ECAI 2023
K. Gal et al. (Eds.)

© 2023 The Authors.
This article is published online with Open Access by IOS Press and distributed under the terms of the Creative Commons Attribution Non-Commercial License 4.0 (CC BY-NC 4.0).
doi:10.3233/FAIA230626

Deep Unsupervised Hashing with Hyperbolic Multi-Structure Learning

Chuang Zhao, Hefei Ling*, Yuxuan Shi, Jiazhong Chen and Qiang Cao

Huazhong University of Science and Technology

Abstract. Unsupervised hashing aims to learn a compact binary hash code to represent complex image content without label information. Existing deep unsupervised hashing methods typically first employ extracted image embeddings to construct semantic similarity structures and then map the images into compact hash codes while preserving the semantic similarity structure. However, the limited representation power of embeddings in Euclidean space and the inadequate exploration of the similarity structure in current methods often result in poorly discriminative hash codes. In this paper, we propose a novel method called **Hyperbolic Multi-Structure Hashing (HMSH)** to address these issues. Specifically, to increase the representation power of embeddings, we propose to map embeddings from Euclidean space to hyperbolic space and use the similarity structure constructed in hyperbolic space to guide hash learning. Meanwhile, to fully explore the structural information, we investigate four kinds of data structures, including local neighborhood structure, global clustering structure, inter/intra-class variation and variation under perturbation. Different data structures can complement each other, which is beneficial for hash learning. Extensive experimental results on three benchmark image datasets show that HMSH significantly outperforms state-of-the-art unsupervised hashing methods for image retrieval.

1 Introduction

The explosive growth of multimedia content, including all kinds of text, image and video data, poses a huge challenge to large-scale information retrieval systems. To meet the need for low storage cost and efficient retrieval efficiency, hashing techniques convert high-dimensional data into compact binary hash codes while preserving the semantic structure between data. Due to the powerful representation ability of deep learning, hashing methods based on deep learning have achieved good performance in recent years. Compared with shallow hash methods that use handcrafted features, deep hashing employs an end-to-end framework to simultaneously learn high-level semantics and binary hash codes. Deep hashing can be generally classified into two categories based on whether labeled information is used in the training process: supervised hashing [3, 7] and unsupervised hashing [6, 5, 26, 27]. Supervised hashing methods use annotated semantic information to train hashing models and achieve good performance. However, manual annotation is often time-consuming and expensive, which hinders the practical application of these methods. Therefore, unsupervised hashing has received increasing attention in recent years due to their ability to leverage widely available unlabeled data.

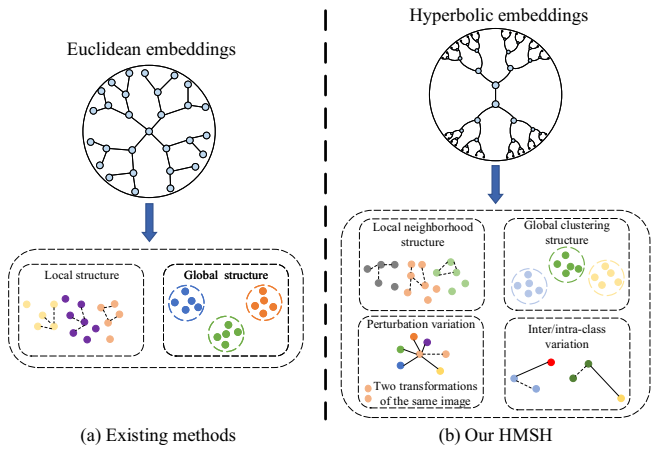


Figure 1: Illustration of current unsupervised hashing methods and our HMSH. (a) Previous unsupervised hashing methods use Euclidean embeddings to construct one or two types of structural patterns, which are then exploited to guide relationship learning within image hashing codes. (b) Our HMSH uses hyperbolic embeddings with more powerful representation capabilities to explore structural information. To fully mine the structure of image data, we study four structural patterns and fully explore the complementarity between them to guide the learning of discriminative hash codes.

For unsupervised deep hashing methods, the key is how to capture the relevance structure of the original data and preserve it well in the hash space. Therefore, most unsupervised hashing methods focus on constructing similarity structures to guide hash learning. More precisely, the Euclidean distance or cosine distance of the image embedding extracted by the deep model is used to construct the relevance structure of the original data, which is employed as guiding information to optimize the hashing network. However, almost all existing unsupervised hashing methods use Euclidean embeddings to construct similarity structures[11, 13, 35], as shown in Figure 1(a). Recent studies have demonstrated that image data in the field of computer vision exhibit a highly non-Euclidean latent anatomy [2, 10]. In this case, embeddings based on Euclidean space obviously cannot accurately represent the intrinsic structure of image data.

On the other hand, in the process of constructing the relevance structure of the original data, the existing works [32, 35, 21] mainly explore one or two types of structural information. For example, SSDH [35] utilizes pairwise neighborhood similarity based on Gaus-

* Corresponding Author. Email: lhefei@hust.edu.cn.

Please check ArXiv or contact the authors for any appendices or supplementary material mentioned in the paper.

sian estimation to construct similarity structures. MLS^{3}RDUH [32] further uses the manifold structure to reduce the noise of data points when the neighborhood structure is defined. UD2H [21] considers both the local structure and the global structure to learn the discriminative hash codes. Despite the performance improvement, it is not feasible to fully represent the complex structure implied in image data using only one or two types of structural patterns. Therefore, the exploration of structural information remains a very challenging problem for unsupervised hashing.

To solve the above problems, we propose a novel unsupervised hashing method called Hyperbolic Multi-Structure Hashing (HMSH), as shown in Figure 1(b), which fully explores the complementarity of various data structures in the hyperbolic space through multi-objective optimization, thus significantly improving the retrieval performance. Specifically, to increase the representation power of embeddings, reduce distortion, and construct more reliable similarity structures, we propose to map the embeddings extracted by the deep model from the initial Euclidean space to the hyperbolic space. Unlike the radius of Euclidean space, which grows polynomially, the radius of hyperbolic space grows exponentially. This feature enables hyperbolic spaces to increase representation power and to embed tree-structured data such as image data in the field of computer vision with low distortion. Moreover, the hyperbolic space can use low-dimensional manifolds for embedding without sacrificing model accuracy and representation power, which makes it more suitable for generating high-quality compact binary hash codes. On the other hand, to fully mine the structural information of image data, we investigate various types of structural patterns, including local neighborhood structures, global clustering structures, inter/intra-class variation and perturbation variation. Each structure has its own strengths, and we explore the complementarity among them to fully express the complicated structures implied in the image data. After that, we maximally preserve the constructed structure information in Hamming space through multi-objective optimization, so as to generate discriminative binary hash codes.

