Exploring Hierarchical Information in Hyperbolic Space for Self-Supervised Image Hashing

Rukai Wei, Yu Liu, *Member, IEEE*, Jingkuan Song, *Senior Member, IEEE*, Yanzhao Xie, and Ke Zhou, *Member, IEEE*

***Abstract*— In real-world datasets, visually related images often form clusters, and these clusters can be further grouped into larger categories with more general semantics. These inherent hierarchical structures can help capture the underlying distribution of data, making it easier to learn robust hash codes that lead to better retrieval performance. However, existing methods fail to make use of this hierarchical information, which in turn prevents the accurate preservation of relationships between data points in the learned hash codes, resulting in suboptimal performance. In this paper, our focus is on applying visual hierarchical information to self-supervised hash learning and addressing three key challenges, including the construction, embedding, and exploitation of visual hierarchies. We propose a new self-supervised hashing method named Hierarchical Hyperbolic Contrastive Hashing (HHCH), making breakthroughs in three aspects. First, we propose to embed continuous hash codes into hyperbolic space for accurate semantic expression since embedding hierarchies in the hyperbolic space generates less distortion than in the hyper-sphere or Euclidean space. Second, we update the K-Means algorithm to make it run in the hyperbolic space. The proposed *hierarchical hyperbolic K-Means* algorithm can achieve the adaptive construction of hierarchical semantic structures. Last but not least, to exploit the hierarchical semantic structures in hyperbolic space, we propose the hierarchical contrastive learning algorithm, including *hierarchical instance-wise* and *hierarchical prototype-wise* contrastive learning. Extensive experiments on four benchmark datasets demonstrate that the proposed method outperforms state-of-the-art self-supervised hashing methods. Our codes are released at https://github.com/HUST-IDSM-AI/HHCH.git.**

***Index Terms*— Self-supervised hashing, contrastive learning, hyperbolic embedding, hierarchical semantic structure, K-Means.**

I. Introduction

THE explosive growth of multimedia data poses a huge challenge to large-scale web search engines. Hashing-based methods [1], [2], [3], [4], [5] which convert high-dimensional features to compact binary hash codes while preserving the original similarity information in Hamming space, have become the dominant solution due to their high computation efficiency and low storage cost. Recently, self-supervised hashing methods [6], [7], [8] have attracted increasing attention since they do not rely on expensive handcrafted labels and are productive for real-world retrieval tasks.

However, employing binary hash codes for efficient image retrieval poses two challenges: 1) The conversion of images to binary codes results in a significant loss of information. Representing the semantics of an image accurately becomes challenging when limited bits are used in binary codes. 2) The relationship learning for binary codes will bring suboptimal performance using measure patterns used in classifications since the granularity of similarity hashing measurement is more delicate than that of classification one. Existing state-of-the-art self-supervised hashing methods do their best to alleviate these issues. They can be roughly divided into two categories, *i.e.*, reconstruction-based methods [9], [10], [11], [12] and contrastive-learning-based methods [13], [14], [15]. Reconstruction-based methods aim to preserve more information about original images in hash codes through the reconstruction of original images using variational autoencoders (VAEs) or generative adversarial networks (GANs). Nevertheless, this method usually suffers from introducing background noise into hash codes [14], [16]. Contrastive-learning-based methods can address this issue by maximizing the similarity between different views of an image. Although state-of-the-art contrastive learning-based methods [13], [14], [15] have demonstrated significant performance improvements compared to reconstruction-based methods, these approaches overlook the latent visual hierarchical information, hindering the model's ability to capture precise relationships between samples. Moreover, they map the relationships in either the Euclidean space or the hyper-sphere space, resulting in significant information loss caused by limited space expressiveness. Inspired by the following two key insights, we decided to address the above issues by enhancing contrastive hashing.

A. Exploring Visual Hierarchical Information

The hierarchical semantic structure, as an inherent property of image datasets, can significantly aid in capturing the latent data distribution. As shown in Fig. 1, several visually related images are grouped into the cluster named $Husky$, and the semantically relevant clusters are further integrated into the larger taxon named Dog. We can always

Manuscript received 22 September 2023; revised 23 November 2023 and 17 January 2024; accepted 5 February 2024. Date of publication 5 March 2024; date of current version 8 March 2024. This work was supported in part by the National Key Research and Development Program under Grant 2023YFB4502701; in part by the National Natural Science Foundation of China under Grant 62232007, Grant 61821003, and Grant 61902135; and in part by the Natural Science Foundation of Hubei Province under Grant 2022CFB060. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Leyuan Fang. *(Corresponding author: Yu Liu.)*

Rukai Wei, Yanzhao Xie, and Ke Zhou are with Wuhan National Laboratory for Optoelectronics, Huazhong University of Science and Technology, Wuhan 430000, China (e-mail: weirukai@hust.edu.cn; yzxie@hust.edu.cn; zhke@hust.edu.cn).

Yu Liu is with the School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430000, China (e-mail: liu_yu@hust.edu.cn).

Jingkuan Song is with the School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China (e-mail: jingkuan.song@gmail.com).

Digital Object Identifier 10.1109/TIP.2024.3371358

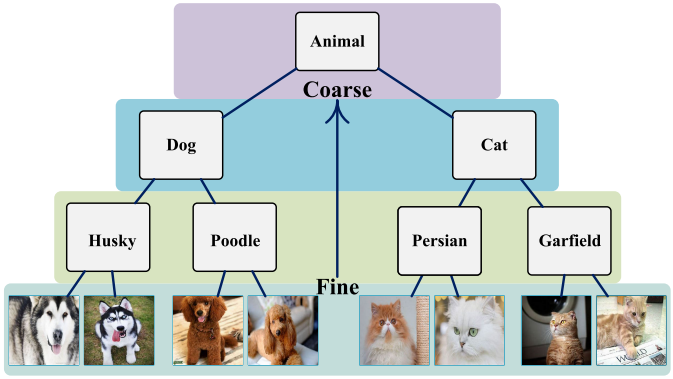

Fig. 1. An illustration of a hierarchical semantic structure. Semantic hierarchy is an inherent property of real-world image datasets, *e.g.*, "image instance → husky → dog → animal" in the order from fine-grained to coarse-grained semantics.

obtain tree-like hierarchical structures with increasingly coarse semantic granularity from bottom to top. The hierarchical information can serve as valuable prior information for the hashing model, enabling the generation of codes that effectively preserve data-data relationships. This presents a great opportunity to improve the performance of similarity-based retrieval. Recently, Lin et al. [15] used homology relationships over a two-layer semantic structure to learn hash codes. While using the two-layer structure yields excellent results, there is still potential for enhancing the construction and utilization of hierarchical semantic structures to obtain more precise cross-layer affiliation and cross-sample similarity.

B. Learning in the Hyperbolic Space

Different metric spaces can result in notable performance variations due to inherent dissimilarities in the employed embedding techniques and similarity computation methods. Recent studies [17], [18], [19] have elucidated the limitations of embedding hierarchies into the Euclidean or hyper-sphere space. These spaces often fail to preserve complex interrelationships and dependencies between different levels and branches in visual hierarchies, leading to information distortion and sub-optimal solutions. In contrast, the hyperbolic space exhibits exponential volume growth concerning the radius [17], [19], [20], distinguishing it from Euclidean spaces characterized by polynomial growth. This distinction arises from the scaling of local distances in hyperbolic spaces, which follows the conformal factor and approaches infinity near the boundary of the ball. This property enables us to utilize low-dimensional manifolds for embedding hierarchies without compromising the model's representation power [17], [19], resulting in less information distortion.

Based on these insights, we propose constructing hierarchical structures and learning hash codes in the hyperbolic space (*e.g.*, the Poincaré ball). To facilitate the construction of hierarchical semantic structures in this space, we designed the *hierarchical hyperbolic K-Means* algorithm. The algorithm performs bottom-up clustering with instances over the bottom layer and with prototypes (hash centers [1]) over other layers by the hyperbolic K-Means algorithm. Compared to the original K-Means [21], the hyperbolic K-Means algorithm incorporates a new hash center calculation and distance measure (See § III-C). In addition, referring to [22], we propose a hierarchical contrastive learning framework for hashing, including *hierarchical instance-wise* contrastive learning and *hierarchical prototype-wise* contrastive learning (See § III-D). The former leverages hierarchical semantic structures to mine accurate cross-sample similarity, reducing the number of *false-negatives* [23], [24] to improve the model's discriminating ability, where false-negatives are training samples that are incorrectly classified as negative by the model during the training phase, even though they have the same semantics as the anchor sample. The latter aligns the hash codes of image instances with the corresponding prototypes over different layers, mining accurate cross-layer affiliation.

As a result, we propose a novel self-supervised hashing method called **H**ierarchical **H**yperbolic **C**ontrastive **H**ashing (HHCH). In the HHCH framework, we learn continuous hash codes and embed them into the hyperbolic space with a projection head (See § III-B). Meanwhile, we perform hierarchical hyperbolic K-Means over the hyperbolic embeddings to construct hierarchical semantic structures before each training epoch. For hash learning within a mini-batch, we employ the proposed hierarchical contrastive learning framework to exploit the captured hierarchies. Finally, we conducted extensive experiments on four benchmark datasets to verify the superiority of HHCH compared with several state-of-the-art self-supervised hashing methods. The experimental results demonstrate that learning hash codes through the construction, embedding, and exploitation of hierarchical semantic structures in the hyperbolic space can significantly improve retrieval performance.

Our research primarily investigates the discovery of inherent visual hierarchical semantic structures in the hyperbolic space, intending to enhance the precision of both cross-sample similarity and cross-layer affiliation in self-supervised hashing. Our contributions are listed below.

- We introduce a novel metric space for hierarchical hash learning, *i.e.*, the hyperbolic space, which proposes to project continuous hash codes into the Poincaré ball for low information distortion.
- We successfully explore latent visual hierarchies in the hyperbolic space with the proposed hierarchical hyperbolic K-Means algorithm, facilitating the model to learn robust hash codes.
- We propose hierarchical prototype-wise contrastive learning and hierarchical instance-wise contrastive learning to take advantage of the captured hierarchies, which help to learn hash codes with superior discriminative performance.

II. Related Work

Deep learning is driving the development of major fields such as image classification [25], [26], [27], image segmentation [28], [29], [30], [31], natural language processing [32], etc. In this paper, we primarily concentrate on using deep learning technology for hashing, and our work is mainly related to the following three research areas.

A. Self-Supervised Hashing

Existing unsupervised hashing methods mainly fall into two lines: reconstruction-based hashing methods and

contrastive hashing methods. The former [9], [10] mostly adopts an encoder-decoder architecture [33] to reconstruct original images from hash codes or to reconstruct pairwise similarity derived from original features in Hamming space, while others employ generative adversarial networks to maximize reconstruction likelihood via the discriminator [12], [34], [35]. The latter can learn distortion-invariant hash codes, alleviating the problem of background noise caused by the reconstruction process and yielding state-of-the-art performance. Specifically, DATE [13] proposes a general distribution-based metric to depict the pairwise distance between images, exploring both semantic-preserving learning and contrastive learning to obtain high-quality hash codes. CIBHash [14] learns hash codes under the SimCLR [36] framework and compresses the model by the Information Bottleneck. Despite their contributions to learning compact hash codes in an unsupervised manner, they overlook rich information from the hierarchical semantic structures inherent to the datasets. Furthermore, our model differs from existing works [37], [38] focusing on the labeling hierarchy in supervised hashing and instead exploring the visual hierarchy in the unsupervised hashing framework. DSCH [15] is aware of hierarchical semantics and tries to exploit them using homology relationships mined by its two-step iterative algorithm. There is still room for the exploration of hierarchical semantics. 1) The customized two-layer hierarchical structure can only represent limited hierarchical information. It lacks an effective learning mechanism to adaptively construct hierarchical structures and provide accurate cross-sample and cross-layer information. 2) Embedding the hierarchical semantic structures in hyper-sphere space is not the optimal solution due to information distortion [19], [20].

