remote sensing

MDPI

Article

Deep Contrastive Self-Supervised Hashing for Remote Sensing Image Retrieval

Xiaoyan Tan [1,†], Yun Zou [1,†], Ziyang Guo [2], Ke Zhou [1,*] and Qiangqiang Yuan [3]

[1] Wuhan National Laboratory for Optoelectronics, Huazhong University of Science and Technology, Wuhan 430074, China; xiaoyan_tan2022@163.com (X.T.); yunz0219@163.com (Y.Z.)
[2] School of Artificial Intelligence, Nanjing University of Information Science and Technology, Nanjing 210044, China; guo_zy@nuist.edu.cn
[3] School of Geodesy and Geomatics, Wuhan University, Wuhan 430079, China; qqyuan@sgg.whu.edu.cn
* Correspondence: zhke@hust.edu.cn
† These authors contributed equally to this work.

Abstract: Hashing has been widely used for large-scale remote sensing image retrieval due to its outstanding advantages in storage and search speed. Recently, deep hashing methods, which produce discriminative hash codes by building end-to-end deep convolutional networks, have shown promising results. However, training these networks requires numerous labeled images, which are scarce and expensive in remote sensing datasets. In order to solve this problem, we propose a deep unsupervised hashing method, namely deep contrastive self-supervised hashing (DCSH), which uses only unlabeled images to learn accurate hash codes. It eliminates the need for label annotation by maximizing the consistency of different views generated from the same image. More specifically, we assume that the hash codes generated from different views of the same image are similar, and those generated from different images are dissimilar. On the basis of the hypothesis, we can develop a novel loss function containing the temperature-scaled cross-entropy loss and the quantization loss to train the developed deep network end-to-end, resulting in hash codes with semantic similarity preserved. Our proposed network contains four parts. First, each image is transformed into two different views using data augmentation. After that, they are fed into an encoder with the same shared parameters to obtain deep discriminate features. Following this, a hash layer converts the high-dimensional image representations into compact binary codes. Lastly, a novel hash function is introduced to train the proposed network end-to-end and thus guide generated hash codes with semantic similarity. Extensive experiments on two popular benchmark datasets of the UC Merced Land Use Database and the Aerial Image Dataset have demonstrated that our DCSH has significant superiority in remote sensing image retrieval compared with state-of-the-art unsupervised hashing methods.

Keywords: remote sensing image retrieval; unsupervised hashing; self-supervised learning; contrastive learning; deep learning

Citation: Tan, X.; Zou, Y.; Guo, Z.; Zhou, K.; Yuan, Q. Deep Contrastive Self-Supervised Hashing for Remote Sensing Image Retrieval. *Remote Sens.* **2022**, *14*, 3643. https://doi.org/10.3390/rs14153643

Academic Editor: Filiberto Pla

Received: 30 May 2022
Accepted: 26 July 2022
Published: 29 July 2022

1. Introduction

In recent years, various remote sensing images have emerged with the rapid development of sensors and aerospace technology [1]. Therefore, searching for desired images from such a big data archive has become a challenging task [2,3]. At present, remote sensing image retrieval, which aims at fulfilling the task in an effective and efficient manner, has received significant attention in the remote sensing community.

Remote sensing image retrieval methods are divided into two paradigms: text-based and content-based image retrieval. In the early days, since text-based image retrieval approaches [4,5] search images by matching text keywords of data, they require domain experts to annotate images with high-quality descriptors in advance. However, remote sensing images have high complexity and cannot be fully represented by only a few

keywords. Instead of manually describing each image, content-based image retrieval [6,7] can directly take a raw image as input and automatically extract its visual information.

Generally, content-based image retrieval includes two main steps: feature extraction and similarity ranking. In the first step, we aim to explore an effective algorithm to represent all database RS images and query images by discriminative visual features. In the second step, based on these features, we measure the similarity between the query and each image in the retrieval database and then return a ranked list of similar images. For feature extraction, numerous methods have been proposed to describe image content. For instance, early studies focus on low-level features based on three aspects: spectrum [8], texture [9,10] and shape [11]. These features tend to be sensitive to scale, rotation or illumination, leading to poor retrieval accuracy. Then a variety of encoding methods, such as locally aggregated descriptors [12] and bag-of-visual words [13], have been proposed to improve robustness by aggregating low-level features into holistic representations. All of the above features are usually real-valued vectors with thousands of dimensions in the above methods, resulting in high storage costs. In addition, retrieval speed is inefficient since it is time-consuming to retrieve similar images in the high-dimensional Euclidean space. Both of them would be a barrier in real-world retrieval practice, especially in the era of big data.