With these designs, HMSH combines hyperbolic embeddings with multiple structural information to guide hash learning, which generates high-quality hash codes and significantly improves retrieval performance. Comprehensive experiments on multiple benchmark datasets demonstrate this. Our main contributions are summarized as follows:

- To enhance the representation power of embeddings and reduce distortion, we propose to project the embedding into hyperbolic space to make it more suitable for embedding tree-like data, thus building a more reliable similarity structure.
- To fully express the complex structures implied in image data, we investigate the complementarity between various types of data structures and preserve them maximally in the Hamming space to generate high-quality binary hash codes.
- Extensive experiments validate that our HMSH outperforms state-of-the-art unsupervised methods on three widely used image retrieval benchmarks.

2 Related Work

Deep unsupervised hashing. In recent years, many deep unsupervised hashing methods learn binary hash codes by reconstructing the semantic similarity structure of the original data in the hash space. SSDH [35] constructs the semantic structure according to the Gaussian distribution of the cosine distance between features. BGAN [29] constructs the similarity structure matrix by the nearest neighbor structure and uses generative adversarial networks to reconstruct the input image. In addition, some work further improve performance by building more accurate semantic structures. DistillHash [36] further enhances semantic structures by extracting data pairs with confident semantic similarity relationships. MLS^{3}RDUH [32] obtains a similarity matrix by utilizing the manifold structure, which reduces the noise in the semantic structure. Moreover, some works learn hash codes through clustering and other techniques. CUDH [11] explores the global structure of the data by k-means clustering, resulting in binary codes with spatial structure preservation. UTH [13] forms a triplet training set by introducing image rotation to learn more discriminative binary representations. UD2H [21] constructs semantic structures from both local and global aspects to guide hashing learning. Recently, inspired by the success in the field of unsupervised representation [12, 4], some works introduce contrastive learning to obtain more discriminative hash codes. CIBHash [23] combines contrastive learning with deep hashing to learn hash codes under the information bottleneck (IB) principle. MeCoQ [34] explores the combination of contrastive learning and deep quantization, and further improves performance with a quantization code memory and a debiased technique. However, most of the above methods explore the structural information of image data from one or two perspectives, which do not accurately express the relevant structure of image data, resulting in poor results. In order to explore more reliable structures, we investigate four types of structural patterns to fully represent the complex structures implied in image data.

Hyperbolic embeddings. Hyperbolic embeddings have been widely applied to computer vision [14] and natural language processing [31] tasks due to their high capacity for data modeling. To the best of our knowledge, work [22] is the first to propose learning embeddings using the Poincaré model (a model in hyperbolic space). It is proved in [22] that Poincaré embedding can greatly outperform Euclidean embedding for some complex data with latent hierarchies, especially in representation capability. In computer vision, hyperbolic space has been applied to some tasks such as few-shot image classification [9], image segmentation [1] and point cloud classification [20]. Work [14] is one of the pioneer approaches for modeling images in hyperbolic space. Work [37] evaluates their model on person re-identification task and demonstrates superiority. Work [8] maps the images to hyperbolic space through a vision transformer and a fully connected layer, and then optimizes the losses. To compensate for the lack of representation power of Euclidean embeddings while constructing more reliable similarity structures, we explore hyperbolic embeddings in deep unsupervised hashing tasks.

3 Method

In this section, we first introduce the problem definition of unsupervised deep hashing. Then, we illustrate our HMSH in two main aspects. First, how to project feature embeddings from Euclidean space to hyperbolic space. Then, how to construct various types of data structures and learn binary hash codes guided by multi-objective optimization.

3.1 Problem Definition and Overview

In deep unsupervised hashing, $X = \{x_i\}_{i=1}^{N}$ represents an unlabeled dataset and the i^{th} image is x_i. Our goal is to learn a hash function:

$$\mathcal{H}_\theta : x_i \rightarrow b_i \in \{-1, 1\}^L, \tag{1}$$

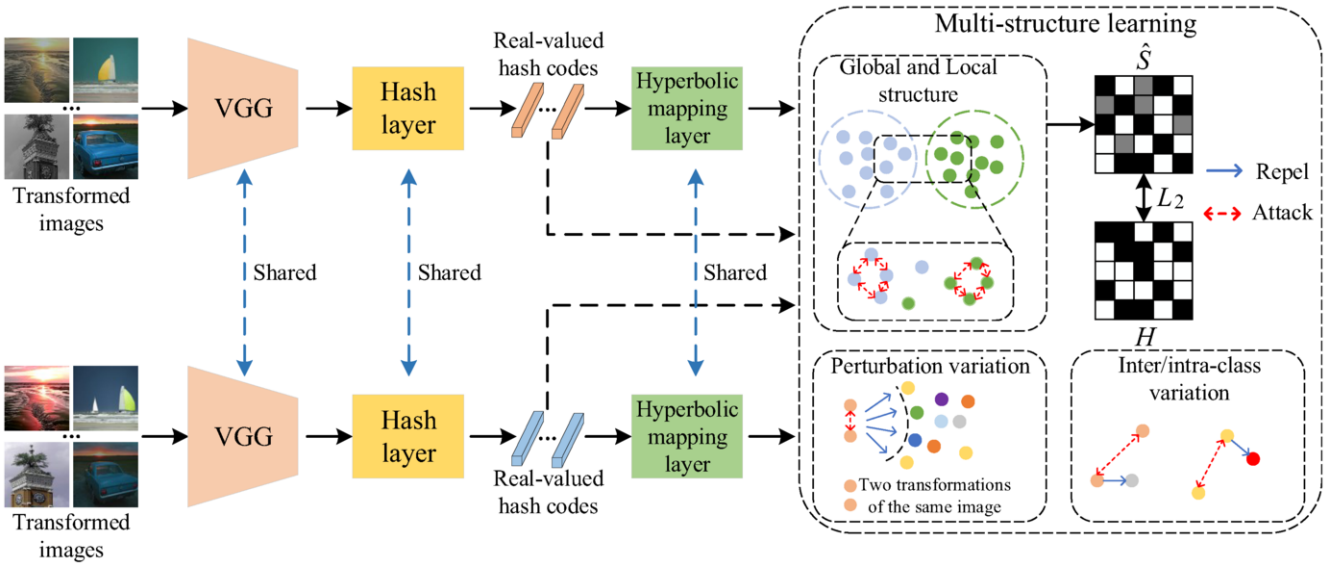


Figure 2: The framework of HMSH. During training, the original images are transformed to produce two sets of transformed images. Each transformed image goes through the VGG and a hash layer to produce a real-valued hash code, which is then mapped from Euclidean space to hyperbolic space by a hyperbolic mapping layer. Before each training epoch, HMSH extracts the hyperbolic embeddings of the training images and then KNN search and K-means clustering are exploited to capture the local and global structures of the data in the hyperbolic space. In hyperbolic space, we utilize multi-structure learning to explore the local and global structure, intra/inter-class variation and perturbation variation and learn high-quality hash codes.

which maps an image x_i to a L bits hash code b_i. To achieve this, we propose an unsupervised hashing method named hyperbolic multi-structure hashing (HMSH). The overall framework of our HMSH is shown in Figure 2.

In the training stage, we apply VGG-16 [28] followed by a hash layer to extract the real-valued hash code h_i of the image x_i. The hash layer consists of two fully-connected layers with 1024 hidden units. To obtain hyperbolic embedding z_i, then we use a hyperbolic mapping layer to project the real-valued hash code from Euclidean space to hyperbolic space. The hyperbolic mapping layer contains a fully connected layer and the exponential map defined in the Equation (4). Before each training epoch, we first extract the hyperbolic embeddings $\{z_i\}_{i=1}^N$ of all training images $\{x_i\}_{i=1}^N$. Then KNN search and K-means clustering are applied to obtain the local neighborhood structure and global clustering structure of the data. And the local structure is refined globally by Equation (8) to obtain a more accurate data structure. During training, we transform each image x_i into two views $x_i^{(1)}$ and $x_i^{(2)}$ by apply a combination of different transformations. Then we get the corresponding hash codes and hyperbolic embeddings. After that, multi-structure learning is performed using Equation (13). In the test phase, we remove the hyperbolic mapping layer and use a sign function to convert the real-valued hash code h_i to the binary hash code b_i for retrieval.

3.2 Hyperbolic Embeddings

To increase the representation power of embeddings and reduce distortion, we propose to map embeddings from Euclidean space to hyperbolic space. Different from Euclidean space with zero curvature, hyperbolic space with negative curvature tends to be more suitable for learning image embeddings.

There exist several isometric models for hyperbolic spaces, similar to [8] and [22], we choose the Poincaré ball model $(\mathbb{D}_c^n, g^{\mathbb{D}})$ as our basic model. It is well suited for gradient-based optimization (i.e., the distance function is differentiable). The model is defined by the manifold $\mathbb{D}_c^n = \{x \in \mathbb{R}^n : c\|x\|^2 < 1, c \geq 0\}$ coupled with a Riemannian metric $g^{\mathbb{D}} = \lambda_c^2 g^E$, where c is the curvature parameter, $\lambda_c = \frac{2}{1-c\|x\|^2}$ is the conformal factor that scales the local distances and $g^E = I_n$ denotes the Euclidean metric tensor. This means that in hyperbolic space, the local distance is scaled near the boundary of the ball by a factor λ_c approaching infinity. Thus, hyperbolic space exhibits the "spatial expansion" property. In Euclidean space, the volume of an object with diameter r expands polynomially with r, while in hyperbolic space, its corresponding volume exponentially expands with r. This property of hyperbolic spaces allows us to efficiently embed complex data even in low dimensions, which is precisely reflected in the embedding theorem for trees and complex nets [24].

Since hyperbolic space is not a vector space in the traditional sense, we cannot use standard operations such as addition, multiplication, etc. To solve this problem, we can generalize many standard operations to hyperbolic spaces by exploiting the formalism of the Möbius gyrovector space. Following [8], we can define the following operations for hyperbolic spaces:

Möbius addition. For two vectors $x, y \in \mathbb{D}_c^n$, their addition is defined as:

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

Hyperbolic distance. The hyperbolic distance between $x, y \in \mathbb{D}_c^n$ is defined as:

$$D_{hyp}(x, y) = \frac{2}{\sqrt{c}}\text{arctanh}(\sqrt{c}\| - x \oplus_c y\|). \quad (3)$$

In particular, when $c \to 0$, the Equation (3) reduces to Euclidean distance $\lim_{c\to 0} D_{hyp}(x, y) = 2\|x - y\|$.

Exponential map. Before performing operations in the hyperbolic space, we need to define a bijection that maps vectors from Euclidean space to hyperbolic space. When mapping from Euclidean space to

the Poincaré model of hyperbolic geometry, such a mapping is called an exponential map, while its inverse is called a logarithmic map [14]. The exponential map is a function $\exp_x^c \colon \mathbb{R}^n \to \mathbb{D}_c^n$ defined as:

$$\exp_x^c(v) = x \oplus_c \left(\tanh\left(\sqrt{c} \frac{\lambda_x^c \|v\|}{2} \right) \frac{v}{\sqrt{c}\|v\|} \right). \tag{4}$$

In practice, we follow the settings of [14] and [8] with base point $x = 0$, so that the formula is less cumbersome and empirically has little effect on the results. In the training process, we project the training samples into the hyperbolic space through the above map. Then we build the relevance structure and optimize the model based on the distance metric of hyperbolic space.

3.3 Multi-structure Learning

To fully explore the complex structure implied in the image data, we investigate various types of structural patterns, including local neighborhood structure, global clustering structure, inter/intra-class variation and perturbation variation.

Local neighborhood structure. From a local perspective, the local neighborhood structure can capture pairwise similarity information. To achieve this, we first need to construct a local structure graph $\bar{S}$, where each element represents the similarity of a pair of images. In this paper, we employ two local structure exploration strategies: threshold filtering and KNN search.

