the performance of the target solver/learner on target domain by transferring the knowledge contained in different but related source domain [13], [14]. As a result, the dependence on sufficient target domain data can be mitigated for target solver construction.

Inspired by TL, a few number of transfer hashing (TH) methods have been proposed recently [11], [15], [16], [17], [18]. The first TH method called TH with privileged information (THPI) [11] marries hashing and TL approaches to solve the data sparsity issue. Unlike THPI, an Optimal Projection guided TH (GTH) [16] seeks for a maximum-likelihood estimation solution of hash functions of target and source domains. The newly developed probability-weighted compact feature learning (PWCF) [17] provides interdomain correlation guidance to promote cross-domain retrieval accuracy, and discriminative hashing learning (DHLing) [18] constructs a domain-invariant memory bank to achieve cross-domain alignment. The above-mentioned methods, however, are all homogeneous TH methods and cannot be directly applied for heterogeneous domains. Specifically, these methods assume that target and source domains share the same feature space with the same feature dimensionality. This assumption may not always hold since the features of the two
domains may be heterogeneous or misaligned. For example, the query and retrieval samples may be of different resolutions and sizes, represented by hand-crafted features with different types and lengths (e.g., color histogram [19] and SIFT [20]), or extracted by distinctive deep neural networks (e.g., VGG [21] and ResNet [22]). In these cases, the known TH methods need an additional feature preprocessing (e.g., image scaling and dimension reduction) for feature alignment, which may cause information loss to a certain extent.

In this article, we first summarize a more challenging and practical transfer retrieval task called generalized image transfer retrieval (GITR), in which the query and retrieval images may come from different domains and their features may be homogeneous or heterogeneous. Fig. 1 gives an illustration of the proposed GITR, which contains four different subtasks, including heterogeneous cross-domain retrieval (HeCDR), homogeneous cross-domain retrieval (HoCDR), heterogeneous single-domain retrieval (HeSDR), and homogeneous single-domain retrieval (HoSDR). Without any limitation on the feature spaces and domains of query and retrieval sets, this article aims to develop a generalized framework to solve the complicated GITR problem. Specifically, given a query image observed in $F_1$ feature space from $D_1$ domain, we aim to retrieve the most similar images represented in $F_2$ feature space from $D_2$ domain, no matter whether these two domains and two feature spaces are the same or not. To achieve this goal, we propose an asymmetric TH (ATH) framework with its unsupervised/semisupervised/supervised realizations. The proposed ATH contains three main components: 1) asymmetric hashing learning, which characterizes domain distribution gap by the discrepancy of two asymmetric hash functions; 2) cross-domain information interaction, which optimizes an adaptive bipartite graph for knowledge transfer and information fusion of cross-domain data; and 3) domain structure preserving, which alleviates negative transfer by maintaining the intrinsic geometrical structure of single-domain data based on domain affinity graph. The main contributions of this article can be summarized as follows.

1) We extend the current learning-to-hash problem to a more general and challenging case, dubbed GITR, which makes no assumption or restrictions on the domains and feature spaces of query and retrieval sets.

2) To solve the GITR problem, we propose an ATH framework with its unsupervised/semisupervised/supervised realizations. Specifically, ATH finds two asymmetric hash functions to encode the samples into discriminative hash codes, and learns an adaptive bipartite graph to characterize the similarity between cross-domain samples.

3) To our best knowledge, ATH is the first heterogeneous TH framework that jointly optimizes the asymmetric hash functions and an adaptive bipartite graph in a unified learning objective, solving the two key problems in GITR simultaneously (see Fig. 2).

4) Extensive experiments conducted on both single-domain and cross-domain benchmarks indicate that the known hashing methods tailored to a special feature space and domain cannot achieve satisfactory performance, while our ATH framework can work well in different scenarios.

The remainder of this article is organized as follows. The proposed GITR problem is illustrated in Section II where we also review some related works. Section III presents the proposed ATH framework with its unsupervised/semisupervised/supervised realizations. In Section IV, we design an iterative algorithm to optimize the objective of ATH and discuss the relationships between ATH and some related works. Section V reports on the experimental results for evaluating the performance of ATH. Conclusions and future work are covered in Section VI.

**II. PRELIMINARIES**

**A. Generalized Image Transfer Retrieval**

Suppose that we have $n = n_s + n_t$ data samples from two domains $D = \{D_s, D_t\}$, where source domain $D_s$ and
target domain. Due to $n_t \neq n_s$, THPI only uses $n_t$ source-domain samples to conduct information interaction with $n_t$ target domain samples, which may weaken the transfer ability of target learner. In addition, how to select $n_t$ samples from the entire source domain remains to be determined.

Unlike THPI, GTH [16] does not require to select a part of data from the source domain for cross-domain information interaction. Its objective can be formulated as follows:

$$
\begin{aligned}
\min_{B_s, B_t, A_s, A_t} & \left| \left| M^2 \odot (A_t - A_s) \right| \right|^2 + \lambda_1 \left| \left| B_s - A_s^T X_s \right| \right|^2 \\
+ & \lambda_2 \left| \left| B_t - A_t^T X_t \right| \right|^2, \\
\text{s.t.} & A_t^T A_t = I, \ A_s^T A_s = I
\end{aligned}
$$

(2)

where $\odot$ denotes element-wise product, $M$ is an error matrix derived from maximum-likelihood estimation, $A_s \in \mathbb{R}^{d_s \times r}$ and $A_t \in \mathbb{R}^{d_t \times r}$ are hashing projections for domain $D_s$ and $D_t$, respectively. GTH assumes that the similarities of images between target and source domains can be reversely characterized by discrepancy between hashing projections of the two domains. In other words, the weighted discrepancy between hashing projections $M^{(1/2)} \odot (A_t - A_s)$ should be minimized to ensure that similar images from the target and source domains can be converted into similar hash codes. In this way, the entire knowledge in the source domain can be used for target learner.

Instead of minimizing the discrepancy between hashing projections, PWCF [17] designs a focal-triplet loss to promote cross-domain correlations and obtains good transfer performance. Moreover, DHLing [18] further considers the problem of insufficient annotated source images and provides a semisupervised approach. Nevertheless, all these TH methods [11], [16], [17], [18] assume that the target and source domains share a homogeneous feature space with aligned features (i.e., $d_s = d_t = d$). This assumption may not always hold since the feature dimensionalities of retrieval and query samples may be misaligned. For example, the query and retrieval samples may be of distinctive resolutions and sizes, represented by hand-crafted features with different lengths, or extracted by different deep neural networks. In these cases, these methods cannot be directly applied to HeSDR and HeCDR subtasks (where $d_t \neq d_t$), which motivates us to develop a novel heterogeneous TH method to alleviate this problem.

### B. Homogeneous Transfer Hashing

The first TH method THPI [11] incorporates privileged information from the source domain into the target domain to assist hashing learning. Its objective function can be formulated as follows:

$$
\begin{aligned}
\min_{B_s, A_s, A_t} & \left| \left| E \right| \right|^2 + \lambda_1 \left| \left| E - A_s^T X_s \right| \right|^2 \\
\text{s.t.} & A_t^T A_t = I, \ A_s^T A_s = I
\end{aligned}
$$

(1)

where $E = B_t - A_t^T X_t$, $X_s \in \mathbb{R}^{d_s \times n_t}$ is $n_t$ selected samples from source domain $D_s$, $A_t \in \mathbb{R}^{d_t \times r}$ and $A_s \in \mathbb{R}^{d_s \times r}$ are orthogonal projection matrices for data $X_t$ and $X_s$, respectively. As we can see, THPI utilizes data information of the source domain to approximate the quantization errors of the

1 Note that, the definition of TH problem is different from that of cross-modal hashing. Refer to [11], [16], [17], and [18] for more details.