B. Hyperbolic Embedding

Hyperbolic embedding technology has been successfully applied to CV [18], [19], [39], [40] and NLP [17], [41] tasks due to the distinctive property of hyperbolic space, *i.e.*, the exponential volume growth concerning the radius rather than the polynomial growth in the Euclidean space. In particular, hyperbolic embedding has shown promising results in modeling relational data, such as graphs, which often exhibit complex hierarchical structures. Researchers have proposed various hyperbolic graph embedding models, such as [42] and [43], which leverage the distinctive properties of hyperbolic space to model hierarchical relationships between nodes in a graph. For readers who are interested in learning more about hyperbolic embedding technology, we recommend taking a look at the seminal paper by Nickel and Kiela [17] and a survey [20]. Our proposed HHCH is the first study that explores the hyperbolic space for embedding visual hierarchies to generate high-quality hash codes.

C. Contrastive Learning

Contrastive learning focuses on view-invariant representations by attracting positive samples and repelling negative samples. The *instance-wise contrastive learning* methods [36], [44], [45] maximize the identical representation between views augmented from the same instance. They highlight data-data correlations and neglect the global distribution of the whole dataset. To compensate for this, the *prototype-wise contrastive learning* methods [46], [47] explicitly exploit the semantic structure and learn the prototypes (*i.e.*, the centers) of each cluster formed by semantically similar instances. HHCH benefits from both kinds of contrastive learning methods as well as recent efforts in hierarchical representation learning [22], [48].

1) Improvements: Our proposed HHCH leverages the advancements made by existing contrastive learning-based hashing methods, such as CIBHash [14], DATE [13], and DSCH [15], particularly in the design of the hash model architecture. Furthermore, it is worth noting that all of these models derive inspiration from the fundamental SimCLR model [36] for shaping both the contrastive learning framework and the learning objectives. However, embedding images into binary codes results in significant information loss, as binary code cannot accurately represent the complete semantics of an image or the relationships between the original images in any way. To address these issues, HHCH makes improvements to explore the latent visual hierarchical relationships in assistance with the model to accurately capture data relationships for robust hash learning. Taking a step further, we introduce hyperbolic embedding technology following [17] to embed continuous hash codes into Poincaré ball and propose the hierarchical hyperbolic K-Means algorithm to mine the visual hierarchies adaptively. As a result, HHCH can generate hash codes with less informant distortion for superior retrieval performance.

III. Methodology

A. Problem Definition and Overview

Given a training set $\mathcal{X} = \{x_i\}_{i=1}^{N}$ of N unlabeled images, we aim to learn a nonlinear hash function $f_{\theta_h} : x \rightarrow h \in \{-1, 1\}^K$ that maps the data from input space $\mathbb{R}^D$ to K-bit Hamming space. The goal is to minimize the Hamming distance between hash codes of similar data and maximize the Hamming distance between hash codes of dissimilar data. This allows us to utilize hash codes for efficient binary retrieval. The framework of HHCH is shown in Fig. 2, and the learning procedure and details are shown in Algorithm 1.

In the training phase, given an image x_i, our proposed f_{θ_h} extracts the feature vector using the VGG-19 model [49] (for far comparisons with baselines) and generates the continuous hash code h_i through the hash layer. Then, h_i is projected to z_i, *i.e.*, embedding the hash code into hyperbolic space (*i.e.*, the Poincaré ball), with a projection head $Exp\_map_{\theta_e}$. The projection head contains a fully-connected layer followed by the exponential mapping function in Equation (3) as shown in Fig. 2. Before each training epoch, we generate the hyperbolic embeddings $Z = \{z_i\}_{i=1}^{N}$ of all training images and perform *hierarchical hyperbolic K-Means* to construct hierarchical semantic structures. For the hash learning within the b-th mini-batch, we transform every image into two views with various augmentation strategies. Then, we acquire the corresponding hash codes $H_b^1 = \{h_i^1\}_{i=1}^{B}$ and $H_b^2 = \{h_i^2\}_{i=1}^{B}$ as well as the hyperbolic embeddings $Z_b^1 = \{z_i^1\}_{i=1}^{B}$ and $Z_b^2 = \{z_i^1\}_{i=1}^{B}$, where B denotes the batch size. Finally,

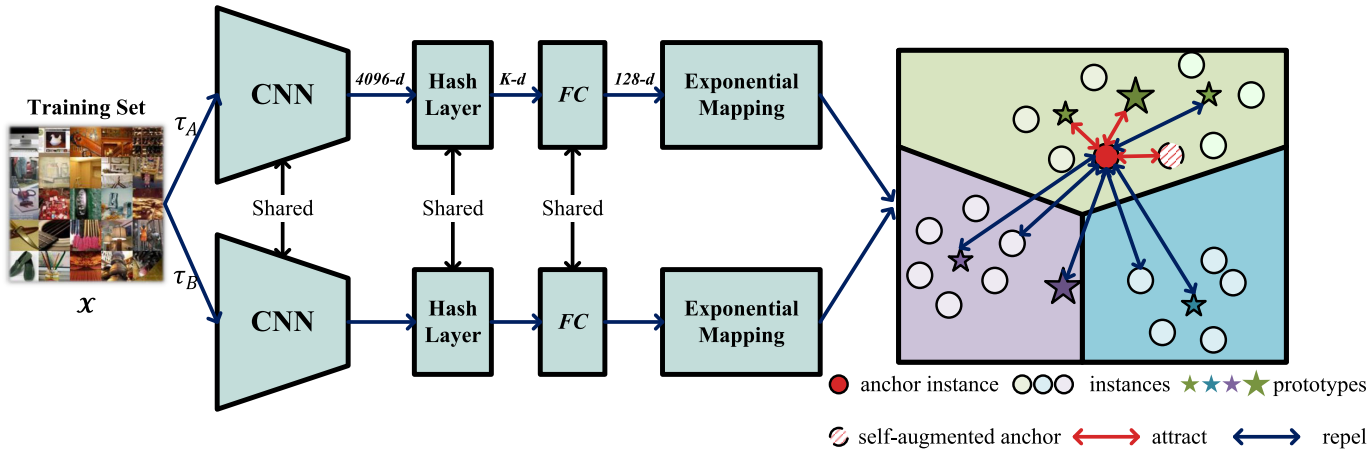

Fig. 2. The framework of HHCH. τ_A and τ_B are augmentations used to transform an image. Each transformed view is encoded by the CNN network and transformed into a K-bit continuous hash code. Then, the code is projected into hyperbolic space by a projection head consisting of a fully connected layer and an exponential mapping function. HHCH captures the hierarchical semantic structures before each training epoch and conducts hierarchical contrastive learning to exploit hierarchical information in the hyperbolic space. In hierarchical contrastive learning, the anchor sample will contrast with both instances and prototypes at different layers. Note that in the above figure, we utilize different sizes to represent prototypes at various layers where distinct colors are used to indicate different clusters.

Algorithm 1 HHCH Algorithm

Input: Training data $\mathcal{X}$; batch size B; Hyper-parameters; Training epochs T; Hash function f_{θ_h} and projection head $Exp\_map_{\theta_e}$.
Output: Optimized hash function f_{θ_h}

Initialize $\boldsymbol{\theta_h}$ and $\boldsymbol{\theta_e}$ randomly;
for t=*1 to* T **do**
  $Z = Exp\_map_{\theta_e}(f_{\theta_h}(\mathcal{X}))$;
  /* Construct the hierarchical semantic structures and output prototypes $\mathcal{P}$ and connections $\mathcal{E}$. */
  $(\mathcal{P}, \mathcal{E}) \leftarrow$ Hierarchical hyperbolic K-Means(Z);
  for $b = 1$ *to* $\frac{N}{B}$ **do**
    $X_b^1, X_b^2 \leftarrow transformation(X_b)$
    H_b^1, H_b^2=$f_{\theta_h}(X_b^1)$, $f_{\theta_h}(X_b^2)$
    Z_b^1, Z_b^2=$Exp\_map_{\theta_e}(H_b^1)$, $Exp\_map_{\theta_e}(H_b^2)$;
    Compute L_{H-inst} and $L_{H-proto}$ with $(Z_b^1, Z_b^2, \mathcal{P}, \mathcal{E})$;
    Compute L_Q with (H_b^1, H_b^2);
    Compute $\mathcal{L}$ with Equation (16);
    Update $\boldsymbol{\theta_h}$ and $\boldsymbol{\theta_e}$ with the Adam optimizer.

we compute the hierarchical contrastive loss, including the hierarchical instance-wise contrastive loss and the hierarchical prototype-wise contrastive loss, with the hyperbolic embeddings and the captured hierarchies $(\mathcal{P}, \mathcal{E})$, where $\mathcal{P}$ is the set of prototypes and $\mathcal{E}$ is the set of connections consisting of prototype-prototype and prototype-instance. In addition, we incorporate the quantization loss to reduce the quantization error.

In the test phase, we only use the well-trained hash function f_{θ_h}, disabling the projection head and the hierarchical semantic structures. All the continuous hash codes will be constrained to $\{-1, 1\}^K$ by the sgn function for performance evaluation, where K denotes the length of the hash code.

In the subsequent subsections, we will elaborate on HHCH from three perspectives: embedding hierarchies in the hyperbolic space, constructing hierarchies through hierarchical hyperbolic K-Means, and exploiting hierarchies via hierarchical contrastive learning.

B. Hyperbolic Space Learning

Formally, n-dimensional hyperbolic space $\mathbb{H}^n$ is a Riemannian manifold of constant negative curvature rather than the constant positive curvature in the Euclidean space. There exist several isomorphic models of hyperbolic space, we specialize in the Poincaré ball model $(\mathbb{D}_c^n, g^{\mathbb{D}})$ with the curvature parameter c (the actual curvature value is then $-c^2$) in this work. The model is defined by the manifold $\mathbb{D}^n = \{x \in \mathbb{R}^n : c\|x\|^2 < 1, c \geq 0\}$ endowed with the Riemannian metric $g^{\mathbb{D}} = \lambda_c^2 g^E$, where $\lambda_c = \frac{2}{1-c\|x\|^2}$ is the conformal factor and $g^E = \mathbf{I}_n$ is the Euclidean metric tensor [18], [19], [20].

Since the hyperbolic space does not behave like a traditional vector space, we must introduce the gyrovector formalism [50] to perform operations like addition [18]. As a result, we can define the following operations in Poincaré ball.