To address the above issue, hash-based retrieval methods (also called hashing) have been developed to compress the images into compact binary codes rather than only high-dimensional features. Besides lowering storage costs, short binary representations also shorten the retrieval time due to searching in low dimensional Hamming space rather than high dimensional Euclidean space. The key to hash-based retrieval [2,14–16] is to learn hash functions, which need to preserve the similarity relation from image space to Hamming space. In other words, similar images are projected to nearby hash codes, and dissimilar images are far away. Hashing can be generally divided into supervised [17–22] and unsupervised [23–26] based on whether predefined labels are used during training.

Deep learning has achieved dramatic success in various computer vision tasks, such as image classification [27–29]. Recently, deep learning has also been introduced to hashing methods for image retrieval, dubbed deep hashing. It also falls into unsupervised and supervised. Deep supervised hashing methods construct an end-to-end supervised deep model to generate hash code by modifying existing convolutional neural networks such as AlexNet, VGG, GoogLenet and ResNet. To make the generated hash codes preserve semantic similarity, it is essential to utilize predefined manual labels as supervision information. Pairwise or triplet-wise labels are the two representative types. For instance, Refs. [21,30,31] provide pairwise labels that show whether two images are similar or dissimilar. Based on this type of label, pairwise cross-entropy loss is developed to train deep networks to preserve semantic similarity. Refs. [32–35] construct a triplet metric learning deep network with the aid of triplet-wise labels. Each of them contains three images: an anchor image, a positive image and a negative image, and indicates a relative relation: an anchor is more similar to the positive image than the negative.

Even deep supervised hashing methods have achieved promising performance; they are heavily dependent on supervision information, which is scarce and expensive, especially in large-scale data archives. Thus, unsupervised deep hashing, adopting deep neural networks to generate semantically-preserving hash code with unlabeled images, will be the focus of this study. Existing deep unsupervised hashing methods use pre-trained convolutional neural networks that are trained on a natural dataset of ImageNet to extract deep features of images and a non-deep structure for coding them. For instance, Fernandez et al. [26] provided an unsupervised algorithm named probabilistic latent semantic hashing, which adopted a pre-trained network of ResNet-18 to obtain features and then used probabilistic topic models to produce the hash code. It cannot simultaneously learn feature and hash code, resulting in unsatisfactory results. Due to the lack of predefined labels, it is challenging to develop an objective function that forces related/unrelated images to be close/far apart in the Hamming space, thus failing to build a deep hashing network that can be trained end-to-end.

Fortunately, self-supervised contrastive learning, which achieves advanced success in feature representation, has recently provided creative thinking for solving the above problem. It can generate powerful image representations from vast amounts of unlabeled data by creating a deep unsupervised network. In addition, it eliminates the need for a predefined label by maximizing the agreement between different views of the same image. However, contrastive learning cannot be directly applied to hashing since it focuses on learning continuous features rather than binary codes. The continuous features are high-dimensional and real-valued, while our aim of binary codes must be short and binary (each element is either -1 or $+1$).

In this paper, we propose a deep unsupervised hash learning method for remote sensing image retrieval by modifying contrastive learning, namely deep contrastive self-supervised hashing (DCSH). Using only unlabeled images, it can build a deep network with end-to-end training. Motivated by contrastive learning, we assume that the two hash codes for different views generated from the same image are similar, while the two hash codes for views generated from different images are not similar. Based on the hypothesis, we design a new loss function to preserve the generated hash codes with semantic similarity and train the unsupervised network end-to-end. In the proposed framework, four parts are included in the training process, as shown in Figure 1. First is the data augmentation part; each image is randomly transformed into two different views. Secondly is the encoder part; the augmented images are fed into an encoder network with shared parameters to obtain deep discriminative features. Thirdly, a hash layer is used to convert the high-dimensional image representations into low-dimensional compact binary codes. Lastly, a novel contrastive loss function is introduced to train the proposed network end-to-end and thus guide the generated hash codes with semantic similarity. In summary, there are three main contributions in this paper:

1. We present a novel deep unsupervised hashing method for remote sensing image retrieval. Motivated by contrast learning, we hypothesize that the hash codes for different views generated from the same image are similar, while the hash codes for views generated from different images are not similar. To the best of our knowledge, we are the first to implement this idea for remote sensing image retrieval. According to the hypothesis, we can build a deep unsupervised hash network with end-to-end training, which can learn discriminative hash codes from a large number of unlabeled data. It avoids the problem of annotating images compared to most deep hashing algorithms.
2. We introduce a novel hashing objective loss to train our deep network. This gives each image a more effective hash code, improving the efficiency of image searching. Additionally, instead of the conventional relaxation strategy, we propose a continuous strategy that converges a non-differentiable sign function using a sequence of differentiable functions, allowing us to explicitly enforce binary constraints on hash codes.
3. In contrast to existing unsupervised hashing methods for the retrieval of remote sensing images, we achieve the state-of-the-art results on benchmark datasets of the UC Merced Land Use Database and the Aerial Image Dataset.

The article is arranged in the following manner. In Section 2, the related work is reviewed. Section 3 gives a detailed description of our proposed unsupervised deep hashing method. Section 4 presents extensive experimental results on two popular benchmark datasets of the UC Merced Land Use Database and the Aerial Image Dataset. Section 5 discusses several hyper-parameters and shows the retrieval results of multiple methods in a visual manner. Section 6 gives a conclusion of this research.

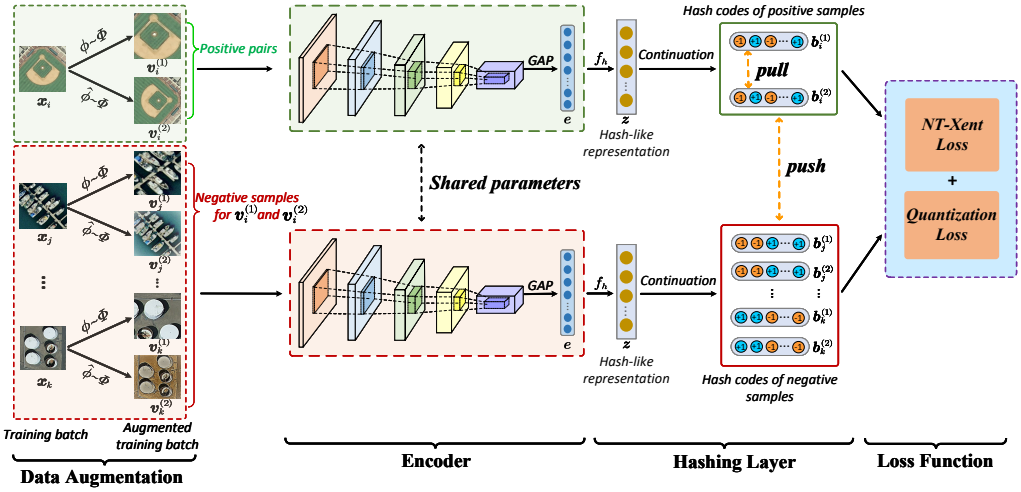


Figure 1. The framework of deep contrastive self-supervised hashing (DCSH). It consists of four components: data augmentation, encoder, hashing layer and objective loss function. **Firstly,** each image $\boldsymbol{x}_i$ in the sampled mini-batch training images $\{\boldsymbol{x}_i\}_{i=1}^{M}$ would be transformed into two different views $\boldsymbol{v}_i^{(1)}$ and $\boldsymbol{v}_i^{(2)}$ by the related data augmentation strategy. After augmentation, we set two views from the same image as the positive pairs, which means, for $\boldsymbol{v}_i^{(1)}$, the only positive sample is $\boldsymbol{v}_i^{(2)}$, and the others are all considered negative samples. $\boldsymbol{v}_i^{(2)}$ is the same. **Secondly**, we utilize the encoder module to extract the discriminative deep feature representation $\boldsymbol{e}$ for each image in the augmented training batch. **Thirdly**, the hashing layer would transform the feature vectors $\boldsymbol{e}_i^{(1)}$, $\boldsymbol{e}_i^{(2)}$ and $\boldsymbol{e}_j^{(1)}$, $\boldsymbol{e}_j^{(2)} (j \neq i)$ to corresponding hash codes $\boldsymbol{b}_i^{(1)}$, $\boldsymbol{b}_i^{(2)}$ and $\boldsymbol{b}_j^{(1)}$, $\boldsymbol{b}_j^{(2)}$. **Finally**, in Hamming space, we employ the normalized temperature-scaled cross-entropy loss (NT-Xent loss) and quantization loss to pull the distances of the positive sample pairs and to push the distances between the negative samples.