---

**Fig. 2.** Technically, the GITR faces two dilemmas: $d_s \neq d_t$ and $n_t \neq n_s$. For $d_s \neq d_t$, ATH learns two distinct hash functions $f_s$ and $f_t$ to transform the heterogeneous data into an alignment mapping space. For $n_t \neq n_s$, ATH optimizes an adaptive bipartite graph to integrate the entire cross-domain data for transfer training. Moreover, in order to alleviate negative transfer, ATH preserves the intrinsic geometrical structure of single-domain data by involving a domain affinity graph. Best viewed in colors.
a so-called similarity-similarity difference function [1] for similarity preservation

\[ \sum_i \sum_j ||f_1(x_i)f_2(x_j) - rS||^2 \]  

(3)

where \( S \) is a predefined label matrix, \( f_1(\cdot) \) and \( f_2(\cdot) \) are two distinct hash functions. Technically, they approximate the similarity between samples \( x_i \) and \( x_j \) as the Hamming distance between \( f_1(x_i) \) and \( f_2(x_j) \). The more similar the two samples \( x_i \) and \( x_j \) are, the smaller the Hamming distance between \( f_1(x_i) \) and \( f_2(x_j) \) should be. Being different from these approaches which focus on single-domain retrieval, this article utilizes the asymmetric hashing mechanism to solve the GITR problem, especially for its challenging HeSDR and HeCDR subtasks. More importantly, since there is no label information provided for the construction of \( S \) in the unsupervised scenario, we design a novel distance-similarity product function equipped with an adaptive bipartite graph to learn the potential semantic information from data.

III. METHODOLOGY

A. Graph and Bipartite Graph

Suppose that \( G = (V, E, W) \) is an undirected weighted graph with vertex set \( V(G) \), edge set \( E(G) \), and adjacent matrix \( W \), where each element of the real symmetric matrix \( W \) measures the affinity for a pair of vertices. In general, there are three main approaches for the construction of the affinity graph: 1) nearest neighbor graph [30], [31], [32], which focuses on the local neighborhood relationship of the data; 2) semantic graph [33], [34], [35], which utilizes prior semantic information of labels; and 3) adaptive graph [36], [37], [38], which learns adaptive graph structure from the data. On the other hand, the bipartite graph has been extensively explored in different applications [39], [40], which can be defined as follows.

Definition 1: A graph \( G = (V, E, W) \) is called bipartite if \( V = V_1 \cup V_2 \) and \( V_1 \cap V_2 = \emptyset \), such that every edge in \( E \) connects one vertex from \( V_1 \) to one vertex from \( V_2 \).

Note that, the affinity matrix \( W \in \mathbb{R}^{|V_1| \times |V_2|} \) of a bipartite graph \( G = (V, E, W) \) will be asymmetric if \( |V_1| \neq |V_2| \).

B. Unsupervised ATH

Fig. 2 illustrates the core idea of the proposed ATH framework, which contains three main parts, including asymmetric hashing learning, cross-domain information interaction, and domain structure preserving. In this section, we first consider an unsupervised ATH (ATH_U), where neither the source domain nor the target domain contain labeled samples as the application scenarios in [11] and [16].

1) Asymmetric Hashing Learning: In ATH, we exploit two distinct hash functions \( f_t \) and \( f_s \) to encode target and source samples separately, as follows:

\[ \min_{B_{tk}, f_t} T_1 = \sum_{k \in \{s, t\}} \alpha_k ||B_k - f_t(X_k)||_F^2 \] 

s.t. \( B_k \in \{-1, 1\}^{r \times n_k}, B_k 1 = 0 \)  

(4)

where \( \alpha_k \in \{\alpha_s, \alpha_t\} \) is a balance parameter, and \( f_k \in \{f_t, f_s\} \) is the hash function of domain \( D_k \). We impose a constraint \( B_k 1 = 0 \) to make each bit to have an equal chance to be 1 or -1, thereby maintaining the most information of data. The designed optimization objective (4) aims to model the fitting error between the binary code \( B_k \) and hash mapping \( f_k(x_k) \) for each domain \( D_k \). Note that, \( f_k(\cdot) \) can be any suitable mapping function, such as linear function and kernel function. For ATH_U, we just exploit a simple yet effective linear function

\[ f_k(x_{k,i}) = A_k^T x_{k,i} \]  

(5)

where \( A_k \in \{A_s, A_t\} \) is a hashing projection. Most notably, \( A_s \in \mathbb{R}^{d_k \times r} \) and \( A_t \in \mathbb{R}^{d_t \times r} \) can be of the same magnitude or not. If \( d_s = d_t \), it can be used for HoSDR and HoCDR. Otherwise, it is tailored to HeSDR and HeCDR.

2) Cross-Domain Information Interaction: We consider conducting direct knowledge transfer and fusion between target and source domains, regardless of whether these two domains share the same feature space or not. Obviously, it is infeasible to minimize the errors between \( A_s \) and \( A_t \) as in [16] since the cross-domain features may be misaligned. According to Definition 1, we observe that if we regard the source and target data as an entire dataset, on which we can naturally construct a bipartite graph \( G_{st} = (V_{st}, E_{st}, W_{st}) \) where \( V_{st} = X_s \cup X_t \) and \( W_{st} \in \mathbb{R}^{m_s \times m_t} \). Nevertheless, how to determine the edge set \( E_{st} \) and edge weight \( W_{st} \) becomes a knotty problem. Note that, we cannot apply a nearest neighbor bipartite graph in HeSDR and HeCDR subtasks since the distances between heterogeneous features of cross-domain data cannot be calculated directly, as shown in Fig. 2. To mitigate this problem, we propose to learn an adaptive bipartite graph from data to characterize the affinities of cross-domain samples, by the following distance-similarity product function:

\[ \min_{W_{st}, f_t} T_2 = \lambda \sum_{i=1}^{n_s} \sum_{j=1}^{n_t} F_{st,ij} W_{st,ij} + \gamma ||W_{st}||_F^2 \] 

s.t. \( \forall i, \ W_{st,i} \geq 0, \ W_{st,i}^T 1 = 1 \)  

(6)

where \( \lambda \) and \( \gamma \) are two hyper-parameters, \( W_{st,i} \) is the \( i \)-th row of \( W_{st} \), and \( F_{st,ij} = ||f_t(x_{s,i}) - f_t(x_{t,j})||_2^2 \) models the distances (or fitting errors) between hash mappings of cross-domain samples. The value of \( W_{st,ij} \) can be viewed as the probability that two cross-domain samples \( x_{s,i} \) and \( x_{t,j} \) belong to the same class. Therefore, we make the constraint \( W_{st,i} 1 = 0 \). In addition, the constraint \( W_{st,i} 1 = 1 \) is imposed to avoid a trivial case where all elements of \( W_{st,i} \) are zero, and the regularization term \( \gamma ||W_{st}||_F^2 \) provides the probability prior of uniform distribution. As we can see, the designed distance-similarity product function (6) enables the solver to simultaneously learn hash functions and affinity relationship of cross-domain samples, regardless of whether the features of the two domains are misaligned or not. This optimization objective stems from a straightforward intuition that the more similar two cross-domain samples (i.e., the larger \( W_{st,ij} \)) are, the less difference the corresponding hash mappings (i.e., the smaller \( F_{st,ij} \)) should be. The joint optimization of asymmetric hash functions and bipartite graph promotes knowledge fusion between
Algorithm 1: Optimization Algorithm for ATH_U

Input: Training sets $X_k \in \{X_t, X_s\}$, maximum iterations $Ite$, parameters $\alpha_k, \beta_k, \lambda, r$.