1) Möbius Addition: For a pair $\mathbf{x}, \mathbf{y} \in \mathbb{D}_c^n$, their addition is defined below.

$$\mathbf{x} \oplus_c \mathbf{y} = \frac{(1 + 2c\langle \mathbf{x}, \mathbf{y} \rangle + c\|\mathbf{y}\|^2)\mathbf{x} + (1 - c\|\mathbf{x}\|^2)\mathbf{y}}{1 + 2c\langle \mathbf{x}, \mathbf{y} \rangle + c^2\|\mathbf{x}\|^2\|\mathbf{y}\|^2}. \quad (1)$$

2) Hyperbolic Distance: The hyperbolic distance between $\mathbf{x}, \mathbf{y} \in \mathbb{D}_c^n$ is defined below.

$$D_{hyp}(\mathbf{x}, \mathbf{y}) = \frac{2}{\sqrt{c}} \text{arctanh}(\sqrt{c}\| - \mathbf{x} \oplus_c \mathbf{y}\|). \quad (2)$$

Note that with $c \to 0$, the distance function (2) reduces to the Euclidean distance: $\lim_{c \to 0} D_{hyp}(\mathbf{x}, \mathbf{y}) = 2\|\mathbf{x} - \mathbf{y}\|$.

3) Exponential Mapping Function: We also need to define a bijective map from Euclidean space to the Poincaré model of hyperbolic geometry. This mapping is termed *exponential*, while its inverse mapping from hyperbolic space to Euclidean space is called *logarithmic*. For some fixed base point $\mathbf{x} \in \mathbb{D}_c^n$,

Algorithm 2 Hierarchical Hyperbolic K-Means

Input: Image hyperbolic embedding Z; Number of hierarchies L, Number of clusters at the l-th hierarchy M_l.
Output: Hierarchical semantic structures $(\mathcal{P}, \mathcal{E})$.
$\{p_j^1\}_{j=1}^{M_1} \leftarrow$ Hyperbolic K-Means(Z)
$\mathcal{E} = \{(z_i, Parent(z_i))\}_{i=1}^{N}$
for l=2 *to* L **do**
  $\{p_j^l\}_{j=1}^{M_l} \leftarrow$ Hyperbolic K-Means($\{p_j^{l-1}\}_{j=1}^{M_{l-1}}$)
  /* Update connections. $Parent(\cdot)$ means the parent node at a higher level. */
  $\mathcal{E} \leftarrow \mathcal{E} \cup \{(p_j^{l-1}, Parent(p_i^{l-1}))\}_{j=1}^{M_{l-1}}$

/* More details about Hyperbolic K-Means */
Initial prototypes using K-Means++ in the hyperbolic space with the distance calculated by Equation (2).
while not converged **do**
  calculate cluster assignments according to Equation (2);
  /* The following is the calculation of new prototypes. */
  Map all points to the Klein model with Equation (6);
  Compute Einstein midpoints with Equation (4);
  Project all midpoints (prototypes) back to the Poincaré ball;

the exponential mapping is a function $\exp_{\mathbf{x}}^c: \mathbb{R}^n \to \mathbb{D}_c^n$ that is defined as follows:

$$\exp_{\mathbf{x}}^c(\mathbf{v}) = \mathbf{x} \oplus_c \left(\tanh\left(\sqrt{c} \frac{\lambda_{\mathbf{x}}^c \|\mathbf{v}\|}{2} \right) \frac{\mathbf{v}}{\sqrt{c}\|\mathbf{v}\|} \right). \tag{3}$$

Usually, the base point $\mathbf{x}$ is set to $\mathbf{0}$, making the above formulas simple but with little bias to the original results [19].

4) Advantages for Embedding Hierarchies: Recent studies [17], [19], [51] have demonstrated that, in the hyperbolic space, the local distances are scaled by the factor λ_c, approaching infinity near the boundary of the ball. As a result, the hyperbolic space has the "space expansion property". While in the Euclidean space, the volume of an object with a diameter of r scales polynomially with r, in the hyperbolic space, the counterpart scales *exponentially* with r. Intuitively, this is a continuous analog of trees: for a tree with a branching factor k, we obtain $O(k^d)$ nodes on level d, which in this case serves as a discrete analog of the radius. This property allows us to efficiently embed hierarchical data even in low dimensions, which is made precise by embedding theorems for trees and complex networks.

C. Hierarchical Hyperbolic K-Means

We aim to construct hierarchical structures by capturing hierarchical relationships among semantic clusters in the hyperbolic space. Different from recent research on hyperbolic clustering [40], we need a flexible as well as concise clustering algorithm for fast hierarchy construction in the Poincaré ball. To this end, we propose the *hierarchical hyperbolic K-Means* algorithm in the Poincaré ball, constructing the structures in a bottom-up manner.

The details of *hierarchical hyperbolic K-Means* are shown in Algorithm 2. We define the number of prototypes at the l-th layer as M_l and the total number of layers as L. First, we obtain the hyperbolic embeddings $Z = \{z_i\}_{i=1}^N$ of all images before each training epoch. Then, we perform *hyperbolic K-Means* with Z to obtain the prototypes of the first/bottom layer, *i.e.*, $\{p_j^1\}_{j=1}^{M_1}$. Similarly, the prototypes of each higher layer are derived by iteratively applying hyperbolic K-Means to the prototypes of the layer below [22]. We record the hierarchical information by maintaining the prototype set $\mathcal{P} = \{\{p_j^l\}_{j=1}^{M_l}\}_{l=1}^{L}$ and the connection set $\mathcal{E}$.

In the above process, the *hyperbolic K-Means* algorithm is the key to achieving the construction in hyperbolic space. Since existing variants of K-Means define the prototype by the Euclidean averaging operation over all embeddings within the cluster, they cannot run in the hyperbolic space. To address this issue, we preserve the basic idea of K-Means which optimizes the prototypes and cluster assignments alternatively, and define 1) the distance metric in the hyperbolic space and 2) the calculation of prototypes. The former has been solved in Equation (2). For the latter, referring to [52] and [53], we compute the **Einstein midpoint** as the prototype in the hyperbolic space. The prototype has the most concise form with the Klein coordinates [18], [20] and can be described as follows:

$$Klein\_Proto(x_1, \ldots, x_N) = \sum_{i=1}^{N} \gamma_i x_i / \sum_{i=1}^{N} \gamma_i, \tag{4}$$

where $\gamma_i = \frac{1}{\sqrt{1-c||x_i||^2}}$ is the Lorentz factor [20]. Since the Klein model and the Poincaré ball model are isomorphic [18], we can transition between $x_{\mathbb{D}}$ in the Poincaré ball and $x_{\mathbb{K}}$ in the Klein model as follows:

$$x_{\mathbb{K}} = \frac{2x_{\mathbb{D}}}{1 + c||x_{\mathbb{D}}||^2}, \tag{5}$$

$$x_{\mathbb{D}} = \frac{x_{\mathbb{K}}}{1 + \sqrt{1 - c||x_{\mathbb{K}}||^2}}. \tag{6}$$

Based on these formulas, we can map all the points in the Poincaré ball to the Klein model, computing the prototypes via Equation (4) first and then projecting them back to the Poincaré model. As a result, we can conduct hyperbolic K-Means clustering via alternative optimization with the distance metric and the prototype calculation like the existing K-Means algorithms.

D. Hierarchical Contrastive Learning

Since the contrastive hashing methods [13], [14], [15] have achieved satisfying retrieval performance without hand-crafted labels, we follow [13], [14], and [15] to introduce contrastive learning into unsupervised hashing. To incorporate hierarchical information into the contrastive hashing framework, we propose hierarchical instance-wise contrastive learning and hierarchical prototype-wise contrastive learning, as well as extending them to work in hyperbolic space. We will elaborate

on how contrastive hashing benefits from hierarchical semantic structures using both learning patterns.

1) Hierarchical Instance-Wise Contrastive Learning: Instance-wise contrastive learning pushes the embeddings of two transformed views of the same image (positives) close to each other and further apart from the embeddings of other images (negatives), where the instance-wise contrastive loss is defined as follows:

$$\tilde{\ell}_i^1 = -\log \frac{exp\left(-D\left(z_i^1, z_i^2\right)/\tau\right)}{exp\left(-D\left(z_i^1, z_i^2\right)/\tau\right) + \sum_{v, z_n \in \mathcal{N}(z_i)} exp\left(-D\left(z_i^1, z_n^v\right)/\tau\right)}, \tag{7}$$

where z_i^v is the representation of the v-th view of x_i, $\tilde{\ell}_i^v$ denotes the instance-wise contrastive loss of z_i^v, $\mathcal{N}(z_i)$ is the negative set of z_i, τ is the temperature parameter [36], [44], and D is the distance metric.

In practice, D can be defined as the hyperbolic distance in Equation (2) to achieve hyperbolic contrastive learning. Meanwhile, we can also define D by cosine distance in hypersphere space:

$$D_{cos}\left(z_i, z_j\right) = \frac{K}{2}\left(1 - cos\left(z_i, z_j\right)\right). \tag{8}$$

As a result, the instance-wise contrastive loss for all views of instance z_i is formulated as

$$L_{inst}\left(z_i, \mathcal{N}(z_i)\right) = \sum_{v=1}^{2} \tilde{\ell}_i^v. \tag{9}$$

Recent studies show that the selection of negative samples is critical for the quality of contrastive learning [23], [36], [54]. Existing contrastive hashing methods [13], [14], [15] suffer from the false-negative problem [24], [54] since they treat all the remaining images z_j^v within a mini-batch as negatives when given a random anchor image z_i^v, even if the negative images share the same semantic as the anchor image. We aim to sample distinctive negative samples according to the hierarchical semantic structure, achieving a solid discriminating ability for the contrastive hashing model.

Compared to flat structures, hierarchical structures can provide hierarchical similarity based on affiliation, resulting in accurate cross-sample similarity for negative sampling. Specifically, we sample negative samples according to the hierarchical structure constructed in § III-C. Given the anchor sample z_i, the negative sample set $\mathcal{N}^l(z_i)$ at the l-th layer can be defined by

$$\mathcal{N}^l\left(z_i\right) = \left\{z_j \mid j = 1, \ldots, B \text{ and } \mathcal{P}^l(z_j) \neq \mathcal{P}^l(z_i)\right\}, \tag{10}$$

where $\mathcal{P}^l(z_i)$ is the ancestor of z_i at the l-th layer. The above equation implies that the negative sample set of the anchor sample z_i only contains samples with a different ancestor than z_i at the l-th layer.

Note that the higher the level, the more inaccurate the relationship captured by hyperbolic K-Means is since the relationships at higher levels are determined by the clustering results at lower levels, which means the error will be amplified at higher levels. As a result, the overall hierarchical instance-wise contrastive learning objective $\mathcal{L}_{H-inst}$ can be formulated as a weighted summation of contrastive loss in Equation (9) at different layers, *i.e.*,

$$\mathcal{L}_{H-inst} = \frac{1}{B \cdot L} \sum_{i=1}^{B} \sum_{l=1}^{L} \frac{1}{l} L_{inst}\left(z_i, \mathcal{N}^l(z_i)\right). \tag{11}$$

2) Hierarchical Prototype-Wise Contrastive Learning: Compared to instance-wise contrastive learning methods [36], [44], [55] that learn data-data correlations, contrasting instance-prototype pairs can capture the global data distribution to acquire accurate cross-layer affiliation. To this end, we introduce prototype-wise contrastive learning [46], where we define the ancestor $\mathcal{P}^l(z_i)$ of z_i as the positive sample and all the remaining prototypes as negative samples. Analogous to Equation (7), the prototype-wise contrastive loss is defined below.

$$\hat{\ell}_i^v = -\log \frac{exp\left(-D\left(z_i^v, \mathcal{P}(z_i)\right)/\tau\right)}{exp\left(-D\left(z_i^v, \mathcal{P}(z_i)\right)/\tau\right) + \sum_{p_n \in \mathcal{N}_p(z_i)} exp\left(-D(z_i^v, p_n)/\tau\right)}, \tag{12}$$

where $N_p(z_i)$ is the negative prototype set of z_i and $p_n \in N_p(z_i)$. The prototype-wise contrastive loss for all views can be defined as follows:

$$L_{proto}(z_i, \mathcal{N}_p(z_i)) = \sum_{v=1}^{2} \hat{\ell}_i^v. \tag{13}$$

We employ a different negative sampling strategy for hierarchical prototype-wise contrastive learning than for hierarchical instance-wise learning for two reasons. On the one hand, since negative prototypes are usually distinct from the anchor image, prototype-wise contrastive learning suffers less from the false-negative problem than instance-wise contrastive learning. On the other hand, since the number of prototypes is much less than the number of instances, the negative sampling strategy for hierarchical instance-wise contrastive learning will cause the problem of under-sampling for negatives in hierarchical prototype-wise contrastive learning.