2. Related Work

Unsupervised hashing aims at encoding images into data-aware binary codes with only unlabeled images, and it is not well-developed in the remote sensing community. The existing methods are limited and mainly contain two modules. The first step is the **feature learning module**. Each remote sensing image is described by a visual representation. The second step is the **hash learning module**. The visual feature is fed into an unsupervised hashing algorithm to produce a hash code. For instance, in [36], Demir et al. proposed a kernel-based unsupervised locality-sensitive hashing method, which characterized each image by a bag-of-visual-words feature and then encoded it into hash code by a set of hash functions learned in kernel space. Later, Li et al. [37] presented a partial randomness hashing approach, which first adopted the hand-crafted holistic descriptor [38] with 512-dimension as visual representation, and then constructed hash functions based on the framework approximated auto-encoder structure [39]. Unlike a traditional auto-encoder, only parameters on the decoder require optimization. Its parameter strategy, partial randomness and partial training makes hash function construction efficient and effective. In [25], Reato et al. represented each image by multi-hash codes instead of single-hash codes. More specially, it described images by primitive-sensitive clusters and then encoded them into multi-hash codes by kernel-based unsupervised locality-sensitive hashing, at the cost of more hash bits to represent each image.

However, the above-mentioned methods have not achieved satisfactory retrieval performance. First, the feature learning module produces low-level hand-crafted features primarily based on textures without sufficient semantic information. Secondly, the modules of feature learning and hash learning are separated without interaction with each other. Thus, hash codes of the above approaches are incompetent when describing the complex semantic content of the remote sensing image. In order to embed more semantics into hash codes, Fernandez et al. [26] provided an unsupervised algorithm named probabilistic latent

semantic hashing, which adopted pre-trained ResNet-18 as a feature learning module and added an extra probabilistic topic model. As proven in various remote sensing imagery tasks, the intermediate outputs of pre-trained deep convolutional neural networks often have stronger capability than hand-crafted descriptors on the characterization of high-level semantic information of RS images. In addition, probabilistic topic models [40] can further extract higher-level semantic content via uncovering hidden semantic patterns from visual features in feature learning modules. However, Ref. [26] may still suffer from two limitations. On the one hand, when the pre-trained convolutional neural network poorly generalizes in the target dataset, the pre-extracted deep visual feature may lead to inferior performance. On the other hand, the feature learning module is still independent of the hash learning module. In other words, when optimizing the loss function of the hash learning module, the feature vectors are fixed and unable to be optimized simultaneously. Thus, visual features from semantically dissimilar/similar image pairs might not have a relatively satisfactory distance to well preserve the similarity. To solve the above two problems, in our paper, we aim to create a deep unsupervised network that can simultaneously optimize feature and hash code learning with only unlabeled images.

3. Proposed Method

Given a training set of N images $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_N\}$ without any labels, for the similarity retrieval task, we expect to learn an appropriate hash function that maps each image $\boldsymbol{x}_i$ into a compact K-bit binary codes $\boldsymbol{b}_i \in \{-1, 1\}^K$, where N denotes the number of images from the training set and K represents the length of the binary codes. Additionally, the hash function should be such that semantically similar images are projected to nearby hash codes while dissimilar images are encoded far away in the Hamming space. Then, image retrieval, searching images semantically similar to a query image, can be efficiently fulfilled by Hamming distance computed with bit-wise XOR operations.

In our paper, we propose a novel deep end-to-end framework to conduct unsupervised hash learning, as shown in Figure 1. **Firstly,** in Section 3.1, each image in the sampled mini-batch training images would be transformed into two different views by different data augmentation strategies. After augmentation, we set two views from the same image as the positive pairs and two views from different images as negative pairs. **Secondly**, we utilize the encoder module (Section 3.2) to extract the discriminative deep feature representation and then the hashing layer (Section 3.3) to transform the feature vectors to corresponding hash codes. **Finally**, the novel loss function is proposed in Section 3.4 to train our network. In Section 3.5, we give the optimizing strategy for our proposed network.

3.1. Data Augmentation

Generally, to equip learned hash code with similarity-preserving properties, the supervised hash learning [31,41,42] needs to leverage the label information to construct similar relations for the training data as supervised information. Specifically, two points are considered similar if they have at least one label in common; otherwise, they are dissimilar. However, due to the absence of label annotations, unsupervised hashing methods [37,43–45] can not model the similarity relations of data points by this means.