Output: Hashing projections $A_k \in \{A_s, A_t\}$.

1. Construct affinity matrices $W_k \in \{W_s, W_t\}$; 
2. Initialize $A_k$ by PCA projections; 
3. Initialize $B_k$ by $B_k = \text{sign}(A_t^T X_k)$; 
4. while converge or reach maximum iterations $Ite$ do 
    5. Update $A_k$ according to (17); 
    6. Update $M_k$ by $M_k = \alpha_k A_k^T X_k$; 
    7. for $i = 1 : r$ do 
        8. Update $B_{k,ij}$ according to (19); 
    end 
    9. for $i = 1 : n_s$ do 
        10. Update $W_{st,i}$ according to (21); 
    end 
11. end

C. Semisupervised ATH

This section provides a semisupervised realization of ATH called ATH_M, where all samples in the source domain are well annotated as the application scenarios in [17] and [18]. The difference of objective functions between ATH_U and ATH_M lies in the construction of source graph. Unlike ATH_U, ATH_M adopts a semantic graph for the source domain. Specifically, ATH_M defines the edge set $E_s$ of source graph $G_s = (X_s, E_s, W_s)$ as follows:

$$E_s = \{ <x_s,i, x_s,j> | y_s,i = y_s,j \}. \quad (11)$$

The edge weight setting of ATH_M is similar to that of ATH_U. As can be seen from (11), ATH_M exploits semantic information of the source domain to perform transfer training, which promotes the performance of hash encoders.

D. Supervised ATH and Kernel Extension

This section presents a supervised realization of ATH dubbed ATH_S, which is tailored to the case where all samples in the source and target domains are well labeled. With the available label information, we do not need to optimize an adaptive bipartite graph for cross-domain knowledge transfer as in (6). Instead, ATH_S proactively constructs a semantic bipartite graph and sets its edge set $E_{st}$ as follows:

$$E_{st} = \{ <x_s,i, x_t,j> | y_s,i = y_t,j \}. \quad (12)$$

In this way, the semantic similarity of cross-domain data can be used to conduct domain alignment and improve the transferability of our model. As for the source and target graphs $G_k = (X_k, E_k, W_k)$, we also utilize semantic information to construct the edge set $E_k$ for each domain $D_k$ as follows:

$$E_k = \{ <x_k,i, x_k,j> | y_k,i = y_k,j \}. \quad (13)$$

As mentioned in (4), we can use a kernel function for realizing $f_k$. To explore the performance of the kernel function, we further present a kernel extension of ATH_S called ATH_K, of which the hash function can be formulated as follows:

$$f_k(x_k,i) = A_k^T \phi(x_k,i) \quad (14)$$

where $\phi(x_k,i)$ is a $m_k \in \{m_s, m_t\}$-dimensional column vector obtained by the Gaussian kernel

$$\phi(x_k,i) = \left[ \exp\left( \frac{||x_k,i - x_k,1||^2}{\sigma} \right), \ldots, \exp\left( \frac{||x_k,i - x_k,m_k||^2}{\sigma} \right) \right]^T \quad (15)$$

where $\{x_k,1\}_{j=1}^{m_k}$ are $m_k$ anchor points randomly selected from domain $D_k$, and $\sigma$ is the kernel width [41].

IV. DISCUSSION

A. Optimization

Due to space limitation, we only provide the optimization algorithm for ATH_U. The optimization approaches of ATH_M, ATH_S, and ATH_K are similar to that of ATH_U. Specifically, our iterative algorithm updates one variable at a time by fixing the others, as listed in Algorithm 1.
The subproblem of $A_k \in \{A_i, A_t\}$ is equal to the following unconstrained minimization problem:

$$
\min_{A_k} \alpha_k \|B_k - A_k^T X_k\|^2_F + \beta_k \text{tr}(A_k^T L_k X_k^T A_k)
+ \lambda_k \text{tr}(-2A_k^T X_k \tilde{W}_{st, k} X_k^T A_k + A_k^T X_k \tilde{D}_{st, k} X_k^T A_k)
+ \lambda_k \text{tr}(A_k^T (X_k(\alpha_k I + \beta_k L_k + \lambda_k \tilde{D}_{st, k}) X_k^T) A_k)
- 2A_k^T X_k (\lambda_k B_k + \lambda_k \tilde{W}_{st, k} X_k^T) A_k)
$$

(16)

where $\tilde{D}_{st, k}$ is a diagonal matrix, $L_k \in \{L_i, L_t\}$ is the graph Laplacian matrix, $L_k = D_k - W_k$, and $D_k \in \mathbb{R}^{n_k \times n_k}$ is a diagonal matrix with $D_{k, i} = \sum W_{k, ij}$. In (16), if $k = s$, $\tilde{D}_{st, i} = \sum W_{st, ij}, \tilde{W}_s = \tilde{W}_t, X_k = X_s$, and $A_k = A_s$. Otherwise, $\tilde{D}_{st, k} = \sum_{j} W_{st, i}, \tilde{W}_k = \tilde{W}_{st, i}, X_k = X_s$, and $A_k = A_t$. By setting the derivative of (16) with respect to $A_k$ to zero, the optimal solution of $A_k$ subproblem can be obtained by:

$$
A_k = \Phi_k^{-1} (\alpha_k X_k B_k^T + \lambda_k X_k \tilde{W}_{st, k} X_k^T \tilde{A}_k)
$$

(17)

where $\Phi_k = X_k(\alpha_k I + \beta_k L_k + \lambda_k \tilde{D}_{st, k}) X_k^T$. The optimal solution of this maximization problem can be directly obtained by sorting the elements of each row of $M_k$

$$
B_{k, i,j} = \begin{cases} 
1, & \text{if } j \in \tau(i) \\
-1, & \text{otherwise}
\end{cases}
$$

(19)

where $\tau(i)$ indicates an index set of the first $n_i/2$ maximal elements of $M_{k, i, i}$, and $M_{k, i, i}$ indicates the $i$th row of $M_k$.

The subproblem of $W_{st, i}$, we need to address $n_i$ decoupled independent problems as follows:

$$
\min_{W_{st, i}} \|W_{st, i} + \frac{\lambda}{2\gamma_i} \tilde{F}_{st, i}\|^2 \\
\text{s.t. } W_{st, i} \geq 0, \quad W_{st, i}^T 1 = 1.
$$

(20)

For problem (20), we have the following theorem.

**Theorem 1:** Let the number of nonzero elements in $W_{st, i}$ be $\eta \in \{1, \ldots, n_i\}$. Then, the optimal solution of (20) is

$$
W_{st, i} = \left(1 + \frac{\sum_{j=1}^{\eta} \frac{\lambda}{2\gamma_i} \tilde{F}_{st, i}}{\eta} \right) - \frac{\lambda}{2\gamma_i} \tilde{F}_{st, i}
$$

(21)

where the operator $(\cdot)_+$ turns the negative elements to zero while keeps the rest, and the elements of $\tilde{F}_{st, i}$ are those of $F_{st, i}$, but with the ascending order.