By contrasting prototypes at different layers, we define the hierarchical prototype-wise contrastive loss as follows:

$$\mathcal{L}_{H-proto} = \frac{1}{B \cdot L} \sum_{i=1}^{B} \sum_{l=1}^{L} \frac{1}{l} L_{proto}\left(z_i, \mathcal{N}_p^l(z_i)\right). \tag{14}$$

In addition, referring to [56], we define the quantization loss to reduce the accumulated quantization error caused by the continuous relaxation, where the loss is defined below.

$$\mathcal{L}_Q = \frac{1}{2} \sum_{i=1}^{B} \sum_{k=1}^{K} \sum_{v=1}^{2} \left(\log cosh(|h_{i,k}^v| - \mathbf{1})\right), \tag{15}$$

where $h_{i,k}$ is the k-th bit of h_i and $\mathbf{1} \in \mathbb{R}^K$ is the vector of ones. Note that h is derived from the output of the hash layer, rather than being quantized from the hyperbolic embedding z. Finally, the overall learning objective of hierarchical contrastive hashing can be formulated as follows:

$$\mathcal{L} = \mathcal{L}_{H-inst} + \mathcal{L}_{H-proto} + \lambda \mathcal{L}_Q, \tag{16}$$

where λ is the hyper-parameter to trade off different loss items.

TABLE I
COMPARISON IN MAP OF HAMMING RANKING FOR DIFFERENT BITS ON IMAGE RETRIEVAL

Method	Reference	ImageNet			CIFAR-10			FLICKR25K			NUS-WIDE		
		16-bit	32-bit	64-bit	16-bit	32-bit	64-bit	16-bit	32-bit	64-bit	16-bit	32-bit	64-bit
SGH [9]	ICML'17	0.557	0.572	0.583	0.286	0.320	0.347	0.608	0.657	0.693	0.463	0.588	0.638
SSDH [6]	IJCAI'18	0.604	0.619	0.631	0.241	0.239	0.256	0.710	0.696	0.737	0.542	0.629	0.635
BGAN [12]	AAAI'18	0.649	0.665	0.675	0.535	0.575	0.587	0.766	0.770	0.795	0.719	0.745	0.761
DistillHash [57]	CVPR'19	0.654	0.671	0.683	0.547	0.582	0.591	0.779	0.793	0.801	0.722	0.749	0.762
MLS^{3}RDUH [58]	IJCAI'20	0.662	0.680	0.691	0.562	0.588	0.595	0.797	0.809	0.809	0.730	0.754	0.764
TBH [10]	CVPR'20	0.636	0.653	0.667	0.432	0.459	0.455	0.779	0.794	0.797	0.678	0.717	0.729
UDH [59]	TIP'21	0.659	0.677	0.688	0.533	0.578	0.594	0.707	0.704	0.708	0.742	0.752	0.757
CIBHash [14]	IJCAI'21	0.719	0.733	0.747	0.547	0.583	0.602	0.773	0.781	0.798	0.756	0.777	0.781
PURPLE [60]	MM'22	0.737	0.749	0.762	0.607	0.633	0.656	0.807	0.819	0.824	0.768	0.785	0.798
CUDH [61]	TCSVT'23	0.708	0.716	0.729	0.575	0.597	0.605	0.762	0.777	0.785	0.723	0.757	0.770
DSCH [15]	AAAI'22	0.749	0.761	0.774	0.624	0.644	0.670	0.817	0.827	0.828	0.770	0.792	0.801
HHCH	**Ours**	**0.783**	**0.814**	**0.826**	**0.631**	**0.657**	**0.681**	**0.825**	**0.838**	**0.842**	**0.797**	**0.820**	**0.828**

IV. EXPERIMENTS

In this section, we conduct experiments on four public benchmark datasets to evaluate the superiority of our proposed HHCH. Note that baseline results are reported from DSCH [15].

A. Dataset and Evaluation Metrics

The public benchmark datasets include ImageNet [62], CIFAR-10 [63], FLICKR25K [64], and NUS-WIDE [65].

ImageNet is a commonly used single-label image dataset. Following [1], [66], and [67], we randomly select 100 categories for the experiments. Besides, we use 5,000 images as the query set and the remaining images as the retrieval set, where we randomly select 100 images per category as the training set.

CIFAR-10 consists of 60,000 images containing 10 classes. We follow the common setting [15] and select 1,000 images (100 per class) as the query set. The remaining images are used as the retrieval set, where we randomly selected 1,000 images per class to form the training set.

NUS-WIDE is a multi-label dataset that contains 269,648 images from 81 classes. Following the commonly used setting [10], [14], [15], we only use images selected from the 21 most frequent classes. Besides, we sample 100 images per class as the query set and use the remaining as the retrieval set, where we randomly select 10,500 images (5,00 images per class) to form the training set.

FLICKR25K is a multi-label dataset containing 25,000 images from 24 categories. Following [15], we randomly sample 1,000 images as the query set, and the remaining images are left for the retrieval set. In the retrieval set, we randomly choose 10,000 images as the training set.

The detailed summary of the dataset setting can be found in Table II. For ImageNet and NUS-WIDE, we follow [1] and [67] to use the commonly adopted index files and compressed datasets from the repository of HashNet [66] to form the splits. Besides, we follow [7] to implement CIFAR-10 and FLICKR25K.

1) Evaluation Protocol: To evaluate retrieval quality, we follow [1], [10], and [58] to employ the following metrics: 1) Mean Average Precision (mAP), 2) Precision-Recall (P-R) curves, 3) Precision curves w.r.t. different numbers of returned samples (P@N), 4) Precision curves within Hamming radius 2 (P@H$\leq$2), 5) Mean intra-class distance d_{intra}, and 6) Mean inter-class distance d_{inter}.

TABLE II
EXPERIMENTAL SETTINGS FOR ALL DATASETS

Dataset	#Train	#Query	#Retrieval	#Class
ImageNet	1,0000	5,000	128,503	100
CIFAR-10	10,000	1,000	59,000	10
FLICKR25K	10,000	1,000	24,000	24
NUS-WIDE	10,500	2,100	193,734	21

The mAp is one popular criterion to measure retrieval performance. For query data, a list of R-ranked points is first returned. Then the average precision (AP) is given as:

$$AP = \frac{1}{N_g} \sum_{r=1}^{R} P_i(r)\delta_i(r), \tag{17}$$

where N_g is the number of ground-truth neighbors of the query, and $P(r)$ is the precision value of the first r returned data points. Note that $\delta_i(r) = 1$ if the r-th retrieved entity is a ground-truth neighbor and $\delta_i(r) = 0$ otherwise. mAP denotes the average APs for queries. R represents the number of retrieval points. Precision-Recall shows recall and the corresponding precision values, and is usually obtained by varying the Hamming radius and evaluating the precision and recall values. P@N gives the precision for different returned data point numbers, while P@H$\leq$2 indicates the precision for returned points that are within the Hamming radius compared to the query. d_{inter} and d_{intra} directly evaluate the compactness and dispersion of the learned hash codes.

According to [10] and [14], we adopt mAP@1000 for ImageNet and CIFAR-10, as well as mAP@5000 for FLICKR25K and NUS-WIDE. In addition, for multi-label datasets, the retrieved images are considered similar to the query if at least one of the labels is the same.

B. Implementation Details

1) Model Details: For fair comparisons, we follow DSCH [15] to adopt VGG19 [49] pre-trained on ImageNet [62] as the backbone, and use the hash layer consisting of two fully-connected layers with $ReLU$ as the activation function for hash code projection [15]. The $3\times224\times224$ image

TABLE III
THE d_{intra} AND d_{inter} ON IMAGENET

Datasets	Method	$d_{intra}\downarrow$			$d_{inter}\uparrow$		
		16-bit	32-bit	64-bit	16-bit	32-bit	64-bit
ImageNet	CIBHash	1.23	2.54	5.12	8.04	16.08	32.16
	DSCH	1.03	2.24	4.59	8.06	16.12	32.24
	HHCH	**0.81**	**1.88**	**3.88**	**8.08**	**16.15**	**32.31**
NUS-WIDE	CIBHash	4.43	**8.59**	**17.06**	6.29	12.45	26.94
	DSCH	**4.31**	9.07	18.74	6.74	13.17	27.78
	HHCH	4.40	9.23	19.11	**7.11**	**14.05**	**28.81**

will be transformed to a $4096-d$ feature vector and then to the K-bit continuous hash code. In addition, we have an auxiliary projection head Exp_map parameterized by θ_e after the hash layer. The K-bit hash code will finally be projected to a d-dimensional hyperbolic embedding in the Poincaré ball.

2) Training Details: We implement HHCH in PyTorch [68] and train the model with an NVIDIA RTX 3090 GPU. Following [10], [14], we freeze the backbone and only train the hash layer and the projection head. For data augmentation, we use the same strategy as CIBHash [14] and DSCH [15]. We set the curvature parameter $c = 0.1$ for ImageNet, $c = 0.01$ for other datasets (See § IV-F), and set the dimensionality of hyperbolic embeddings to $d = 128$. The temperature parameter $\tau = 0.2$ (Equations (7) and (12)). The default M_l and L are set to $[1500 \rightarrow 1000 \rightarrow 800]$ for ImageNet, $[100 \rightarrow 80 \rightarrow 50]$ for CIFAR-10, $[200 \rightarrow 150 \rightarrow 80]$ for FLICKR25K, and $[200 \rightarrow 150 \rightarrow 80]$ for NUS-WIDE (See § IV-F for detailed investigation with different clustering settings). We set the batch size $B = 64$ and adopt the Adam optimizer [69] with a learning rate $lr = 0.001$.

C. Comparison With State-of-the Art

The mAP results on four benchmark datasets are shown in Table I. It is clear that our proposed HHCH consistently achieves the best retrieval performance among the four image datasets, with an average increase of 4.5%, 1.6%, 1.3%, and 3.5% on ImageNet, CIFAR-10, FLICKR25K, and NUS-WIDE compared with DSCH, respectively. Besides, the advantageous results are significant with probability value $p \leq 0.01$ on all datasets compared with the most competitive method DSCH. We also report the mean intra-class distance d_{intra} and mean inter-class distance d_{inter} on ImageNet and NUS-WIDE in Table III. We can observe that on a single label dataset, HHCH can achieve superior d_{intra} and d_{inter}, indicating that samples are well separated. However, on the NUS-WIDE dataset, we observe comparatively lower d_{inter} scores. This discrepancy arises because d_{inter} fails to accurately reflect the true compactness of clusters in multi-label datasets. Nevertheless, our method consistently outperforms the baselines and attains higher d_{inter} values. These findings demonstrate that HHCH effectively learns more compact hash codes while exhibiting superior disentangling capabilities.