Threshold filtering. We set a threshold t. Then we consider image pairs whose distance is less than t to be similar and those whose distance is greater than t to be dissimilar. Specifically, the local neighborhood structure $\bar{S}$ is defined as:

$$\bar{S}_{ij} = \begin{cases} 1, & \text{if } D\left(z_i, z_j\right) \leq t \\ -1, & \text{if } D\left(z_i, z_j\right) > t \end{cases}, \tag{5}$$

where D is a distance (D_{hpy} or D_{cos}). The hyperbolic distance D_{hpy} is defined by Equation (3). And the cosine distance D_{cos} between z_i and z_j is defined as: $D_{cos}\left(z_i, z_j\right) = 1 - \frac{z_i \cdot z_j}{\|z_i\|_2 \|z_j\|_2}$.

KNN search. In addition, we also use KNN search to construct the local neighborhood structure. For two images, if they are one of the K-Nearest Neighbors of each other, then the two images are considered to be similar. From this, we can construct the local neighborhood structure $\bar{S}$:

$$\bar{S}_{ij} = \begin{cases} 1, & \text{if } z_i \in nn_p(z_j) \& z_j \in nn_p(z_i) \\ -1, & \text{otherwise} \end{cases}, \tag{6}$$

where $nn_p(z_i)$ denotes the set of p nearest neighbors of z_i.

Global clustering structure. Global clustering can intuitively reflect the global structure of data. We utilize K-means clustering to divide all samples into k classes, thus establishing the global structure $\widetilde{S}$ of data. Specifically, we consider z_i and z_j to be similar if they are in the same cluster, otherwise, they are dissimilar. Formally, we define global structure $\widetilde{S}$ as follow:

$$\widetilde{S}_{ij} = \begin{cases} 1 & \text{if } z_i \simeq z_j \\ -1 & \text{otherwise} \end{cases}, \tag{7}$$

where $z_i \simeq z_j$ indicates that z_i and z_j are in the same cluster.

Combining local and global structures. KNN search or threshold filtering can provide a local perspective for similarity structure construction, but the local neighborhood structure may introduce some noisy data points due to the lack of global supervision. K-means clustering can provide a global distribution perspective for similarity structure construction, but there may be some false semantic similar pairs at the boundary points of clustering. To take full advantage of both, we can combine the local and global structures to construct a more accurate data structure:

$$\hat{S}_{ij} = \begin{cases} 1 & \bar{S}_{ij} = 1 \& \widetilde{S}_{ij} = 1 \\ -1 & \bar{S}_{ij} = -1 \& \widetilde{S}_{ij} = -1 \\ 0 & \text{otherwise} \end{cases}. \tag{8}$$

To preserve the learned semantic structure, we minimize the L_2 loss between the hash code semantic structure and the semantic structure constructed above. Formally,

$$\mathcal{L}_s = \frac{1}{M^2} \sum_{i=1}^{M} \sum_{j=1}^{M} \left(H_{ij} - S_{ij}\right)^2, \tag{9}$$

where $H_{ij} = \frac{1}{L} b_i^\top b_j$ denotes the hash code semantic structure and $b_i = sign(h_i)$. S_{ij} can be replaced by $\bar{S}_{ij}$, $\widetilde{S}_{ij}$ or $\hat{S}_{ij}$. M is the number of samples in a mini-batch.

Inter/intra-class variation. Both the local neighborhood structure and the global clustering structure above reflect the semantic structure of the data space through pairwise similarity. However, since the ranking information is not fully utilized, the performance of pairwise similarity-based approaches may be suboptimal in image retrieval. As a complement to the pairwise similarity structure, we additionally use a triplet ranking loss to achieve large inter-class variation and small intra-class variation among samples. Several past studies on image hashing have also demonstrated the validity of the triplet similarity relation [15]. We use the results obtained by clustering as label information, and thus our triplet hashing loss [25] formulation is as follows:

$$\mathcal{L}_t = \max\left(0, D\left(z_i, z_i^+\right) - D\left(z_i, z_i^-\right) + m\right), \tag{10}$$

where the (z_i, z_i^+) image pair is positive pair and (z_i, z_i^-) is negative pair. m is the distance margin between positive and negative data pairs.

Perturbation variation. Existing unsupervised hashing methods rarely consider the robustness of the hash model [19, 16], which can affect the quality of the hash code under perturbation. By introducing perturbation variation, we aim to constrain the hash codes generated from the same image under different transformations to be consistent, so that the hash model has better robustness. Contrastive learning has been shown to produce high-quality representations for downstream tasks [4, 12], which can meet our needs well. Motivated by this, we sample a mini-batch of M images, producing $2M$ random transformed images. We treat the two transformations $\{x_i, x_j\}$ of the same image in the batch as positive sample pairs and the remaining $2M - 2$ images as negative samples. To constrain the consistency of hash codes under perturbation, our perturbation consistency loss is:

$$\mathcal{L}_c = \frac{1}{2M} \sum_{k=1}^{M} \left(\ell(2k-1, 2k) + \ell(2k, 2k-1)\right), \tag{11}$$

$$\ell(i,j) = -\log \frac{\exp\left(-D\left(z_i, z_j\right)/\tau\right)}{\exp\left(-D\left(z_i, z_j\right)/\tau\right) + \sum_{k \neq i,j}^{2M} \exp\left(-D\left(z_i, z_k\right)/\tau\right)}, \tag{12}$$

where τ is a temperature parameter.

We investigate four different structural patterns in hash learning, including local neighborhood structure, global clustering structure,

Algorithm 1 Learning algorithm for HMSH

Input: Training images $X = \{x_i\}_{i=1}^N$, length of hash codes L, hyper-parameters, batch size M.
Output: Parameters θ of hash function $\mathcal{H}(\cdot)$, hash codes $B = \{b_i\}_{i=1}^N$.
Procedure:

```
Initialize parameters θ for hash function H(·) and parameters θ_h for hyperbolic mapping layer.
repeat
    Extracte the hyperbolic embeddings {z_i}_{i=1}^N of the images {x_i}_{i=1}^N.
    Construct the similarity structure Ŝ by the equation (8).
    Sample M images and then obtain the transformed images.
    Extract the corresponding hash codes and hyperbolic embeddings.
    Calculate the loss function by the equation (13).
    Update the parameters θ and θ_h through back propagation.
until convergence
```

inter/intra class variation and perturbation variation. First, the local neighborhood is commonly used in existing hashing methods, which reflects the pairwise similarity between images. However, the neighborhood does not reflect the global distribution of the samples and may contain some noise. As a complement, global clustering can obtain the statistics of the whole sample set. And we also combine local and global structures to construct a more accurate data structure. In addition, both neighborhood and clustering do not focus on promoting intra-class compactness and inter-class separability, which have proven useful in recognition and retrieval tasks [25]. Therefore, we adopt triplet loss to keep small intra-class variation and large inter-class variation. Finally, to learn high-quality hash codes, we incorporate contrastive learning to improve the robustness of hash codes under perturbations. Furthermore, we conduct experiments on various combinations of four structures and study their complementarity.