**Proof:** The Lagrangian function of the $W_{st, i}$ subproblem can be formulated as follows:

$$
\mathcal{L} = \frac{1}{2} \|W_{st, i} + \frac{\lambda}{2\gamma_i} \tilde{F}_{st, i}\|^2 - \varphi(W_{st, i}^T 1 - 1) - u^T W_{st, i}
$$

(22)

where $\varphi$ and $u$ are Lagrangian multipliers. By taking the derivate of $\mathcal{L}$ with respect to $W_{st, i}$ and setting it zero, with the KKT condition, we obtain

$$
\frac{\partial \mathcal{L}}{\partial W_{st, i}} = W_{st, i} + \frac{\lambda}{2\gamma_i} \tilde{F}_{st, i} - \varphi I - u = 0
$$

(23)

For the third equation, if $W_{st, i} > 0$, then we have $u_j = 0$. Therefore, the solution of $W_{st, i}$ is

$$
W_{st, i} = \left(\varphi I - \frac{\lambda}{2\gamma_i} \tilde{F}_{st, i}\right)^+.
$$

(24)

Since $W_{st, i}$ has $\eta$ nonzero elements and $W_{st, i} \geq 0$, we have

$$
\varphi - \frac{\lambda}{2\gamma_i} \tilde{F}_{st, i,j} > 0 \bigcap \varphi - \frac{\lambda}{2\gamma_i} \tilde{F}_{st, i,(\eta+1)} \leq 0.
$$

(25)

According to $W_{st, i}^T 1 = 1$ and (24), we obtain

$$
\sum_{j=1}^{\eta} (\varphi - \frac{\lambda}{2\gamma_i} \tilde{F}_{st, i,j}) = 1 \Rightarrow \varphi = \frac{1 + \sum_{j=1}^{\eta} \frac{\lambda}{2\gamma_i} \tilde{F}_{st, i,j}}{\eta}.
$$

(26)

Substitute (26) into (24), and we can obtain the same solution as stated in Theorem 1.

**B. Parameter Setting of $\gamma$**

In this section, we show how to determine the value of parameter $\gamma$. Combining (25) and (26), we can obtain

$$
\frac{\lambda}{2\gamma_i} \tilde{F}_{st, i,j} < \gamma_i \leq \frac{\lambda}{2\gamma_i} \tilde{F}_{st, i,(\eta+1)} - \frac{\lambda}{2\gamma_i} \sum_{j=1}^{\eta} \tilde{F}_{st, i,j}
$$

(27)

In order to make the optimal solution of $W_{st, i}$ contain $\eta$ nonzero elements, we can set $\gamma_i$ as follows:

$$
\gamma_i = \frac{\lambda}{2\gamma_i} \tilde{F}_{st, i,(\eta+1)} - \frac{\lambda}{2\gamma_i} \sum_{j=1}^{\eta} \tilde{F}_{st, i,j}.
$$

(28)

As a result, we do not need to turn the value of parameter $\gamma$. We can set its value as the average of $\gamma_i$ for $i \in \{1, 2, \ldots, n_i\}$

$$
\gamma = \frac{1}{n_i} \sum_{i=1}^{n_i} \left(\frac{\lambda}{2\gamma_i} \tilde{F}_{st, i,(\eta+1)} - \frac{\lambda}{2\gamma_i} \sum_{j=1}^{\eta} \tilde{F}_{st, i,j}\right).
$$

(29)

**C. Connections to Previous Methods**

We analyze the connections and differences between the proposed ATH and some related hashing methods, including anchor-graph-based hashing [31, 42, 43], asymmetric hashing [9, 26, 29], and TH [11, 16, 17, 18].

The anchor-graph-based hashing methods [31, 42, 43] aim to perform fast graph construction for large-scale image retrieval. In fact, an anchor graph can be viewed as a special bipartite graph. Unlike these methods focusing on single-domain image retrieval, we utilize the bipartite graph to integrate cross-domain information for TH. Moreover, these methods are all two-step approaches, which first preconstruct an anchor graph through clustering and then use it to learn
hash functions. Nevertheless, our proposed ATH can learn the bipartite graph and hash functions simultaneously in a unified framework.

The asymmetric hashing methods [9], [26], [29] emphasize achieving similarity approximation with short binary codes. To achieve this goal, they usually adopt the so-called similarity-similarity difference function [1] for similarity preservation. Unlike these approaches, the proposed ATH aims to learn distinguish hash functions with unaligned inputs for the source and target domains by incorporating the asymmetric hashing mechanism. Therefore, ATH can solve the TH problem, especially for the HeCDR task with misaligned features. Moreover, ATH constructs a distance-similarity product function [1] for similarity preserving, which helps to jointly optimize the bipartite graph and hash functions.

The TH methods [16], [17], [18] are proposed to deal with cross-domain retrieval task. These methods find either a shared hash function or two different hash functions with aligned inputs for source and target domains. As a result, they cannot be directly applied in HeSDR and HeCDR tasks, since the feature dimensionalities of source and target domains in these two tasks are inconsistent. The proposed ATH can handle this problem by learning asymmetric hash functions. Moreover, the TH method [11] need to select a part of source samples to integrate information with the target samples, while our ATH can perform cross-domain information interaction for all source and target sample through the learning of a bipartite graph.

D. Time Complexity

As listed in Algorithm 1, the main computation burden of ATH_U comes from iteratively updating \( A_k, B_k, \) and \( W_{st} \). The main computation cost for optimizing \( A_k \) and \( B_k \) are \( O(d_k^3) \) and \( O(m_k \log n_k) \), respectively. And, we need \( O(n_s n_t \log n_s) \) for the optimization of \( W_{st} \). In the TH task, there usually exists \( n_s \ll n_t \). Therefore, the computation cost of our algorithm is acceptable.

V. EXPERIMENTS

In our experiments, we compare the proposed ATH_U with unsupervised hashing methods ITQ [44], SP [45], DPLM [46], SHSR [47], and TH method GTH [16]. For ATH_K and ATH_S, we compare them with supervised hashing methods KSH [48], SDH [41], FSDH [49], and SCDH [50]. For ATH_M, we compare it with KSH, SDH and TH methods PWCF [17], and DHLing [18].

A. Datasets

To evaluate the performance of our ATH framework, we conduct experiments both on the single-domain and cross-domain datasets. The used datasets are described as follows.

1) Single-Domain Datasets: We first evaluate the proposed ATH_U on single-domain datasets, including CIFAR-10 and SUN397 [51] datasets. The CIFAR-10 dataset contains 60,000 images of ten classes, where 59,000 images (5900 images sampled from each class) are randomly selected for training and the rest images are used for testing. The SUN397 dataset includes 108,753 scene images from 397 well-sampled categories, where each category contains at least 100 samples. In our experiments, the training set contains 106,953 images and the rest 1800 images are used for testing.