In addition, we report the P-R curves, P@N curves, and P@H$\leq$ 2 curves at 64 bits in Fig. 3. Obviously, HHCH outperforms all compared methods by large margins on both ImageNet and FLICKR25K w.r.t. the three metrics. As a conclusion, these comparisons of various dimensions demonstrate that the proposed HHCH, which explores hierarchical information in a new way, can produce high-quality hash codes, *i.e.*, hash codes of similar data are more aggregated while hash codes of dissimilar data are more distant.

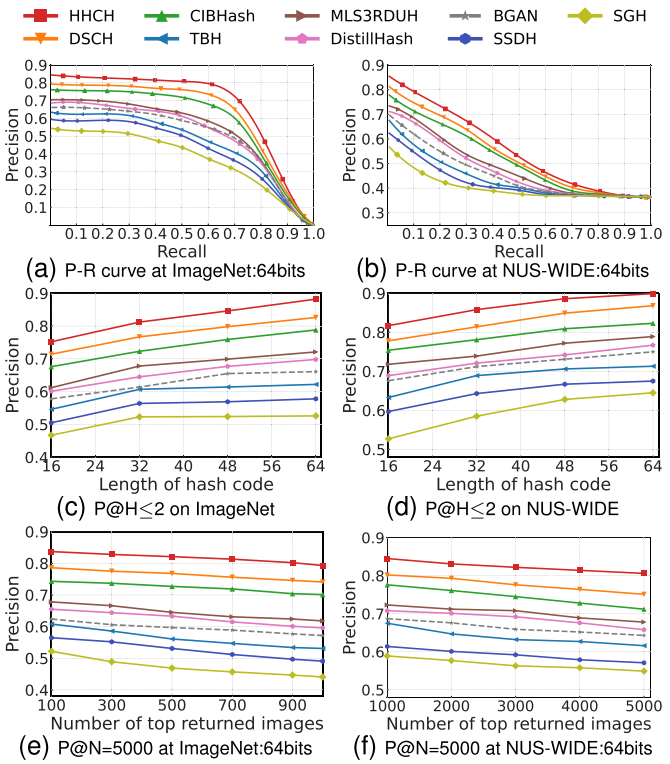

Fig. 3. P-R curves, P@N, and P@H≤2 of HHCH and comparison methods on ImageNet and NUS-WIDE.

In addition, our proposed HHCH achieves a larger performance improvement on the ImageNet dataset compared to CIFAR-10. There are two reasons for this: 1) Compared to CIFAR-10, the ImageNet dataset is more diverse, with more classes and samples derived from the real world, which suggests that latent hierarchical semantic structures are more prominent on ImageNet. 2) The CIFAR-10 dataset contains low-resolution images, which leads to inaccurate visual hierarchies.

D. Performance With Different Backbones

Feature extraction plays a critical role in determining the retrieval performance of the hashing model. In recent times, transformer-based feature backbones have demonstrated their superiority in extracting image features. To explore the advantages of a more powerful backbone in our approach, we replace the VGG network with pre-trained models such as ViT [70], Swin Transformer [71], and the visual branch of the CLIP model [72]. The corresponding mean average precision (mAP) performance is reported in Table IV.

We can observe that transformer-based backbones result in significant performance improvements compared to the VGG network baseline model. Particularly noteworthy is the Swin Transformer, with its hierarchical structure enabling the capture of both local and global image information, surpassing the standard ViT model. Ultimately, the CLIP model, which is pretrained on an extensive image-text dataset, attains the highest performance, showcasing remarkable average gains

TABLE IV
THE MAP PERFORMANCE CONCERNING DIFFERENT FEATURE BACKBONES

Backbone	ImageNet			NUS-WIDE		
	16-bit	32-bit	64-bit	16-bit	32-bit	64-bit
VGG	0.783	0.814	0.826	0.797	0.820	0.828
ViT	0.803	0.837	0.845	0.804	0.825	0.834
Swin	0.811	0.848	0.854	0.809	0.828	**0.839**
CLIP	**0.828**	**0.857**	**0.872**	**0.812**	**0.829**	0.838

TABLE V
THE MAP PERFORMANCE OF INTRODUCING HYPERBOLIC SPACE FOR DIFFERENT COMPONENTS. ✓ DENOTES THAT WE INTRODUCE HYPERBOLIC SPACE FOR THIS MODULE, WHILE ✗ MEANS THAT WE USE HYPER-SPHERE SPACE. **CL** AND **HC** REPRESENT THE HIERARCHICAL CONTRASTIVE LEARNING MODULE AND HIERARCHICAL CLUSTERING MODULE, RESPECTIVELY

Module		ImageNet			NUS-WIDE		
CL	HC	16-bit	32-bit	64-bit	16-bit	32-bit	64-bit
✗	✗	0.750	0.774	0.783	0.779	0.797	0.801
✓	✗	0.769	0.791	0.798	0.790	0.806	0.811
✗	✓	0.774	0.799	0.806	0.792	0.813	0.819
✓	✓	**0.783**	**0.814**	**0.826**	**0.797**	**0.820**	**0.828**

TABLE VI
THE MAP PERFORMANCE OF DIFFERENT EMBEDDING SPACE

Space	ImageNet			NUS-WIDE		
	16-bit	32-bit	64-bit	16-bit	32-bit	64-bit
Euclidean	0.597	0.637	0.697	0.741	0.766	0.783
Hyper-sphere	0.750	0.774	0.783	0.779	0.797	0.801
Hyperbolic	**0.783**	**0.814**	**0.826**	**0.797**	**0.820**	**0.828**

of 5.53% and 1.40% compared to VGG on ImageNet and NUS-WIDE, respectively.

E. Ablation Study

To justify how each component of HHCH contributes to final retrieval performance, we conduct studies on the effectiveness of 1) the introduction of hyperbolic space and 2) the utilization of hierarchical semantic structures.

1) Effect of Hyperbolic Embedding: We report the mAP performance of embedding in Euclidean space, hyper-sphere space, and hyperbolic space, as presented in Table V and VI. Note that the contrastive learning (CL) module and the hierarchical clustering (HC) module can be performed in hyperbolic space independently. We first test HHCH completely without hyperbolic space (*i.e.*, both CL and HC are conducted in hyper-sphere space, where we use cosine similarity as the distance metric), and the results are reported in the first row. We can observe a significant drop in the performance of HHCH when hyperbolic embedding is not enabled. However, incorporating hyperbolic embedding into the CL and HC modules of HHCH leads to notable enhancements, with average gains of 2.2% and 3.1% respectively, as demonstrated in the second and third rows of the table (in comparison to the first row). In addition, HHCH achieves the best performance when introducing hyperbolic embedding technology into both CL and HC modules.

Furthermore, we conducted experiments to examine the disparities among three embedding spaces by utilizing their respective distance metrics to compute similarity in Table VI. It is evident that, compared to Euclidean space and Hyper-sphere space, the average performance of the model on ImageNet exhibits improvements of 25.81% and 5.02%, respectively. Additionally, the enhancements in performance on NUS-WIDE are 6.78% and 2.85%, respectively.

As a conclusion, these results demonstrate that incorporating hyperbolic space into hierarchical hash learning leads to a substantial improvement in retrieval performance. This enhancement can be attributed to the excellent ability of hyperbolic space to embed hierarchical data with low dimensions.

2) Effect of Hierarchical Semantic Structures: In Sec. III-C, we use the hyperbolic hierarchical K-Means algorithm to mine the latent visual hierarchies in real-world datasets. We conduct experiments to evaluate the performance of hierarchical semantic structures under different settings, and the results are presented in Table VII. IC and PC denote the baseline models using instance-wise contrastive learning and prototype-wise contrastive learning, respectively. They have no perception of latent hierarchical semantic structures, resulting in sub-optimal retrieval performance. Comparing the third and first rows of Table VII (HIC, *i.e.*, hierarchical instance-wise contrastive learning vs. IC, *i.e.*, instance-wise contrastive learning without hierarchies), we can observe respective 3.4% and 1.5% performance gains on ImageNet and NUS-WIDE after adding the hierarchical information to the instance-wise contrastive learning. In addition, we count the number of false-negative samples for each anchor within a batch, where batch sizes range from 64 to 256. Note that our definition of false-negatives specifically refers to the interconnections among training samples within a mini-batch. It is a notion for training sample management rather than the definition used for evaluation in the testing phase, where the definition of false negatives in the testing phase involves the samples that are not retrieved but should be retrieved. The results presented in Table VIII highlight that the integration of HIC leads to a substantial reduction in the average number of false-negative samples. These results verify that hierarchies can effectively help instance-wise contrastive learning to sample more accurate negatives and mine more accurate cross-sample similarity. In addition, referring to the results in the fourth and second rows of Table VII (HPC vs. PC), we can conclude that HHCH achieves 4.2% and 2.0% performance improvements on ImageNet and NUS-WIDE, respectively, when employing prototype-wise contrastive learning with hierarchies. It demonstrates that accurate cross-layer affiliation, provided by hierarchical semantic structures, is beneficial for contrastive learning for hashing. Note that HIC+HPC achieves the best performance. We attribute it to the fact that HHCH consists of both hierarchical instance-wise contrastive learning and hierarchical prototype-wise contrastive learning, which promises to fully utilize hierarchical information from both local and global perspectives.

F. Sensitivity Analysis

In this section, we give a detailed analysis of the hyper-parameters in the model training phase, including the

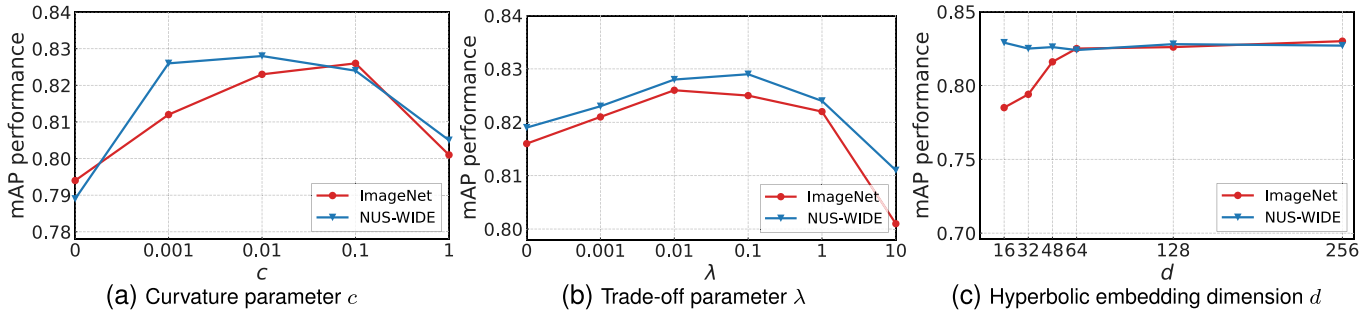

Fig. 4. The mAP performance w.r.t. different c and λ at 64 bits on ImageNet and NUS-WIDE.