Model optimization. Combining the proposed four structures, we use multi-objective optimization to construct the overall loss:

$$\mathcal{L} = \mathcal{L}_s(x_i^{(1)}) + \mathcal{L}_s(x_i^{(2)}) + \lambda(\mathcal{L}_t(x_i^{(1)}) + \mathcal{L}_t(x_i^{(2)})) + \mathcal{L}_c(x_i), \tag{13}$$

where $x_i^{(1)}$ and $x_i^{(2)}$ mean two transformed samples of x_i. And the equilibrium coefficient λ is used to balance different losses. At the training stage, to tackle the ill-posed gradient of sign function, we adopt the $tanh(\cdot)$ to approximate the results of $sign(\cdot)$. The whole learning procedure are shown in Algorithm 1.

4 EXPERIMENTS

In this section, we conduct extensive experiments on various public datasets compared with state-of-the-art unsupervised hashing methods to verify the superiority of our HMSH. More details and experimental results can be found in the Appendix.

4.1 Datasets and Settings

CIFAR-10 is a dataset containing 60,000 images divided into 10 categories, where each category contains 6,000 images of size 32×32. We randomly select 1000 images from each category as the query set and use the remaining images as the database, and 500 images from each category as the training set.

NUS-WIDE is a multi-label dataset that contains nearly 270,000 images with 81 labels, and each image belongs to one or more labels. Following [34], we use the subset with images from 21 most frequent categories. We randomly select 2100 images from each category as query images, and the remaining images form database and the training set.

MSCOCO is a large-scale dataset for object detection, segmentation and image annotation. It contains about 82,783 training images and 40,504 validation images, where each image is labeled by some of the 80 categories. Following [23], we use a traditional set containing 12,2218 images. We randomly select 5,000 images as query images and use other images as database, and 10,000 images from the database as a training set.

4.2 Evaluation Protocol

We adopt the typical metric Mean Average Precision (MAP) to measure the quality of obtained hash codes. Following [23, 34], we adopt MAP@1000 for CIFAR-10, MAP@5000 for NUS-WIDE and MSCOCO. We compare HMSH with several state-of-the-art unsupervised hashing methods, including DeepBit [18], SGH [5], BGAN [29], BinGAN [38], SSDH [35], GreedyHash [30], DVB [26], TBH [27], MLS^{3}RDUH [32], BiHash [17], CIBhash [23] and MeCoQ [34]. We carefully collect their results from related literature.

4.3 Implementation Details

During training, we resize all images to 224×224 as input. Following [23], we employ data augmentation strategies including random cropping, random color distortions and Gaussian blur, etc., to transform the image into different views. We adopt a pre-trained VGG-16 network [28] as the backbone network. The hash layer transforms the image features into L-dimensional real-valued hash codes. After that, our hyperbolic mapping layer maps the real-valued hash codes into 128-dimensional hyperbolic embeddings on the Poincaré ball. We implement our model on PyTorch and use the optimizer Adam for optimization. We set the learning rate to be set to 0.001 and the batch size to 64. The curvature parameter c is set to 0.01. The equilibrium coefficient λ is set to 0.01, the margin m is set to 0.5 and the temperature parameter τ is set to 0.5. And we select the number of neighbors p and the number of clusters k via cross-validation.

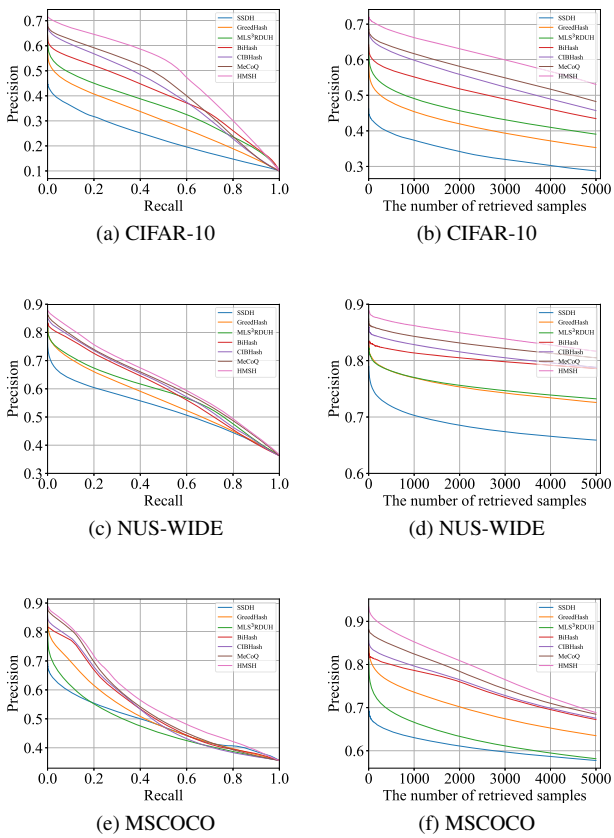
4.4 Result Analysis

In Table 1, we report the retrieval results of our HMSH and existing unsupervised hashing methods on datasets CIFAR-10, NUS-WIDE and MSCOCO with code lengths of 16, 32 and 64. As shown in Table 1, on all three datasets, our HMSH significantly outperforms all competing methods for hash codes of different lengths. Specifically, compared to the strong competitor CIBHash, the average retrieval performance of HMSH on CIFAR-10, NUS-WIDE and MSCOCO is improved by an average of 5.43%, 2.80% and 5.23%, respectively. Notably, compared to the current best-performing method MeCoQ, the average retrieval performance of HMSH on the three datasets is improved by 3.17%, 1.33%, and 2.80%, respectively. These results fully demonstrate that our HMSH can adequately capture the complex semantic structure and utilize it to generate more discriminative hash codes.

To sufficiently reveal the overall performance of HMSH, we report the PR curve and Precision@5000 curves on three datasets in Figure 3. As can be seen, our HMSH consistently outperforms all competing

Table 1: Mean Average Precision (MAP) results for different number of bits on CIFAR-10, NUS-WIDE and MSCOCO.