2) Cross-Domain Datasets: For the proposed GITR task, we test our ATH framework on three cross-domain benchmarks, including Office Home [52], Caltech&Amazon [53], and VisDA [54]. Specifically, for HoSDR and HoCDR subtasks, we follow [17] and [18] and conduct experiments on Office Home, which is a well-known cross-domain benchmark containing four different domains: 1) Artistic (A); 2) Clip Art (C); 3) Product (P); and 4) Real World (R). For simplicity, we use # → & to indicate that the source domain is # and the target domain is &. For HeSDR and HeCDR subtasks, we conduct experiments on Caltech&Amazon and VisDA benchmarks. Caltech&Amazon is constructed from Office-Caltech10 dataset [53], which includes images of ten classes from different domains. VisDA consists of a synthetic domain and two real-image domains containing the same 12 object categories. The training set of VisDA contains synthetic images generated from 3-D CAD models and the validation set includes real images selected from COCO dataset [55]. In our experiments, we set the training set of VisDA as source domain and validation set as target domain. For each domain of VisDA, we randomly select 500 images from each class for training. For unsupervised methods, all samples in the source and target domains are used for training. For semisupervised methods, since there are no label samples in the target domain, we only use the samples in the source domain for training. Note that, the target domain samples can be used for training by our proposed ATH_M method.

B. Experimental Setup

1) Feature Extraction: For CIFAR-10 dataset, each image is transformed to a 512-D GIST [56] feature vector. For SUN397 dataset, each image is reshaped to a 1600-D feature vector. For Office Home, we follow [17], [18] and each image is represented by a 4096-D feature extracted by VGG-16 network [21], thereby providing a homogeneous feature space. For Caltech&Amazon, Amazon utilizes SURF features and each image is encoded by 800-bin histograms, while Caltech adopts deep features and each image is encoded by a 4096-D CNN feature vector [57]. For VisDA, the source images are extracted by ResNet-50 network [22] (obtaining 2048-D features) while the target images are extracted by VGG-16 network [21] (obtaining 4096-D features).

2) Dimensionality Reduction: For HeSDR and HeCDR subtasks, the known hashing methods cannot be directly applied since the feature dimensions of source and target domains are misaligned. For comparison, we first perform dimensionality reduction and then take the aligned features...
as the inputs of comparison methods. Specifically, if the feature dimension of source domain is larger than that of target domain, PCA [58] dimensionality reduction is performed on source domain, and the number of reduced features is determined by the feature dimension of target domain. Otherwise, we perform dimensionality reduction on target domain. Note that, our methods do not require such a feature preprocessing, thus avoiding information loss caused by dimensionality reduction.

3) Implementation Details: The proposed ATH is implemented in MATLAB on a standard PC with Intel 2.20-GHz CPU and 128-GB RAM. For the comparison methods, we use their public codes and the parameter selection observes from the corresponding papers. Following the experimental settings in [17] and [18], for each dataset, we pick up 500 samples from the target domain randomly as the query samples, and the rest images are used for training. The length of hash codes is selected from the range of {16, 32, 48, 64, 96, 128}. For each method, we simply set the number of nearest neighbors \( \eta_k \in \{\eta_s, \eta_t\} \) to 10 and the number of iterations \( \text{Ite} \) is set to 10. For the parameters \( \alpha_s, \alpha_t, \beta_s, \beta_t, \) and \( \lambda \), we set them to \( 10^{-2}, 10^{-1}, 10^{-3}, 10^{-1}, \) and 1 for ATH_U, \( 10^{-1}, 10^{-2}, 10^{0}, 10^{-2}, \) and 1 for ATH_M, \( 10^{-2}, 1, 10^{-3}, 10^{-3}, \) and \( 10^{-2} \) for ATH_K, and \( 10^{-3}, 10^{-1}, 10^{-2}, 10^{-2}, \) and \( 10^{-2} \) for ATH_S, respectively. To obtain stable results, we run ten independent experiments for each method and the average value of ten experiments are reported.

4) Evaluation Protocol: Following the settings in [17], we adopt the widely used mean average precision (MAP) as the evaluation metric and provide the precision-recall (PR) curves to evaluate the performance of different methods. Besides, we also provide feature visualizations to illustrate the performance differences between different methods.

C. Performance on Single-Domain Datasets

Table I provides the MAP results of unsupervised hashing methods on the single-domain benchmarks (the highest and second highest are marked in red and blue, respectively). As we can see, the similarity-preservation-based methods (i.e., DPLM, SHSR, and ATH_U) always achieve better performance than the projection-based (i.e., ITQ, SP, and GTH) methods in the single-domain setting, which stresses the importance to preserve the similarity relationship between pairwise samples. Moreover, although the proposed ATH_U is tailored for transfer retrieval, it can also outperform the traditional hashing methods on the single-domain benchmarks. The potential reason is that the proposed ATH_U can adaptively learn the similarity relationship between samples from the data during the training stage.
TABLE II
MAP (%) RESULTS OF DIFFERENT METHODS WITH VARYING CODE LENGTHS FROM 16 TO 128 ON THE HoSDR SUBTASK

| Method | 16 | 32 | 48 | 64 | 96 | 128 |
|--------|----|----|----|----|----|----|
| ITQ    | 14.40 | 23.31 | 28.25 | 30.88 | 33.18 | 34.21 |
| SP     | 18.10 | 32.82 | 37.11 | 38.33 | 39.99 | 40.22 |
| DPLM   | 24.54 | 36.40 | 40.52 | 42.73 | 47.06 | 47.41 |
| SISR   | 19.23 | 35.56 | 38.73 | 39.50 | 41.60 | 41.63 |
| GTH    | 17.48 | 25.81 | 28.61 | 31.62 | 33.98 | 33.96 |
| ATH_U  | 37.86 | 46.54 | 48.50 | 48.65 | 48.70 | 48.43 |

TABLE III
MAP (%) RESULTS OF DIFFERENT METHODS WITH VARYING CODE LENGTHS FROM 16 TO 128 ON THE HoCDR SUBTASK

| Method | 16 | 32 | 48 | 64 | 96 | 128 |
|--------|----|----|----|----|----|----|
| ITQ    | 6.16 | 10.03 | 12.07 | 13.16 | 14.09 | 14.89 |
| SP     | 7.49 | 14.04 | 16.07 | 17.09 | 17.72 | 17.98 |
| DPLM   | 10.53 | 15.76 | 18.68 | 19.94 | 21.95 | 22.00 |
| SISR   | 13.22 | 17.87 | 18.68 | 18.60 | 18.35 | 16.55 |
| GTH    | 10.15 | 10.09 | 12.08 | 13.10 | 14.31 | 14.88 |
| ATH_U  | 17.00 | 23.76 | 25.68 | 27.05 | 27.21 | 26.49 |

TABLE IV
MAP (%) RESULTS OF DIFFERENT METHODS WITH VARYING CODE LENGTHS FROM 16 TO 128 ON THE HeSDR SUBTASK

| Method | 16 | 32 | 48 | 64 | 96 | 128 |
|--------|----|----|----|----|----|----|
| ITQ    | 21.40 | 25.02 | 25.95 | 26.60 | 27.04 | 27.51 |
| SP     | 22.01 | 27.58 | 29.15 | 28.73 | 29.16 | 29.14 |
| DPLM   | 17.92 | 21.60 | 21.44 | 23.79 | 23.30 | 24.15 |
| SISR   | 19.73 | 22.35 | 22.70 | 22.33 | 24.35 | 24.27 |
| GTH    | 22.28 | 23.44 | 25.27 | 26.00 | 26.52 | 26.93 |
| ATH_U  | 23.87 | 25.67 | 27.42 | 28.36 | 28.30 | 28.22 |