TABLE VII

ABLATION STUDIES ON HIERARCHICAL SEMANTIC STRUCTURES (*i.e.*, HIERARCHICAL CLUSTERING). **IC**: INSTANCE-WISE CONTRASTIVE LEARNING WITHOUT HIERARCHIES; **HIC**: HIERARCHICAL INSTANCE-WISE CONTRASTIVE LEARNING; **PC**: PROTOTYPE-WISE CONTRASTIVE LEARNING WITHOUT HIERARCHIES; **HPC**: HIERARCHICAL PROTOTYPE-WISE CONTRASTIVE LEARNING

Setting	ImageNet			NUS-WIDE		
	16-bit	32-bit	64-bit	16-bit	32-bit	64-bit
IC	0.735	0.758	0.763	0.755	0.781	0.789
PC	0.729	0.747	0.758	0.744	0.777	0.784
HIC	0.755	0.784	0.795	0.768	0.787	0.805
HPC	0.750	0.782	0.798	0.761	0.788	0.802
HIC+HPC	**0.783**	**0.814**	**0.826**	**0.797**	**0.820**	**0.828**

TABLE VIII

STATISTICS REGARDING THE AVERAGE NUMBER OF FALSE-NEGATIVE SAMPLES PER ANCHOR VARIES WITH THE BATCH SIZE (BS)

Setting	ImageNet				NUS-WIDE			
	64-bs	128-bs	256-bs	512-bs	64-bs	128-bs	256-bs	512-bs
IC	0.621	1.25	2.52	4.95	19.72	39.56	78.32	157.01
HIC	**0.579**	**1.08**	**2.25**	**4.05**	**15.31**	**33.16**	**64.31**	**115.89**

TABLE IX

SENSITIVITY ANALYSIS ON THE NUMBER OF LAYERS L AND THE NUMBER OF PROTOTYPES AT DIFFERENT LAYERS M_l. WE PRESENT THE CONFIGURATION OF M_l AND L AS $M_1 \rightarrow M_2 \rightarrow \ldots, \rightarrow M_L$

Dataset	Configuration of M_l and L	mAP@64-bit
ImageNet	500	0.789
	1500	0.806
	$1500 \rightarrow 1000$	0.817
	$2000 \rightarrow 1500 \rightarrow 800$	0.825
	$1500 \rightarrow 1000 \rightarrow 800$	**0.826**
	$1000 \rightarrow 800 \rightarrow 500$	0.822
	$1500 \rightarrow 1000 \rightarrow 800 \rightarrow 500$	0.824
NUS-WIDE	100	0.798
	200	0.794
	$200 \rightarrow 150$	0.817
	$300 \rightarrow 120 \rightarrow 100$	0.826
	$200 \rightarrow 150 \rightarrow 80$	**0.828**
	$100 \rightarrow 80 \rightarrow 50$	0.822
	$200 \rightarrow 150 \rightarrow 80 \rightarrow 40$	0.827

M_l and L of the hierarchical hyperbolic K-Means, the curvature parameter c of the hyperbolic space, and the trade-off parameter λ. Since parameters like the batch size B and the temperature parameter τ, *etc.*, have been analyzed in the related works [14], [36], we do not experiment on these parameters.

1) Sensitivity to M_l and L: We test the model's performance with a variation of the number of layers and the number of prototypes at each layer. As shown in Table IX, we can draw the following conclusions: 1) Deeper hierarchies can improve retrieval performance. Compared with learning with only one layer, HHCH achieves 2.5% and 4.3% improvement on ImageNet and NUS-WIDE under the best settings (*i.e.*, $1500 \rightarrow 1000 \rightarrow 800$ and $200 \rightarrow 150 \rightarrow 80$), respectively. However, it is worth noting that even when constructing four-layer hierarchies for ImageNet and NUS-WIDE, we only achieve mAP values of 0.824 and 0.827, respectively. This observation suggests that the performance does not exhibit a linear relationship with the depth of the hierarchy. There exists a trade-off between performance and the computational overhead involved. 2) HHCH relies on sufficient prototypes to fully capture the latent distribution. As depicted in Table IX, the configurations of $2000 \rightarrow 1500 \rightarrow 800$ and $1000 \rightarrow 800 \rightarrow 500$ exhibit inferior performance compared to the optimal setting on ImageNet. In the case of a three-level hierarchy, increasing the number of prototypes leads to performance enhancements. However, excessive prototypes do not yield further performance improvements. This observation holds for NUS-WIDE as well.

2) Sensitivity to c: We investigate the effect of the curvature parameter c. Intuitively, the smaller c is, the flatter the Poincaré ball is. As shown in Fig. 4 (a), mAP increases as c at the beginning. It gets a peak value at $c = 0.01$ or $c = 0.1$ but drops off sharply after $c = 0.1$. In addition, we find that the optimal mAP for ImageNet is higher than that for NUS-WDIE. We attribute it to the clear hierarchical semantic structures of the ImageNet dataset that are organized according to the WordNet [73] hierarchy.

3) Sensitivity to λ: We test the model's performance depending on the trade-off parameter λ at 64 bits on ImageNet and NUS-WIDE. As shown in Fig. 4 (b), we can observe that both small and large λ will decrease the mAP performance. A small λ (*e.g.*, $\lambda < 0.001$) cannot reduce the accumulated quantization error caused by the continuous relaxation, resulting in considerable information loss. In contrast, a large lambda (*e.g.*, $\lambda > 1$) will force the quantization loss item to dominate the overall learning objective, resulting in the difficulty of optimization. As a result, we opt for $\lambda = 0.01$.

4) Sensitivity to d: The dimension of the hyperbolic embedding also plays a crucial role in determining the ultimate retrieval performance. Our experiments on ImageNet and NUS-WIDE, using hash code lengths of 64 bits, are presented

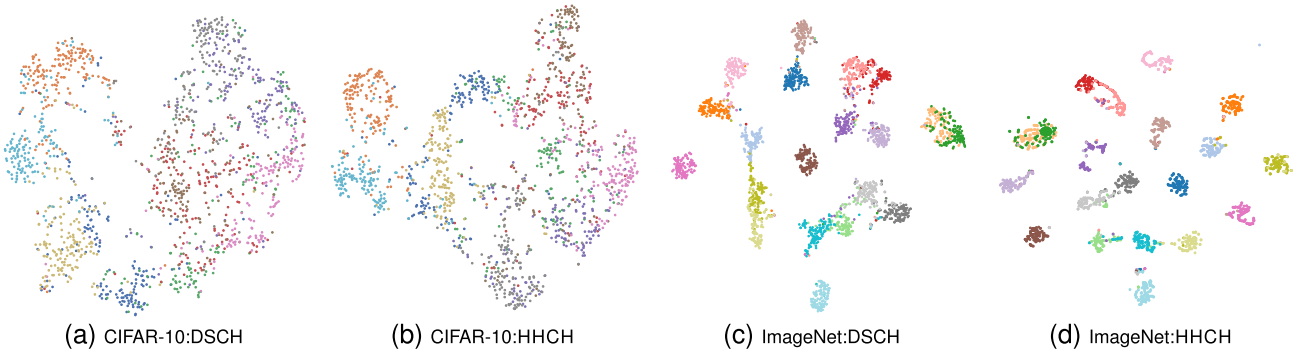

Fig. 5. The t-SNE visualization of the learned 64-bit hash codes from the training sets. The scattered elements of the same color indicate the same category. Note that we only visualize the first 20 classes for ImageNet.

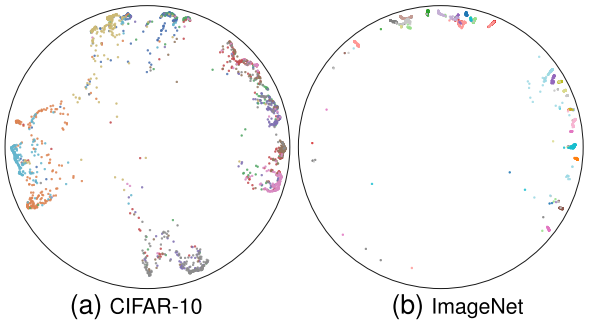

Fig. 6. The UMAP visualization of 128-d hyperbolic embeddings from ImageNet (The first 20 classes) and CIFAR-10 in the Poincaré ball.

in Fig. 4 (c). Consequently, it can be concluded that a larger dimension d (ranging from 64 to 256) ensures stable and satisfactory performance, whereas a smaller dimension, such as $d = 16$, hampers performance due to the limited expression of features in low-dimensional space.

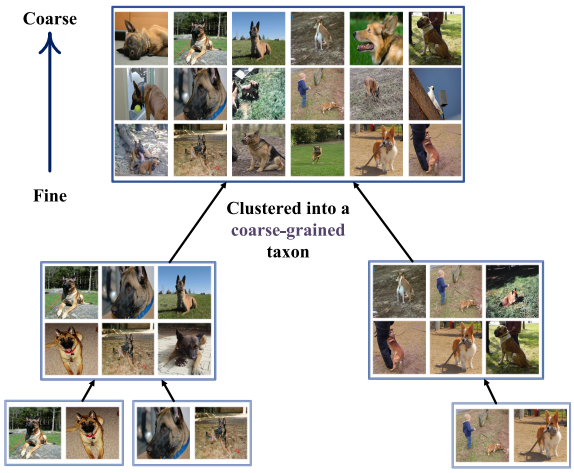

Fig. 7. Visualization of a captured visual hierarchical semantic structure. The arrow denotes that samples from the low layer are clustered into a coarse-grained taxon. Note that we only use a small part of the images for the visualization.

G. Visualization

1) Visualization of Hash Codes: Fig. 5 shows the t-SNE visualization [74] of the hash codes at 64 bits on CIFAR-10 and ImageNet. The hash codes generated by HHCH show favorable intra-class compactness and inter-class separability compared with the state-of-the-art hashing method DSCH. It demonstrates that HHCH can generate high-quality hash codes.

2) Visualization of Hyperbolic Embeddings: We illustrate the hyperbolic embeddings of CIFAR-10 and ImageNet in the Poincaré ball using UMAP [75] with the "hyperboloid" distance metric [19]. As shown in Fig. 6, we can see that the samples are clustered according to the labels, and each cluster is pushed to the border of the ball, indicating that the learned embeddings are distinguishable enough.

3) Visualization of Top-10 Retrieved Results: Fig. 8 illustrates the top-10 retrieved images and reports P@10 comparisons between HHCH and DSCH [15]. Our proposed HHCH achieves 90% and 100% in terms of P@10 when given the query images labeled as "Norfolk terrier" and "Sky&Clouds" on ImageNet and NUS-WIDE, respectively. It can be seen that our proposed HHCH yields more relevant and accurate retrieval results than DSCH.

4) Visualization of Hierarchical Semantics: In Fig. 7, we visualize the partial results of the hierarchical hyperbolic K-Means. We can observe that at the bottom layer, only dogs with high visual similarity are clustered together, whereas, at the top layer, they are grouped into a class with significant diversity. It is clear that images at low layers express finer-grained semantics and the high layers contain coarser-grained semantics, *e.g.*, images at the bottom of the hierarchy are naturally more visually similar, while the images at the top of the hierarchy are more diverse. These results indicate that HHCH is capable of capturing the hierarchical semantics of the data very well.