Method	Reference	CIFAR-10			NUS-WIDE			MSCOCO		
		16bits	32bits	64bits	16bits	32bits	64bits	16bits	32bits	64bits
DeepBit	CVPR16	0.194	0.249	0.277	0.392	0.403	0.429	0.407	0.419	0.430
SGH	ICML17	0.435	0.437	0.433	0.593	0.590	0.607	0.594	0.610	0.618
BGAN	AAAI18	0.525	0.531	0.562	0.684	0.714	0.730	0.645	0.682	0.707
BinGAN	NIPS18	0.476	0.512	0.520	0.654	0.709	0.713	0.651	0.673	0.696
SSDH	IJCAI18	0.333	0.383	0.401	0.580	0.593	0.610	0.552	0.591	0.620
GreedyHash	NIPS18	0.448	0.473	0.501	0.633	0.691	0.731	0.582	0.668	0.710
DVB	IJCV19	0.403	0.422	0.446	0.604	0.632	0.665	0.570	0.629	0.623
TBH	CVPR20	0.532	0.573	0.578	0.717	0.725	0.735	0.706	0.735	0.722
MLS^{3}RDUH	IJCAI20	0.557	0.581	0.594	0.713	0.727	0.750	0.716	0.731	0.737
BiHash	AAAI21	0.500	0.520	0.554	0.769	0.783	0.799	0.722	0.765	0.772
CIBHash	IJCAI21	0.590	0.622	0.641	0.790	0.807	0.815	0.737	0.760	0.775
MeCoQ	AAAI22	0.629	0.641	0.651	0.802	0.822	0.832	0.762	0.783	0.800
HMSH	**Ours**	**0.656**	**0.672**	**0.688**	**0.819**	**0.835**	**0.842**	**0.794**	**0.812**	**0.823**

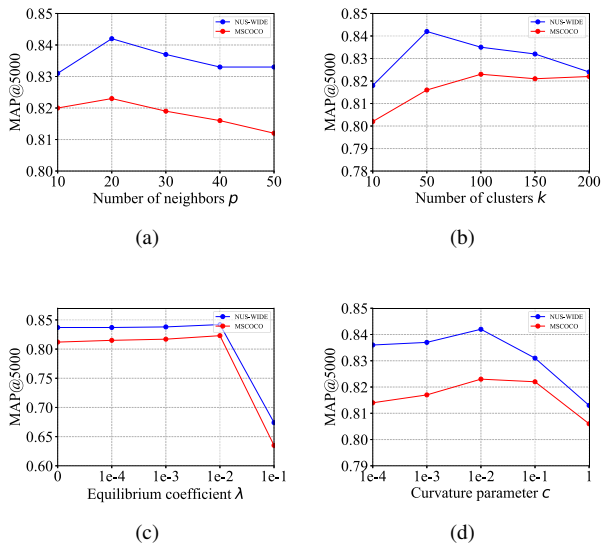


(a) CIFAR-10 (b) CIFAR-10

(c) NUS-WIDE (d) NUS-WIDE

(e) MSCOCO (f) MSCOCO

Figure 3: Precision-recall (PR) curves and Precision@top-N curves with code length 64 on CIFAR-10, NUS-WIDE and MSCOCO



(a) (b)

(c) (d)

Figure 4: Parameter analysis with 64 bits hash codes.

methods on all three datasets. In addition, HMSH also has higher accuracy for the same number of returned images. All these results imply that our HMSH has a stable and superior performance on three datasets.

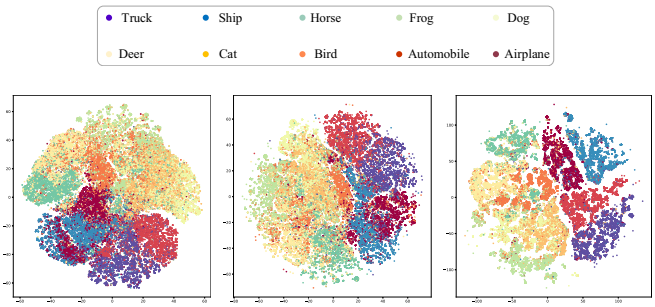
4.5 Ablation Study

To investigate the impact of the different components of the proposed method, we set up several variants of HMSH to analyze the contribution of the components. In Table 2, *hyp* means hyperbolic embeddings; *tf* means threshold filtering; *knn* means KNN search; *gcs* means global clustering structure; *iiv* means inter/intra-class variation; *pv* means perturbation variation. V_1: Using the local structure constructed by hyperbolic embeddings and threshold filtering to optimize the hash model.(Here t is set to 0.3.) V_2: Using the local structure constructed by hyperbolic embeddings and KNN search to optimize the hash model. V_3: Combining local (KNN search) and global structures in hyperbolic space to optimize the hash model. V_4: Combining local structure, global structure and inter/intra-class variation in hyperbolic space to optimize the hash model. V_5: Using perturbation variation in hyperbolic space to optimize the hash model. V_6: In Euclidean space combining all four structures to optimize the hash model. V_7: In hyperbolic space combining all four structures to optimize the hash model.

The ablation results are shown in Table 2. Different from V_1 using threshold filtering, V_2 utilizes KNN search to construct the local structure and optimize the network. As can be seen from the table, V_2 performs significantly better than V_1, which indicates that similar or dissimilar image pairs cannot be accurately selected by using threshold filtering only. Compared to V_2, V_3 combines local and global structures to further increase performance, which shows that the noise in the local structure can be reduced by global clustering. On the basis of V_3, V_4 add inter/intra-class variation with certain

Table 2: Ablation study results for different number of bits on CIFAR-10, NUS-WIDE and MSCOCO.