TABLE V
MAP (%) RESULTS OF DIFFERENT METHODS WITH VARYING CODE LENGTHS FROM 16 TO 128 ON THE HeCDR SUBTASK

| Method | 16 | 32 | 48 | 64 | 96 | 128 |
|--------|----|----|----|----|----|----|
| ITQ    | 9.76 | 10.10 | 9.86 | 9.92 | 9.78 | 9.79 |
| SP     | 9.93 | 9.72 | 9.64 | 9.60 | 9.79 | 9.78 |
| DPLM   | 9.61 | 9.99 | 9.91 | 9.61 | 10.14 | 9.88 |
| SISR   | 9.76 | 9.96 | 9.62 | 9.98 | 9.91 | 9.91 |
| GTH    | 9.70 | 9.65 | 9.71 | 9.74 | 9.67 | 9.61 |
| ATH_U  | 12.21 | 13.14 | 13.84 | 13.83 | 13.02 | 12.08 |

D. Performance on Cross-Domain Datasets

Tables II–V list the MAP results of unsupervised methods under four different GITR subtasks. Figs. 3–6 provide the PR curves (with 64 bits) of all unsupervised and supervised methods under different GITR subtasks. As we can see, the proposed ATH_U obtains the best performance under different settings (i.e., HoSDR, HoCDR, HeSDR, and HeCDR) in most cases, which demonstrates the effectiveness and versatility of our ATH framework for different GITR subtasks. Moreover, we observe that the compared unsupervised methods are as effective as the proposed ATH_U on HeSDR subtask, but their performances are worse than that of ATH_U on HeCDR subtask. This is because the features and domains of query and retrieval sets in the HeCDR subtask are misaligned and different. By simultaneously learning asymmetric hash functions and the affinity relationship of cross-domain data, our method can perform knowledge transfer between domains and avoid information loss caused by feature alignment.

As we can see from Table IV, some results of DPLM are better than those of ATH_U in the HeSDR task. The potential reason is that DPLM adopts a discrete proximal linearized minimization approach to handle the discrete constraints during the learning process [46], which makes the learned binary codes more effective. Since discreteness is not the main focus of this manuscript, the proposed ATH_U obtains binary codes by simply utilizing a relaxation strategy (i.e., the real-valued embedding is first learned, which are then thresholded to be binary codes), which may cause quantization error to some extent. However, as can be seen from Table V, when cross-domain information interaction rather than discreteness becomes the core issue in the HeCDR task, the proposed ATH_U achieves better performance than DPLM.

In addition, Table VI lists the MAP results of supervised methods under different GITR subtasks and Table VII provides the MAP results of different methods in the semisupervised setting. Since the source code of DHLing is not publicly available, we directly cite the results of DHLing with 64 bits using the same experimental settings with ours. From the experimental results, we observe that our ATH_M, ATH_S, and ATH_K method obtain better retrieval performance than the

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
state-of-the-art hashing methods. The reason is that our methods can perform positive transfer during training, as illustrated in the next section.

Table VI

| Method | HoSDR | HeSDR |
|--------|-------|-------|
|        | P→A  | R→C  | P→C  | C→A  | A→P  | C→P  |
| KSH    | 64.33 | 44.83 | 41.63 | 49.59 | 51.78 | 58.39 |
| SDH    | 62.45 | 50.81 | 47.32 | 55.04 | 44.20 | 66.33 |
| FSCH   | 66.61 | 55.66 | 51.83 | 59.14 | 73.27 | 70.85 |
| SCDH   | 67.23 | 65.23 | 57.37 | 67.73 | 81.36 | 81.54 |
| ATH_K  | 77.33 | 68.46 | 64.76 | 62.10 | 90.23 | 83.10 |
| ATH_S  | 77.40 | 69.08 | 67.72 | 65.99 | 91.32 | 86.65 |

Table VII

| Method | HoCDR | HeCDR |
|--------|-------|-------|
|        | P→R  | R→P  | C→R  | A→R  | R→A  |
| KSH    | 32.02 | 34.42 | 21.56 | 18.51 | 23.87 | 20.04 |
| SDH    | 25.75 | 27.90 | 15.97 | 16.72 | 32.06 | 22.79 |
| PWCP   | 34.03 | 34.44 | 24.22 | 18.42 | 34.57 | 28.95 |
| DHLing | 48.47 | 45.24 | 30.81 | 25.15 | 43.30 | 38.68 |
| ATH_M  | 50.69 | 52.16 | 33.79 | 29.14 | 50.97 | 42.80 |

Fig. 7. Accuracy and convergence curves on different datasets. (a) Office Home. (b) VisDA.

E. Positive Transfer or Not

We take ATH_M as an example to study whether the proposed ATH framework can achieve positive transfer or not. First, we use a KNN model (k = 1) trained on a labeled source domain to classify the unlabeled target domain data. In each iteration ite, the pseudo-labels \( \bar{Y}_{ite} \) of target domain samples are obtained by

\[
\bar{Y}_{ite} = \text{KNN}\left( A_t^{the} X_t, A_s^{the} X_s, Y_s \right).
\]  

Fig. 8. Parameter sensitivity on A → C task. (a) \( \alpha_t \) versus \( \alpha_s \). (b) \( \beta_t \) versus \( \beta_s \).
For initial $\tilde{Y}_t^1$, we set
\begin{equation}
\tilde{Y}_t^1 = \begin{cases} 
\text{KNN}(X_t, X_s, Y_s), & \text{if homogeneous} \\
\text{KNN}(A_{t,\text{pca}} X_t, A_{s,\text{pca}} X_s, Y_s), & \text{otherwise}
\end{cases}
\end{equation}

where $A_{t,\text{pca}}$ and $A_{s,\text{pca}}$ are initialized by PCA projections. Then, in each iteration $i$, the classification accuracy $\text{Accuracy}_{i}$ with respect to pseudo-labels $Y_{i,\text{ite}}$ on target domain can be calculated by
\begin{equation}
\text{Accuracy}_{i} = \frac{1}{n_t} \sum_{i=1}^{n_t} \mathcal{I}(Y_{i,\text{ite}}, Y_{i,j})
\end{equation}

where $Y_{i,j}$ is the $j$th column of $Y_i$, $\mathcal{I}(a, b)$ is an indicator function, if vector $a$ equals to $b$, $\mathcal{I}(a, b) = 1$, otherwise, $\mathcal{I}(a, b) = 0$. Fig. 7 shows the accuracy and convergence curves of the proposed ATH_M on homogeneous (A $\rightarrow$ P) and heterogeneous (VisDA train $\rightarrow$ validation) tasks. As we can see, our designed optimization algorithm converges fast. Moreover, with the increase of iteration times, the accuracy gradually increases and finally reaches stability, which demonstrates the validity of the mapping space learned by the proposed ATH_M. Therefore, our ATH_M can achieve positive transfer during the training process, which is a potential reason that why our methods outperform the other comparison methods on different GITR subtasks.

F. Parameter Sensitivity

Fig. 8 shows the parameter sensitivity of the proposed ATH_S on Office Home A $\rightarrow$ C task. In this experiment, we fix $\lambda = 10^{-2}$ and $r = 16$. From Fig. 8, we observe that the performance of ATH_S is relatively stable to $\beta_k$ in the range of $[10^{-6}, 10^{-1}]$. However, $\alpha_k$ affects the MAP scores of ATH_S to a great extent. Specifically, ATH_S can work well when $\alpha_t$ and $\alpha_s$ are in the ranges of $[10^{-3}, 10^{-1}]$ and $[10^{-2}, 10^0]$, respectively.