H. Time Complexity Analysis

Since HHCH involves an extra hierarchical hyperbolic K-Means algorithm before each epoch for the training phase, we discuss the possible extra time overhead according to time complexity. Note that N, M_l, L, and B denote the dataset size, number of prototypes at the l-th layer, number of layers, and the mini-batch size, respectively.

On the one hand, the time complexity of vanilla K-Means is $\mathcal{O}(NMt)$, where t is the number of iterations in K-Means and we set $t = 30$. For hierarchical hyperbolic K-Means, the extra time complexity of each training step is $\mathcal{O}(NM_1t + M_1M_2t + \ldots + M_{L-1}M_Lt)/(N/B)$. Since $M_l \ll N$, we can simplify it to $\mathcal{O}(BM_1t)$.

On the other hand, the time complexity of hierarchical instance-wise contrastive learning is $\mathcal{O}(B^2)$, and the counterpart of hierarchical prototype-wise contrastive learning is

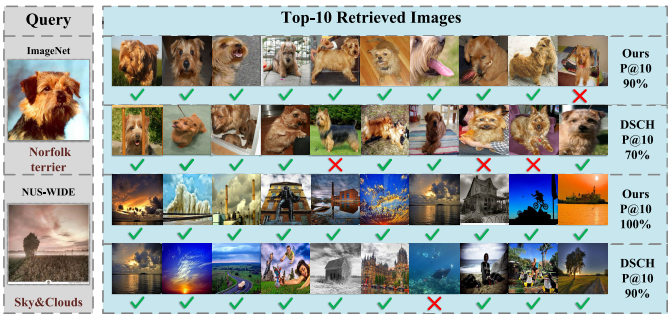

Fig. 8. Retrieval comparisons to DSCH [15] at 64 bits on three datasets. The Top-10 retrieved images are returned according to the Hamming distance between the query image and the database images. We report precision within the top-10 retrieved images (P@10) and the results demonstrate that our proposed HHCH outperforms DSCH with more relevant and accurate returned images as well as fewer contradictions.

TABLE X
ENCODING TIME COMPARISON OF VARIOUS DEEP HASH METHODS

Methods	TBH	CIBHash	DSCH	HHCH
Encoding time	2.5ms	2.6ms	2.9ms	2.7ms
Parameters	65.1M	136.4M	141.7M	141.7M

$\mathcal{O}(BM_1 + \ldots + BM_L) = \mathcal{O}(BM_1)$. As a result, the time complexity of hierarchical contrastive loss computation is $\mathcal{O}(B^2) + \mathcal{O}(BM_1)$.

In conclusion, the complete time complexity of HHCH is $\mathcal{O}(B^2) + \mathcal{O}(BM_1) + \mathcal{O}(BM_1t) = \mathcal{O}(BM_1t)$, which is consistent with state-of-the-art contrastive hashing methods DSCH [15] but a little higher than CIBHash [14] when $M_1t > B$.

Moreover, considering that the time it takes to generate binary codes is a crucial factor in practical retrieval systems, we also evaluate the encoding time of our method in comparison to several baseline approaches. We count the encode time from the image feature to the binary hash code, and record the average time for 100 images in Table X. Note that we run TBH, CIBHash, DSCH, and HHCH on the same platform to ensure fairness. It is clear that HHCH achieves significantly better results compared to baseline methods while requiring comparable time investment. In addition, we compare the number of parameters among different methods, as shown in Table X. It is evident that HHCH has a similar number of parameters as the highly competitive method DSCH, as they both employ the same backbone. However, HHCH has slightly more parameters than TBH, primarily due to the disparity between VGG19 and AlexNet architectures.

V. LIMITATIONS AND FUTURE WORK

Although our proposed HHCH achieves satisfying retrieval performance, there are still some limitations to be addressed in our future work.

- Expand to large-scale datasets. In our experiments, we evaluate our method on four datasets following baselines, the largest of which comprises hundreds of thousands of images. However, given the potential of our model, it is possible that these datasets do not reveal their full capacity. To further assess the scalability of our proposed method, we plan to continue enlarging the test datasets and deploy HHCH to real-world search engines.
- Adaptive selection of near-optimal L and M_l. Currently, we set the values for the number of layers L and the number of prototypes M_l at different layers based on prior experience and statistical analysis. While this approach provides greater flexibility, it can be challenging to identify near-optimal values for real-world datasets. Therefore, we aim to explore more practical solutions that enable adaptive search for M_l and L, non-parametric hierarchical clustering, *etc*.
- Discrete optimization for lower information loss. we learn continuous hash codes in our training phase, which will cause considerable information loss. We plan to explore more effective binary optimization strategies to achieve learning discrete binary code with minimal information loss.
- *Potential future applications.* Despite being trained on image datasets, the baseline HHCH can be seamlessly applied to video and multi-modal datasets with minimal modifications, as these datasets inherently possess hierarchical structures. Our future research endeavors will focus on exploring the extraction of hierarchical information from multi-modal datasets and conducting extensive experiments to empirically validate the effectiveness of HHCH in such diverse scenarios. Furthermore, the incorporation of hyperbolic embedding holds the promise of stimulating the research community's exploration into novel learning metrics, thereby inspiring more solid works.

VI. CONCLUSION

In this paper, we propose to learn hash codes by exploring the visual hierarchical semantic structures in hyperbolic space. As a result, we proposed a novel unsupervised hashing method named HHCH. In HHCH, we embed the continuous hash codes into hyperbolic space (*i.e*., the Poincaré ball) to achieve less information distortion. Furthermore, we extend the K-Means algorithm to hyperbolic space and perform hierarchical hyperbolic K-Means to capture the latent hierarchical semantic structures adaptively. In addition, we designed hierarchical contrastive learning, including hierarchical instance-wise contrastive learning and hierarchical prototype-wise contrastive learning, to take full advantage of the hierarchies. Extensive experiments on four benchmarks demonstrate that HHCH can benefit from hierarchies and outperforms the state-of-the-art unsupervised methods. Finally, HHCH can be easily integrated into most similarity search engines, providing efficient computation and storage for image retrieval tasks.

REFERENCES

[1] L. Yuan et al., "Central similarity quantization for efficient image and video retrieval," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 3080–3089.

[2] Y. Liang, Y. Pan, H. Lai, W. Liu, and J. Yin, "Deep listwise triplet hashing for fine-grained image retrieval," *IEEE Trans. Image Process.*, vol. 31, pp. 949–961, 2022.

[3] J.-N. Guo, X.-L. Mao, T. Lan, R.-X. Tu, W. Wei, and H. Huang, "LASH: Large-scale academic deep semantic hashing," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 2, pp. 1734–1746, Feb. 2023.

[4] Y. Hu, M. Liu, X. Su, Z. Gao, and L. Nie, "Video moment localization via deep cross-modal hashing," *IEEE Trans. Image Process.*, vol. 30, pp. 4667–4677, 2021.

[5] Y. Cao et al., "Scalable distributed hashing for approximate nearest neighbor search," *IEEE Trans. Image Process.*, vol. 31, pp. 472–484, 2022.

[6] E. Yang, C. Deng, T. Liu, W. Liu, and D. Tao, "Semantic structure-based unsupervised deep hashing," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, Jul. 2018, pp. 1064–1070.

[7] Y. Li and J. van Gemert, "Deep unsupervised image hashing by maximizing bit entropy," in *Proc. AAAI*. Washington, DC, USA: AAAI Press, 2021, pp. 2002–2010.

[8] R. Tu et al., "Unsupervised hashing with semantic concept mining," 2022, *arXiv:2209.11475*.

[9] B. Dai, R. Guo, S. Kumar, N. He, and L. Song, "Stochastic generative hashing," in *Proc. ICML*, in Proceedings of Machine Learning Research, vol. 70, 2017, pp. 913–922.

[10] Y. Shen et al., "Auto-encoding twin-bottleneck hashing," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 2815–2824.

[11] Y. Shen, L. Liu, and L. Shao, "Unsupervised binary representation learning with deep variational networks," *Int. J. Comput. Vis.*, vol. 127, nos. 11–12, pp. 1614–1628, Dec. 2019.

[12] J. Song, T. He, L. Gao, X. Xu, A. Hanjalic, and H. T. Shen, "Binary generative adversarial networks for image retrieval," in *Proc. AAAI*. Washington, DC, USA: AAAI Press, 2018, pp. 394–401.

[13] X. Luo et al., "A statistical approach to mining semantic similarity for deep unsupervised hashing," in *Proc. ACM Int. Conf. Multimedia*. New York, NY, USA: ACM, 2021, pp. 4306–4314.

[14] Z. Qiu, Q. Su, Z. Ou, J. Yu, and C. Chen, "Unsupervised hashing with contrastive information bottleneck," in *Proc. 13th Int. Joint Conf. Artif. Intell.*, Aug. 2021, pp. 959–965.

[15] Q. Lin, X. Chen, Q. Zhang, S. Cai, W. Zhao, and H. Wang, "Deep unsupervised hashing with latent semantic components," in *Proc. AAAI*. Washington, DC, USA: AAAI Press, 2022, pp. 7488–7496.

[16] R. Wei, Y. Liu, J. Song, Y. Xie, and K. Zhou, "Deep debiased contrastive hashing," *Pattern Recognit.*, vol. 139, Jul. 2023, Art. no. 109483.

[17] M. Nickel and D. Kiela, "Poincaré embeddings for learning hierarchical representations," in *Proc. NeurIPS*, 2017, pp. 6338–6347.

[18] V. Khrulkov, L. Mirvakhabova, E. Ustinova, I. Oseledets, and V. Lempitsky, "Hyperbolic image embeddings," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 6417–6427.

[19] A. Ermolov, L. Mirvakhabova, V. Khrulkov, N. Sebe, and I. V. Oseledets, "Hyperbolic vision transformers: Combining improvements in metric learning," 2022, *arXiv:2203.10833*.

[20] W. Peng, T. Varanka, A. Mostafa, H. Shi, and G. Zhao, "Hyperbolic deep neural networks: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 12, pp. 10023–10044, Dec. 2022.

[21] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proc. Berkeley Symp. Math. Statist. Prob.*, 1967, pp. 281–297.

[22] Y. Guo et al., "HCSC: Hierarchical contrastive selective coding," 2022, *arXiv:2202.00455*.

[23] Y. Zhang, R. Zhang, S. Mensah, X. Liu, and Y. Mao, "Unsupervised sentence representation via contrastive learning with mixing negatives," in *Proc. AAAI*. Washington, DC, USA: AAAI Press, 2022, pp. 11730–11738.

[24] R. Cao et al., "Exploring the impact of negative samples of contrastive learning: A case study of sentence embedding," in *Proc. Findings Assoc. Comput. Linguistics (ACL)*, 2022, pp. 3138–3152.

[25] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[26] Z. Chen, Q. Cui, B. Zhao, R. Song, X. Zhang, and O. Yoshie, "SST: Spatial and semantic transformers for multi-label image recognition," *IEEE Trans. Image Process.*, vol. 31, pp. 2570–2583, 2022.

[27] J. Yue, L. Fang, and M. He, "Spectral–spatial latent reconstruction for open-set hyperspectral image classification," *IEEE Trans. Image Process.*, vol. 31, pp. 5227–5241, 2022.

[28] C. Liu, H. Ding, Y. Zhang, and X. Jiang, "Multi-modal mutual attention and iterative interaction for referring image segmentation," *IEEE Trans. Image Process.*, vol. 32, pp. 3054–3065, 2023.