To address this problem, inspired by SimCLR [46], we introduce the data augmentation strategy to generate similar/dissimilar pairs for the training dataset. Specifically, we consider that two augmented views generated from the same image are similar, and inversely, two views produced from different images are dissimilar. We use common visual transformations to augment images. More specifically, we sequentially apply the following data augmentation techniques; that is, random cropping and resizing, random horizontal flipping, random color jittering, random grayscale, and random Gaussian blur, as shown in Figure 2.

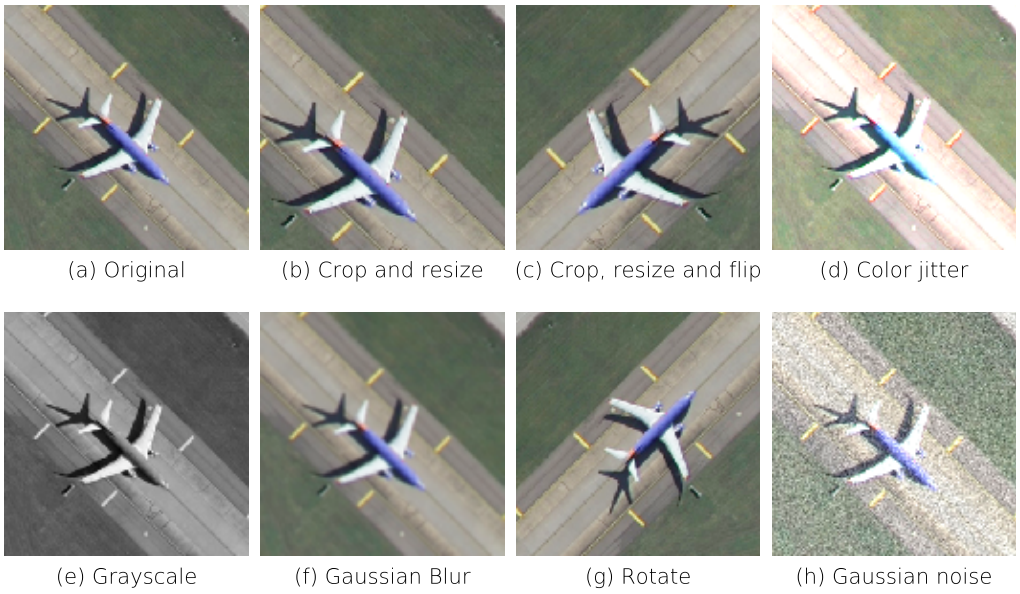


Figure 2. We show data augmentation operators through a series of illustrations. Note that some augmentation technologies have a few internal parameters that need to be chosen randomly, such as Gaussian blur and noise.

Given the original image $\boldsymbol{x}_i$, we get two augmented views $\boldsymbol{v}_i^{(1)}$ and $\boldsymbol{v}_i^{(2)}$ by randomly selecting two different augmented transformations ϕ and $\hat{\phi}$ presented in Figure 2, which can be modeled by the following expression:

$$\begin{aligned} \boldsymbol{v}_i^{(1)} &= \phi(\boldsymbol{x}_i) \\ \boldsymbol{v}_i^{(2)} &= \hat{\phi}(\boldsymbol{x}_i) \end{aligned} \tag{1}$$

3.2. Encoder

Recently, deep convolutional neural networks (CNNs) have shown powerful feature learning capability. Therefore, we tailor one of them to our encoder, which aims to project each input augmented image into discriminative representation. There exist various well-known neural networks, such as AlexNet [47], VGG [48], GoogLeNet [49], and ResNet [50]. Among them, ResNet-based networks have achieved remarkable performance in diverse computer vision tasks, so we extend our encoder network from ResNet-50, which has a convolutional layer conv1, four residual blocks conv2_x–conv5_x, a global average pooling layer, and a fully connected layer (classifier layer). We remove all layers after the global average pooling layer, and the detailed structural information of the encoder is shown in Table 1. Given an augmented image $\boldsymbol{v}_i^{(c)}$, we can get a deep feature vector $\boldsymbol{e}_i^{(c)}$ with 2048-dimension, which can be formulated as:

$$\boldsymbol{e}_i^{(c)} = E_n\left(\boldsymbol{v}_i^{(c)}; \theta_e\right) \tag{2}$$

where θ_e denotes the encoder parameters, and E_n is our encoder network.