G. Ablation Study

Table VIII lists ablation experimental results of the proposed ATH_U. Obviously, ATH_U obtains better performance under different GITR subtasks, which indicates the effectiveness of each component of ATH_U. Moreover, without preserving domain structure (i.e., $T_3$), the MAP values of ATH_U will decrease significantly in most cases, which demonstrates the necessity of maintaining the intrinsic geometrical structure of data in their respective domains.

H. Visualization

Fig. 9 shows t-SNE visualizations [59] learned by GTH, SDH, FSDH, ATH_U, ATH_K, and ATH_S on heterogeneous Caltech $\rightarrow$ Amazon task, where the hexagon and circle points indicate the projected data from source and target domains, respectively. As we can see, without semantic information, GTH and ATH_U cannot obtain distinct class structure. For SDH and FSDH, due to feature heterogeneity of data and information loss caused by dimensionality reduction, massive points of different classes are mixed together and hard to distinguish. More importantly, our ATH_K and ATH_S can make the source data of each class close to the target data of the same class, and far away from the data of different classes in both two domains.
We have conducted additional experiments to show whether the validity of the proposed algorithm depends on the size of the training samples. Specifically, we conducted the experiments on the SUN397 dataset. In the experiments, we changed the number of training samples and calculated the MAP results on the test set. Fig. 10 shows the curve of MAP results (16 bits) changing with the number of training samples. It is easy to see that our algorithm can achieve good performance with only about 30% of the data ($35 \times 10^3$). Moreover, when the number of training samples increases to a certain extent, the performance of our method tends to be stable.

VI. CONCLUSION

This article presents a more difficult image transfer retrieval problem called GITR, which makes no restriction on the domains and feature spaces of query and retrieval sets. To solve the GITR problem, we propose an ATH framework with its unsupervised/semisupervised/supervised realizations, based on asymmetric hashing strategy and adaptive bipartite graph learning. Extensive experiments conducted on different GITR subtasks and label settings demonstrate the effectiveness of our ATH framework. A potential limitation of the proposed ATH is that the class space of target domain needs to be aligned with that of source domain. For the class space misaligned problem, a possible solution is to integrate a zero-shot learning approach into the proposed model, which is a potential research direction for our future work.

REFERENCES

[1] J. Wang, T. Zhang, J. Song, N. Sebe, and H. T. Shen, “A survey on learning to hash,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 40, no. 4, pp. 769–790, Apr. 2018.
[2] X. Zhou et al., “Graph convolutional network hashing,” IEEE Trans. Cybern., vol. 50, no. 4, pp. 1460–1472, Apr. 2020.
[3] S. He et al., “Bidirectional discrete matrix factorization hashing for image search,” IEEE Trans. Cybern., vol. 50, no. 9, pp. 4157–4168, Sep. 2020.
[4] L. Liu, M. Yu, and L. Shao, “Unsupervised local feature hashing for image similarity search,” IEEE Trans. Cybern., vol. 46, no. 11, pp. 2548–2558, Nov. 2016.
[5] S. Wang, C. Li, and H.-L. Shen, “Equivalent continuous formulation of general hashing problem,” IEEE Trans. Cybern., vol. 51, no. 8, pp. 4089–4099, Aug. 2021.
[6] W. W. Ng, S. Xu, J. Zhang, X. Tian, T. Rong, and S. Kwong, “Hashing-based undersampling ensemble for imbalanced pattern classification problems,” IEEE Trans. Cybern., vol. 52, no. 2, pp. 1269–1279, Feb. 2022.
[7] Y. Wang, Z. Chen, X. Luo, R. Li, and X.-S. Xu, “Fast cross-modal hashing with global and local similarity embedding,” IEEE Trans. Cybern., vol. 52, no. 10, pp. 10064–10077, Oct. 2022.
[8] B. Sun, S. Wang, D. Kong, L. Wang, and B. Yin, “Real-time human action recognition using locally aggregated kinematic-guided Skeletonlet and supervised hashing-by-analysis model,” IEEE Trans. Cybern., vol. 52, no. 6, pp. 4837–4849, Jun. 2022.
[9] C. Da et al., “Nonlinear asymmetric multi-valued hashing,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 41, no. 11, pp. 2660–2676, Nov. 2019.
[10] J. Lu, Z. Lai, J. Lin, Q. Lin, and J. Zhou, “Label self-adaptation hashing for image retrieval,” in Proc. 25th Int. Conf. Pattern Recognit., 2020, pp. 2256–2263.
[11] J. T. Zhou, H. Zhao, X. Peng, M. Fang, Z. Qin, and R. S. M. Goh, “Transfer hashing: From shallow to deep,” IEEE Trans. Neural Netw. Learn. Syst., vol. 29, no. 12, pp. 6191–6201, Dec. 2018.
[12] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang, “Domain adaptation via transfer component analysis,” IEEE Trans. Neural Netw., vol. 22, no. 2, pp. 199–210, Feb. 2011.
[13] M. Jing, J. Zhao, J. Li, L. Zhu, Y. Yang, and H. T. Shen, “Adaptive component embedding for domain adaptation,” IEEE Trans. Cybern., vol. 51, no. 7, pp. 3390–3403, Jul. 2021.
[14] B. H. Nguyen, B. Xue, P. Andreae, and M. Zhang, “A hybrid evolution- ary computation approach to inducing transfer classifiers for domain adaptation,” IEEE Trans. Cybern., vol. 51, no. 12, pp. 6319–6332, Dec. 2021.
[15] Z. Cao, M. Long, C. Huang, and J. Wang, “Transfer adversarial hashing for hamming space retrieval,” in Proc. AAAI Conf. Artif. Intell., 2018, pp. 6698–6705.
[16] L. Zhang, J. Liu, Y. Yang, F. Huang, F. Nie, and D. Zhang, “Optimal projection guided transfer hashing for image retrieval,” IEEE Trans. Circuits Syst. Video Technol., vol. 30, no. 10, pp. 3788–3802, Oct. 2020.
[17] F. Huang, L. Zhang, Y. Yang, and X. Zhou, “Probability weighted compact feature for domain adaptive retrieval,” in Proc. Int. Conf. Comput. Vis. Pattern Recognit., 2020, pp. 9579–9588.
[18] H. Xia, T. Jing, C. Chen, and Z. Ding, “Semi-supervised domain adaptive retrieval via discriminative hashing learning,” in Proc. ACM Int. Conf. Multimedia, 2021, pp. 3853–3861.
[19] N. I. Swain and D. H. Ballard, “Indexing via color histograms,” in Proc. Int. Conf. Comput. Vis., 1990, pp. 390–393.
[20] D. G. Lowe, “Distinctive image features from scale-invariant keypoints,” Int. J. Comput. Vis., vol. 60, no. 2, pp. 91–110, 2004.
[21] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” in Proc. Int. Conf. Learn. Represent., 2015, pp. 1–14.
[22] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proc. Int. Conf. Comput. Vis. Pattern Recognit., 2016, pp. 770–778.
[23] A. Gordo, F. Perronnin, Y. Gong, and S. Lazebnik, “Asymmetric distances for binary embeddings,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 36, no. 1, pp. 33–47, Jan. 2014.
[24] W. Dong, M. Charikar, and K. Li, “Asymmetric distance estimation with sketches for similarity search in high-dimensional spaces,” in Proc. Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval, 2008, pp. 123–130.
[25] B. Neyshabur, N. Srberbo, R. R. Salakhutdinov, Y. Makarychev, and P. Yudollahpour, “The power of asymmetry in binary hashing,” in Proc. Adv. Neural Inf. Process. Syst., 2013, pp. 2823–2831.
[26] Y. Luo, Y. Li, F. Shen, Y. Yang, P. Cui, and Z. Huang, “Collaborative learning for extremely low bit asymmetric hashing,” IEEE Trans. Knowl. Data Eng., vol. 33, no. 12, pp. 3675–3685, Dec. 2021.
[27] J. Lu, H. Wang, J. Zhou, M. Yan, and J. Wen, “Deep asymmetric hashing with dual semantic regression and class structure quantization,” Inf. Sci., vol. 589, pp. 235–249, Apr. 2022.
[28] F. Shen, X. Gao, L. Liu, Y. Yang, and H. T. Shen, “Deep asymmetric pairwise hashing,” in Proc. ACM Int. Conf. Multimedia, 2017, pp. 1522–1530.
[29] Q. Jiang and W. Li, “Asymmetric deep supervised hashing,” in Proc. AAAI Conf. Artif. Intell., 2018, pp. 3342–3349.
[30] A. Iscen, G. Tolias, Y. Avrithis, T. Furon, and O. Chum, “Efficient diffusion on region manifolds: Recovering small objects with compact CNN representations,” in Proc. Int. Conf. Comput. Vis. Pattern Recognit., 2017, pp. 926–935.
[31] Z. Lai, Y. Chen, J. Wu, W. K. Wong, and F. Shen, “Jointly sparse hashing for image retrieval,” IEEE Trans. Image Process., vol. 27, pp. 6147–6158, 2018.
[32] C. Liu et al., “Guided similarity separation for image retrieval,” in Proc. Adv. Neural Inf. Process. Syst., 2019, pp. 1554–1564.
[33] J. Lu, Z. Lai, H. Wang, Y. Chen, J. Zhou, and L. Shen, “Generalized embedding regression: A framework for supervised feature extraction,” IEEE Trans. Neural Netw. Learn. Syst., vol. 33, no. 1, pp. 185–199, Jan. 2022.