[29] X. He, Z. Zhong, L. Fang, M. He, and N. Sebe, "Structure-guided cross-attention network for cross-domain OCT fluid segmentation," *IEEE Trans. Image Process.*, vol. 32, pp. 309–320, 2023.

[30] Y. Wang, Z. Xuan, C. Ho, and G.-J. Qi, "Adversarial dense contrastive learning for semi-supervised semantic segmentation," *IEEE Trans. Image Process.*, vol. 32, pp. 4459–4471, 2023.

[31] L. Wu, L. Fang, X. He, M. He, J. Ma, and Z. Zhong, "Querying labeled for unlabeled: Cross-image semantic consistency guided semi-supervised semantic segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 7, pp. 8827–8844, Jul. 2023.

[32] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, vol. 1. Minneapolis, MI, USA: Association for Computational Linguistics, Jun. 2019, pp. 4171–4186.

[33] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," 2013, *arXiv:1312.6114*.

[34] M. Zieba, P. Semberecki, T. El-Gaaly, and T. Trzcinski, "BinGAN: Learning compact binary descriptors with a regularized GAN," in *Proc. NeurIPS*, 2018, pp. 3612–3622.

[35] K. G. Dizaji, F. Zheng, N. S. Nourabadi, Y. Yang, C. Deng, and H. Huang, "Unsupervised deep generative adversarial hashing network," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 3664–3673.

[36] T. Chen, S. Kornblith, M. Norouzi, and G. E. Hinton, "A simple framework for contrastive learning of visual representations," in *Proc. ICML*, in Proceedings of Machine Learning Research, vol. 119, 2020, pp. 1597–1607.

[37] C. Sun, X. Song, F. Feng, W. X. Zhao, H. Zhang, and L. Nie, "Supervised hierarchical cross-modal hashing," in *Proc. 42nd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jul. 2019, pp. 725–734.

[38] Y.-W. Zhan, X. Luo, Y. Wang, and X.-S. Xu, "Supervised hierarchical deep hashing for cross-modal retrieval," in *Proc. 28th ACM Int. Conf. Multimedia*, Oct. 2020, pp. 3386–3394.

[39] J. Yan, L. Luo, C. Deng, and H. Huang, "Unsupervised hyperbolic metric learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 12465–12474.

[40] F. Lin, B. Bai, K. Bai, Y. Ren, P. Zhao, and Z. Xu, "Contrastive multi-view hyperbolic hierarchical clustering," in *Proc. 31st Int. Joint Conf. Artif. Intell.*, Jul. 2022, pp. 3250–3256.

[41] B. Dhingra, C. Shallue, M. Norouzi, A. Dai, and G. Dahl, "Embedding text in hyperbolic spaces," in *Proc. 12th Workshop Graph-Based Methods for Natural Lang. Process. (TextGraphs-12)*. New Orlean, LA, USA: Association for Computational Linguistics, 2018, pp. 59–69.

[42] I. Chami, A. Wolf, D.-C. Juan, F. Sala, S. Ravi, and C. Ré, "Low-dimensional hyperbolic knowledge graph embeddings," in *Proc. 58th Annu. Meeting Assoc. Comput. Linguistics*, 2020, pp. 6901–6914.

[43] I. Chami, Z. Ying, C. Ré, and J. Leskovec, "Hyperbolic graph convolutional neural networks," in *Proc. NeurIPS*, 2019, pp. 4869–4880.

[44] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 9726–9735.

[45] X. Chen and K. He, "Exploring simple Siamese representation learning," in *Proc. IEEE Comput. Soc. Conf. Comput. Vision Pattern Recognit.*, Jun. 2021, pp. 15750–15758.

[46] J. Li, P. Zhou, C. Xiong, and S. C. H. Hoi, "Prototypical contrastive learning of unsupervised representations," 2020, *arXiv:2005.04966*.

[47] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin, "Unsupervised learning of visual features by contrasting cluster assignments," 2020, *arXiv:2006.09882*.

[48] M. Xu, H. Wang, B. Ni, H. Guo, and J. Tang, "Self-supervised graph-level representation learning with local and global structure," in *Proc. ICML*, in Proceedings of Machine Learning Research, vol. 139, 2021, pp. 11548–11558.

[49] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.

[50] A. A. Ungar, *A Gyrovector Space Approach to Hyperbolic Geometry* (Synthesis Lectures on Mathematics & Statistics). San Rafael, CA, USA: Morgan & Claypool, 2009.

[51] R. Sarkar, "Low distortion Delaunay embedding of trees in hyperbolic plane," in *Proc. 19th Int. Symp. Graph Drawing (GD)*, in Lecture Notes in Computer Science, vol. 7034. New York, NY, USA: Springer, 2011, pp. 355–366.

[52] Ç. Gülçehre et al., "Hyperbolic attention networks," in *Proc. ICLR*, 2019. [Online]. Available: https://openreview.net/forum?id=rJxHsjRqFQ

[53] 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.

[54] J. Xia, L. Wu, G. Wang, J. Chen, and S. Z. Li, "ProGCL: Rethinking hard negative mining in graph contrastive learning," in *Proc. ICML*, in Proceedings of Machine Learning Research, vol. 162, 2022, pp. 24332–24346.

[55] J. Grill et al., "Bootstrap your own latent—A new approach to self-supervised learning," in *Proc. NIPS*, 2020, pp. 21271–21284.

[56] H. Zhu, M. Long, J. Wang, and Y. Cao, "Deep hashing network for efficient similarity retrieval," in *Proc. AAAI*. Washington, DC, USA: AAAI Press, 2016, pp. 2415–2421.

[57] E. Yang, T. Liu, C. Deng, W. Liu, and D. Tao, "DistillHash: Unsupervised deep hashing by distilling data pairs," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 2946–2955.

[58] R.-C. Tu, X.-L. Mao, and W. Wei, "MLS3RDUH: Deep unsupervised hashing via manifold based local semantic similarity structure reconstructing," in *Proc. 29th Int. Joint Conf. Artif. Intell.*, Jul. 2020, pp. 3466–3472.

[59] S. Jin, H. Yao, Q. Zhou, Y. Liu, J. Huang, and X. Hua, "Unsupervised discrete hashing with affinity similarity," *IEEE Trans. Image Process.*, vol. 30, pp. 6130–6141, 2021.

[60] Z. Ma, W. Ju, X. Luo, C. Chen, X.-S. Hua, and G. Lu, "Improved deep unsupervised hashing via prototypical learning," in *Proc. 30th ACM Int. Conf. Multimedia*, Oct. 2022, pp. 659–667.

[61] X. Zhang, X. Wang, and P. Cheng, "Unsupervised hashing retrieval via efficient correlation distillation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 33, no. 7, pp. 3529–3541, Jul. 2023.

[62] 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*, Jun. 2009, pp. 248–255.

[63] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Univ. Toronto, Toronto, ON, Canada, Tech. Rep., 2009.

[64] M. J. Huiskes and M. S. Lew, "The MIR Flickr retrieval evaluation," in *Proc. 1st ACM Int. Conf. Multimedia Inf. Retr.* New York, NY, USA: ACM, Oct. 2008, pp. 39–43.

[65] T.-S. Chua, J. Tang, R. Hong, H. Li, Z. Luo, and Y. Zheng, "NUS-WIDE: A real-world web image database from national university of Singapore," in *Proc. ACM Int. Conf. Image Video Retr.* New York, NY, USA: ACM, Jul. 2009, pp. 1–9.

[66] Z. Cao, M. Long, J. Wang, and P. S. Yu, "HashNet: Deep learning to hash by continuation," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 5609–5618.

[67] Y. Cao, M. Long, B. Liu, and J. Wang, "Deep Cauchy hashing for Hamming space retrieval," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 1229–1237.

[68] A. Paszke et al., "PyTorch: An imperative style, high-performance deep learning library," in *Proc. NeurIPS*, Dec. 2019, pp. 8024–8035.

[69] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. ICLR*, 2015.

[70] A. Dosovitskiy et al., "An image is worth 16 × 16 words: Transformers for image recognition at scale," in *Proc. ICLR*, 2021.

[71] Z. Liu et al., "Swin Transformer: Hierarchical vision transformer using shifted windows," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 9992–10002.

[72] A. Radford et al., "Learning transferable visual models from natural language supervision," in *Proc. ICML*, in Proceedings of Machine Learning Research, vol. 139, M. Meila and T. Zhang, Eds., 2021, pp. 8748–8763.

[73] G. A. Miller, "WordNet: A lexical database for English," *Commun. ACM*, vol. 38, no. 11, pp. 39–41, Nov. 1995.

[74] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, no. 86, pp. 2579–2605, 2008.

[75] L. McInnes, J. Healy, N. Saul, and L. Großberger, "UMAP: Uniform manifold approximation and projection," *J. Open Source Softw.*, vol. 3, no. 29, p. 861, Sep. 2018.

Rukai Wei received the B.S. degree from the School of Software Engineering, Huazhong University of Science and Technology (HUST), in 2022. He is currently pursuing the Ph.D. degree with Wuhan National Laboratory for Optoelectronics, HUST. His current research interests include information retrieval, deep learning, and machine learning.

Yu Liu (Member, IEEE) received the Ph.D. degree in computer science from Huazhong University of Science and Technology (HUST), China, in 2017. He is currently an Associate Researcher with the School of Computer Science, HUST. He has published some papers in international journals and conferences, including SIGMOD, ACM MM, IJCAI, DAC, IEEE TRANSACTIONS ON CYBERNETICS, CIKM, ICME, ICCD, and IEEE TRANSACTIONS ON MULTIMEDIA. His current research interests include machine learning, large-scale multimedia search, big data, and storage.

Jingkuan Song (Senior Member, IEEE) received the Ph.D. degree from The University of Queensland (UQ), Australia, in 2014. He is currently a Full Professor with the University of Electronic Science and Technology of China (UESTC). His research interests include large-scale multimedia retrieval, image/video segmentation, and image/video understanding using hashing, graph learning, and deep learning techniques. He is a PC Member of CVPR 2018, MM 2018, and IJCAI 2018. He was the Winner of the Best Paper Award from ICPR (2016, Mexico), the Best Student Paper Award from the Australian Database Conference (2017, Australia), and the Best Paper Honorable Mention Award (2017, Japan). He is the Guest Editor of IEEE TRANSACTIONS ON MULTIMEDIA and *WWWJ*.

Yanzhao Xie received the master's degree from the School of Software Engineering, Huazhong University of Science and Technology (HUST). He is currently pursuing the Ph.D. degree with Wuhan National Laboratory for Optoelectronics, HUST. He has published papers in international conferences and journals, including IJCAI, CIKM, ICMR, IEEE TRANSACTIONS ON MULTIMEDIA, *Neural Computing and Applications*, and *Multimedia Tools and Applications*. His current research interests include machine learning, graph neural networks, and reinforcement learning.

Ke Zhou (Member, IEEE) received the B.S., M.S., and Ph.D. degrees from Huazhong University of Science and Technology (HUST) in 1996, 1999, and 2003, respectively. He is currently a Professor with HUST. He has published more than 100 papers in famous academic journals and conferences, including SIGMOD, VLDB, FAST, ATC, DAC, ACM MM, *TOS*, and IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS. His main research interests include network and cloud storage, data security and service, and parallel I/O.