	Components						CIFAR-10			NUS-WIDE			MSCOCO		
	hyp	*tf*	*knn*	*gcs*	*iiv*	*pv*	16bits	32bits	64bits	16bits	32bits	64bits	16bits	32bits	64bits
V_1	✓	✓					0.435	0.472	0.523	0.765	0.775	0.782	0.605	0.679	0.702
V_2	✓		✓				0.488	0.546	0.582	0.778	0.804	0.812	0.748	0.768	0.785
V_3	✓		✓	✓			0.536	0.568	0.610	0.794	0.811	0.823	0.767	0.785	0.803
V_4	✓		✓	✓	✓		0.545	0.579	0.617	0.800	0.815	0.826	0.774	0.793	0.809
V_5	✓					✓	0.601	0.631	0.653	0.798	0.816	0.823	0.743	0.764	0.781
V_6			✓	✓	✓	✓	0.635	0.656	0.667	0.807	0.823	0.834	0.781	0.800	0.811
V_7	✓		✓	✓	✓	✓	**0.656**	**0.672**	**0.688**	**0.819**	**0.835**	**0.842**	**0.794**	**0.812**	**0.823**



(a) BiHash (b) CIBHash (c) HMSH

Figure 5: The t-SNE visualizations of 64 bits hash codes generated by BiHash, CIBHash and HMSH on CIFAR-10.

performance improvement, which shows that inter/intra-class variation can generate a beneficial effect on our hash model. V_5 only uses the contrastive learning framework to constrain the consistency of hash codes for the same image under different perturbations, which achieves good performance and demonstrates the importance of hash code robustness for retrieval performance. And the embedding in hyperbolic space (V_7) has stronger representation power compared to the embedding in Euclidean space (V_6), thus the model performance is further improved. Our final HMSH (V_7) combines all structures in hyperbolic space to obtain the best performance. This proves that different structures are complementary to each other, and also demonstrates the feasibility of the proposed multi-structure learning.

4.6 *Sensitivity Study*

In this section, we further study the performance of HMSH under different settings of number of neighbors p, number of clusters k, equilibrium coefficient λ and curvature parameter c. The results are shown in Figure 4. In Figure 4(a), the number of neighbors p is an important index for constructing the local structure. The performance of our model decreases when p is taken too large or too small, and the model performance reaches the best when p is 20. In Figure 4(b), the number of clusters k can reflect the global structure of data. Since MSCOCO has a more complex class division, the model performance drops sharply when the number of clustering is too small. Our model achieves the best performance when the number of clustering is 50 or 100. In Figure 4(c), the equilibrium coefficient λ determines the degree of influence of inter/intra-class variation loss on the model optimization. The model performance is optimal when λ is taken as 0.01, and decreases sharply when λ is taken as 0.1. This indicates that a large proportion of inter/intra-class variation loss in the whole learning objective will adversely affect model performance. In Figure 4(d), the curvature parameter c is proportional to the radius of the Poincaré ball. Intuitively, if the value of c tends to 0, making the Poincaré ball as flat as Euclidean space. When c is between 0.001 and 0.1, our model performance has small fluctuations. When c is 1, the model performance drops sharply.

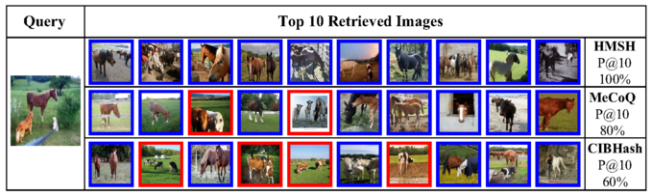


Figure 6: Examples of the top 10 retrieved images and Precision@10 on NUS-WIDE with 64 bits hash codes.

4.7 *Visualization*

t-SNE visualization. To visually investigate the performance of our HMSH, we present the t-SNE visualization [33] results of BiHash, CIBHash and HMSH with 64-bit hash codes on CIFAR-10. As shown in Figure 5, the hash codes generated by HMSH have a clearer structure. For instance, the clusters of $ship$, $automobile$, $truck$ and $airplane$ are more compact and can be easily distinguished from the HMSH, while those of BiHash and CIBHash are mixed. This indicates that the semantic structure of the hash code produced by HMSH is significantly preserved.

Visualization of retrieval results. In Figure 6, we show the top 10 images returned by HMSH, MeCoQ, and CIBHash on the NUS-WIDE dataset based on 64-bit hash codes. Specifically, "Blue" denotes the relevant results, and "red" denotes the irrelevant results. Benefiting from the powerful representation capability of hyperbolic embeddings and multi-structure learning, HMSH can provide more relevant retrieved image results. These results show that our HMSH can generate higher-quality hash codes.

5 Conclusion

In this paper, we propose a novel unsupervised hashing method called HMSH. To increase the representation capability of embeddings and construct more reliable similarity structures, we propose to map the embeddings extracted by the deep model from the initial Euclidean space to the hyperbolic space. Meanwhile, we investigate various types of structural patterns, including local neighborhood structure, global clustering structure, inter/intra-class variation and perturbation variation. We explore the complementarity among them to fully capture the complex structures implied in the image data. Extensive experiments on three datasets demonstrate the superiority of HMSH.

Acknowledgements

This work was supported in part by the Natural Science Foundation of China under Grant 61972169, in part by the Major Scientific and Technological Project of Hubei Province (2022BAA046, 2022BAA042), in part by China Postdoctoral Science Foundation 2022M711251, in part by the National key research and development program of China(2019QY(Y)0202, 2022YFB2601802), in part by the Research Programme on Applied Fundamentals and Frontier Technologies of Wuhan(2020010601012182) and the Knowledge Innovation Program of Wuhan-Basic Research.

References

[1] Mina Ghadimi Atigh, Julian Schoep, Erman Acar, Nanne Van Noord, and Pascal Mettes, 'Hyperbolic image segmentation', in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4453–4462, (2022).

[2] Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst, 'Geometric deep learning: going beyond euclidean data', *IEEE Signal Processing Magazine*, **34**(4), 18–42, (2017).

[3] Yue Cao, Mingsheng Long, Bin Liu, and Jianmin Wang, 'Deep cauchy hashing for hamming space retrieval', in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1229–1237, (2018).

[4] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton, 'A simple framework for contrastive learning of visual representations', in *International conference on machine learning*, pp. 1597–1607. PMLR, (2020).

[5] Bo Dai, Ruiqi Guo, Sanjiv Kumar, Niao He, and Le Song, 'Stochastic generative hashing', in *International Conference on Machine Learning*, pp. 913–922. PMLR, (2017).

[6] Thanh-Toan Do, Anh-Dzung Doan, and Ngai-Man Cheung, 'Learning to hash with binary deep neural network', in *European Conference on Computer Vision*, pp. 219–234. Springer, (2016).

[7] Khoa D Doan, Peng Yang, and Ping Li, 'One loss for quantization: Deep hashing with discrete wasserstein distributional matching', in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9447–9457, (2022).