[34] C. Zhang and W. Zheng, “Semi-supervised multi-view discrete hashing for fast image search,” IEEE Trans. Image Process., vol. 26, pp. 2604–2617, 2017.

[35] Y. Chen, S. Wang, J. Lu, Z. Chen, Z. Zhang, and Z. Huang, “Local graph convolutional networks for cross-modal hashing,” in Proc. ACM Int. Conf. Multimedia, 2021, pp. 1921–1928.

[36] J. Lu, J. Lin, Z. Lai, H. Wang, and J. Zhou, “Target redirected regression with dynamic neighborhood structure,” Inf. Sci., vol. 544, pp. 564–584, Jan. 2021.

[37] C. Li, L. Lin, W. Zuo, J. Tang, and M. Yang, “Visual tracking via dynamic graph learning,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 41, no. 11, pp. 2770–2782, Nov. 2019.

[38] J. Lu, H. Wang, J. Zhou, Y. Chen, Z. Lai, and Q. Hu, “Low-rank adaptive graph embedding for unsupervised feature extraction,” Pattern Recognit., vol. 113, May 2021, Art. no. 107758.

[39] Y. Luo, Z. Huang, Z. Wang, Z. Zhang, and M. Baktashmotlagh, “Adversarial bipartite graph learning for video domain adaptation,” in Proc. ACM Int. Conf. Multimedia, 2020, pp. 19–27.

[40] J. Wen, Z. Zhang, Z. Zhang, L. Fei, and M. Wang, “Generalized incomplete multi-view clustering with flexible locality structure diffusion,” IEEE Trans. Cybern., vol. 51, no. 1, pp. 101–114, Jan. 2021.

[41] F. Shen, C. Chen, W. Liu, and H. Shen, “Supervised discrete hashing,” in Proc. Int. Conf. Comput. Vis. Pattern Recognit., 2015, pp. 37–45.

[42] W. Liu, J. Wang, S. Kumar, and S. Chang, “Hashing with graphs,” in Proc. Int. Conf. Mach. Learn., 2011, pp. 1–8.

[43] Y. Chen, Z. Lai, Y. Ding, K. Lin, and W. Wong, “Deep supervised hashing with anchor graph,” in Proc. Int. Conf. Comput. Vis., 2019, pp. 9795–9803.

[44] 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.

[45] Y. Xia, K. He, P. Kohli, and J. Sun, “Sparse projections for high-dimensional binary codes,” in Proc. Int. Conf. Comput. Vis. Pattern Recognit., 2015, pp. 3332–3339.

[46] 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 Process., vol. 25, pp. 5610–5621, 2016.

[47] D. Hu, F. Nie, and X. Li, “Discrete spectral hashing for efficient similarity retrieval,” IEEE Trans. Image Process., vol. 28, pp. 1080–1091, 2019.

[48] W. Liu, J. Wang, R. Ji, Y. Jiang, and S. Chang, “Supervised hashing with kernels,” in Proc. Int. Conf. Comput. Vis. Pattern Recognit., 2012, pp. 2074–2081.

[49] J. Gui, T. Liu, Z. Sun, D. Tao, and T. Tan, “Fast supervised discrete hashing,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 40, no. 2, pp. 490–496, Feb. 2018.

[50] Y. Chen, Z. Tian, H. Zhang, J. Wang, and D. Zhang, “Strongly constrained discrete hashing,” IEEE Trans. Image Process., vol. 29, pp. 3596–3611, 2020.

[51] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba, “SUN database: Large-scale scene recognition from abbey to zoo,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2010, pp. 3485–3492.

[52] H. Venkateswara, J. Eusebio, S. Chakraborty, and S. Panchanathan, “Deep hashing network for unsupervised domain adaptation,” in Proc. Int. Conf. Comput. Vis. Pattern Recognit., 2017, pp. 5385–5394.

[53] B. Gong, Y. Shi, F. Sha, and K. Grauman, “Geodesic flow kernel for unsupervised domain adaptation,” in Proc. Int. Conf. Comput. Vis. Pattern Recognit., 2012, pp. 2066–2073.

[54] X. Peng, B. Usman, K. Saito, N. Kaushik, J. Hoffman, and K. Saenko, “Syn2Real: A new benchmark forsythetic-to-real visual domain adaptation,” 2018. arXiv:1806.09755.

[55] T. Lin et al., “Microsoft COCO: Common objects in context,” in Proc. Int. Conf. Comput. Vis. Pattern Recognit., 2014, pp. 740–755.

[56] 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, 1991.

[57] J. Donahue et al., “DeCAF: A deep convolutional activation feature for generic visual recognition,” in Proc. Int. Conf. Mach. Learn., 2014, pp. 687–695.

[58] M. A. Turk and A. Pentland, “Face recognition using eigenfaces,” in Proc. Int. Conf. Comput. Vis. Pattern Recognit., 1991, pp. 586–591.

[59] V. D. M. Laurens and G. Hinton, “Visualizing data using t-SNE,” J. Mach. Learn. Res., vol. 9, no. 2605, pp. 2579–2605, 2008.