[8] Aleksandr Ermolov, Leyla Mirvakhabova, Valentin Khrulkov, Nicu Sebe, and Ivan Oseledets, 'Hyperbolic vision transformers: Combining improvements in metric learning', in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7409–7419, (2022).

[9] Pengfei Fang, Mehrtash Harandi, and Lars Petersson, 'Kernel methods in hyperbolic spaces', in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10665–10674, (2021).

[10] Octavian Ganea, Gary Bécigneul, and Thomas Hofmann, 'Hyperbolic neural networks', *Advances in neural information processing systems*, **31**, (2018).

[11] Yifan Gu, Shidong Wang, Haofeng Zhang, Yazhou Yao, Wankou Yang, and Li Liu, 'Clustering-driven unsupervised deep hashing for image retrieval', *Neurocomputing*, **368**, 114–123, (2019).

[12] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick, 'Momentum contrast for unsupervised visual representation learning', in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9729–9738, (2020).

[13] Shanshan Huang, Yichao Xiong, Ya Zhang, and Jia Wang, 'Unsupervised triplet hashing for fast image retrieval', in *Proceedings of the on Thematic Workshops of ACM Multimedia 2017*, pp. 84–92, (2017).

[14] Valentin Khrulkov, Leyla Mirvakhabova, Evgeniya Ustinova, Ivan Oseledets, and Victor Lempitsky, 'Hyperbolic image embeddings', in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6418–6428, (2020).

[15] Hanjiang Lai, Yan Pan, Ye Liu, and Shuicheng Yan, 'Simultaneous feature learning and hash coding with deep neural networks', in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3270–3278, (2015).

[16] Yue Nan Li, Ping Wang, and Yu Ting Su, 'Robust image hashing based on selective quaternion invariance', *IEEE signal processing letters*, **22**(12), 2396–2400, (2015).

[17] Yunqiang Li and Jan van Gemert, 'Deep unsupervised image hashing by maximizing bit entropy', in *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 2002–2010, (2021).

[18] Kevin Lin, Jiwen Lu, Chu-Song Chen, and Jie Zhou, 'Learning compact binary descriptors with unsupervised deep neural networks', in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1183–1192, (2016).

[19] Yu A Malkov and Dmitry A Yashunin, 'Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs', *IEEE transactions on pattern analysis and machine intelligence*, **42**(4), 824–836, (2018).

[20] Antonio Montanaro, Diego Valsesia, and Enrico Magli, 'Rethinking the compositionality of point clouds through regularization in the hyperbolic space', *arXiv preprint arXiv:2209.10318*, (2022).

[21] Zhuyi Ni, Zexuan Ji, Long Lan, Yun-Hao Yuan, and Xiaobo Shen, 'Unsupervised discriminative deep hashing with locality and globality preservation', *IEEE Signal Processing Letters*, **28**, 518–522, (2021).

[22] Maximillian Nickel and Douwe Kiela, 'Poincaré embeddings for learning hierarchical representations', *Advances in neural information processing systems*, **30**, (2017).

[23] Zexuan Qiu, Qinliang Su, Zijing Ou, Jianxing Yu, and Changyou Chen, 'Unsupervised hashing with contrastive information bottleneck', in *IJCAI*, pp. 959–965, (2021).

[24] Rik Sarkar, 'Low distortion delaunay embedding of trees in hyperbolic plane', in *Graph Drawing: 19th International Symposium, GD 2011, Eindhoven, The Netherlands, September 21-23, 2011, Revised Selected Papers 19*, pp. 355–366. Springer, (2012).

[25] Florian Schroff, Dmitry Kalenichenko, and James Philbin, 'Facenet: A unified embedding for face recognition and clustering', in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 815–823, (2015).

[26] Yuming Shen, Li Liu, and Ling Shao, 'Unsupervised binary representation learning with deep variational networks', *International Journal of Computer Vision*, **127**(11), 1614–1628, (2019).

[27] Yuming Shen, Jie Qin, Jiaxin Chen, Mengyang Yu, Li Liu, Fan Zhu, Fumin Shen, and Ling Shao, 'Auto-encoding twin-bottleneck hashing', in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2818–2827, (2020).

[28] Karen Simonyan and Andrew Zisserman, 'Very deep convolutional networks for large-scale image recognition', *arXiv preprint arXiv:1409.1556*, (2014).

[29] Jingkuan Song, Tao He, Lianli Gao, Xing Xu, Alan Hanjalic, and Heng Tao Shen, 'Binary generative adversarial networks for image retrieval', in *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, (2018).

[30] Shupeng Su, Chao Zhang, Kai Han, and Yonghong Tian, 'Greedy hash: Towards fast optimization for accurate hash coding in cnn', *Advances in neural information processing systems*, **31**, (2018).

[31] Alexandru Tifrea, Gary Bécigneul, and Octavian-Eugen Ganea, 'Poincar\'e glove: Hyperbolic word embeddings', *arXiv preprint arXiv:1810.06546*, (2018).

[32] Rong-Cheng Tu, Xianling Mao, and Wei Wei, 'Mls3rduh: Deep unsupervised hashing via manifold based local semantic similarity structure reconstructing.', in *IJCAI*, pp. 3466–3472, (2020).

[33] Laurens Van der Maaten and Geoffrey Hinton, 'Visualizing data using t-sne.', *Journal of machine learning research*, **9**(11), (2008).

[34] Jinpeng Wang, Ziyun Zeng, Bin Chen, Tao Dai, and Shu-Tao Xia, 'Contrastive quantization with code memory for unsupervised image retrieval', in *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 2468–2476, (2022).

[35] Erkun Yang, Cheng Deng, Tongliang Liu, Wei Liu, and Dacheng Tao, 'Semantic structure-based unsupervised deep hashing', in *Proceedings of the 27th international joint conference on artificial intelligence*, pp. 1064–1070, (2018).

[36] Erkun Yang, Tongliang Liu, Cheng Deng, Wei Liu, and Dacheng Tao, 'Distillhash: Unsupervised deep hashing by distilling data pairs', in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 2946–2955, (2019).

[37] Liang Zheng, Yi Yang, and Alexander G Hauptmann, 'Person re-identification: Past, present and future', *arXiv preprint arXiv:1610.02984*, (2016).

[38] Maciej Zieba, Piotr Semberecki, Tarek El-Gaaly, and Tomasz Trzcinski, 'Bingan: Learning compact binary descriptors with a regularized gan', *Advances in neural information processing systems*, **31**, (2018).