Table 1. The Architecture for the encoder.

Layer Name	Output Size (Width × Height × Channel)	Configuration
conv1	112 × 112 × 64	7 × 7, 64, stride 2
conv2_x	56 × 56 × 256	3 × 3 max pool, stride 2 $\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix} \times 3$
conv3_x	28 × 28 × 512	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix} \times 4$
conv4_x	14 × 14 × 1024	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix} \times 6$
conv5_x	7 × 7 × 2048	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix} \times 3$
global average pooling layer	2048	global average pooling

3.3. Hashing Layer

The hashing layer aims to compress high-dimensional continuous vectors $\boldsymbol{e}$ into low-dimensional compact binary hash codes $\boldsymbol{b}$, resulting in efficient retrieval and storage. We use two steps to attain this goal. The first step is feature reduction. In this paper, we adopt Multi-Layer Perception (MLP) with two fully connected layers to compress the 2048-dimensional feature vector to hash-like representation $\boldsymbol{z}$ with K-dimension, where K is the length of hash codes. The process is shown as follows:

$$\boldsymbol{z} = f_h(\boldsymbol{e}; W) = W^{(2)}\sigma\left(W^{(1)}\boldsymbol{e}\right) \tag{3}$$

where $W^{(1)}$ and $W^{(2)}$ denote parameters of the two fully connected layers, respectively, and $W = \left[W^{(1)}; W^{(2)}\right]$, as shown in Figure 3a, σ is a non-linear function of rectified linear units (ReLU); that is,

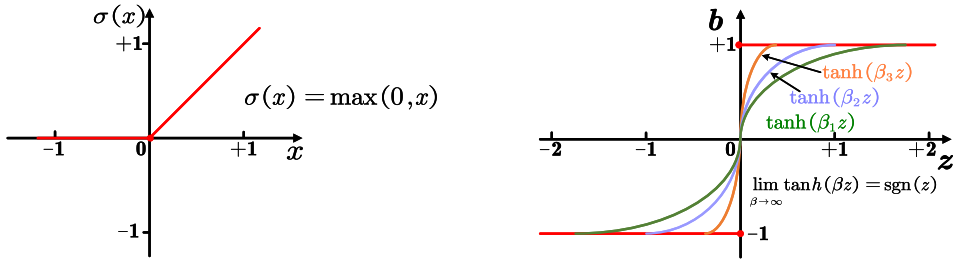
$$\sigma(x) = \max(0, x) \tag{4}$$

Note that $\boldsymbol{z}$ in Equation (3) is still continuous rather than binary. Thus the second step is to quantize the continuous vector into a binary code by an activation function, which is generally performed by the element-wise sign function. The function can be formulated as:

$$\boldsymbol{b} = sgn(z) = \begin{cases} +1, & z \geq 0 \\ -1, & z < 0 \end{cases} \tag{5}$$

3.4. Loss Function

The goal of our proposed framework is to produce binary hash codes of the input images. The generated hash codes should meet two constraints. The first one is preserving similarity. We satisfy this limit by the normalized temperature-scaled cross-entropy loss called NT-Xent loss due to their consistent aim of semantic preservation, i.e., pulling similar images together and pushing dissimilar ones apart in embedding space. The second one is quantization loss, which forces embedded features to be -1 or $+1$.



(**a**) (**b**)

Figure 3. Rectified linear units (ReLU) function and continuation optimization. (**a**) ReLU function. (**b**) Continuation optimization.

For a minibatch image $\{x_i\}_{i=1}^{M}$ randomly sampled from the training set, we can get the corresponding M pairs of binary codes $\left\{\left(\boldsymbol{b}_i^{(1)}, \boldsymbol{b}_i^{(2)}\right)\right\}_{i=1}^{M}$. We apply *NT-Xent* loss on all M pairs; that is,

$$L_{NT-Xent} = \frac{1}{M}\sum_{i=1}^{M} l(i) \tag{6}$$

where $l(i)$ denotes an expression of loss function on i-th pair, which aims at forcing related pairs $\left\{\boldsymbol{b}_i^{(1)}, \boldsymbol{b}_i^{(2)}\right\}$ from the same image $\boldsymbol{x}_i$ close and unrelated pairs $\left\{\boldsymbol{b}_i^{(c)}, \boldsymbol{b}_j^{(k)}\right\}_{j\neq i;c=1,2;k=1,2}$ from different images are far away. The definition is as follows:

$$\begin{aligned} l(i) &= -\frac{1}{2}\sum_{c=1,2}\log\frac{\exp\left(sim\left(\boldsymbol{b}_i^{(1)}, \boldsymbol{b}_i^{(2)}\right)/\tau\right)}{\exp\left(sim\left(\boldsymbol{b}_i^{(1)}, \boldsymbol{b}_i^{(2)}\right)/\tau\right) + \sum_{j\neq i}\sum_{k=1,2}\exp\left(sim\left(\boldsymbol{b}_i^{(c)}, \boldsymbol{b}_j^{(k)}\right)/\tau\right)} \\ &= \frac{1}{2}\sum_{c=1,2}\left[\log\left(1+\sum_{j\neq i}\sum_{k=1,2}\exp\left(\frac{sim(b_i^{(c)}, b_j^{(k)}) - sim(b_i^{(1)}, b_i^{(2)})}{\tau}\right)\right)\right] \end{aligned} \tag{7}$$

where τ is a temperature parameter, and $sim(\cdot,\cdot)$ is a metric function to measure the pairwise similarity of two binary codes. It must be such that the smaller Hamming distance between two binary codes, the larger the output of $sim(\cdot,\cdot)$. In order to meet this condition, we choose the cosine similarity:

$$sim(\boldsymbol{b}_i, \boldsymbol{b}_j) = \frac{\boldsymbol{b}_i^T \boldsymbol{b}_j}{\|\boldsymbol{b}_i\|\|\boldsymbol{b}_j\|} \in [-1, 1] \tag{8}$$

where $\|\cdot\|$ denotes the L2-norm, since there is a relationship between the Hamming distance $dist_H(\boldsymbol{b}_i, \boldsymbol{b}_j)$ and cosine similarity $sim(\boldsymbol{b}_i, \boldsymbol{b}_j)$: $dist_H(\boldsymbol{b}_i, \boldsymbol{b}_j) = \frac{K}{2}(1 - sim(\boldsymbol{b}_i, \boldsymbol{b}_j))$.

Equation (7) resembles softmax loss. It can shorten the similarity distance of positive pairs $\left(\boldsymbol{b}_i^{(1)}, \boldsymbol{b}_i^{(2)}\right)$ and enlarge the similarity distance of $2(M-1)$ negative pairs $\left\{\left(\boldsymbol{b}_i^{(c)}, \boldsymbol{b}_j^{(k)}\right)\right\}$ where $j \neq i; k = 1, 2; c = 1, 2$. Thus, compared to the triplet loss [51] with only one negative example, each sample in our loss function (Equation (7)) can interact with multiple negative examples, which eases the problem of slow convergence and poor local optima [52].

Based on Equation (7), when $l(i) \to 0$, we get $sim(b_i^{(c)}, b_j^{(k)}) - sim(b_i^{(1)}, b_i^{(2)}) \to -2$, where $\left(b_i^{(c)}, b_j^{(k)}\right)$ is a dissimilar pair, and $\left(b_i^{(1)}, b_i^{(2)}\right)$ is a similar pair. Thus the loss function used in this paper can make the hash codes of similar pairs close and the hash codes of dissimilar pairs far away.

3.5. Optimization by Continuation

Note that Equation (7) can not be optimized directly since the binary codes are produced by the sign function (Equation (5)) that suffers from the vanishing gradient problem. Therefore, our proposed deep hashing network fails to be trained by the back-propagation algorithm [53–55] in a truly end-to-end manner. However, numerous research studies [42,56–58] have shown that end-to-end training significantly improves the retrieval accuracy of deep hashing.

To train our framework end-to-end, we must address the binary constraint, and there are two kinds of methods. The first kind is an iterative alternating optimization algorithm [59,60], which alternately optimizes continuous vector $\boldsymbol{e}$ and binary hash codes $\boldsymbol{b}$ while fixing the other. That is to say, it skips the ill-posed gradient challenge caused by the sign function but substantially weakens the deep hashing model flexibility. The second kind is a widely-used relaxation scheme [61–65], which relaxes the discrete constraint $\{-1, +1\}$ to a continuous interval $[-1, +1]$ by seeking a smooth function (sigmoid or tanh) to approximate the sign function. Most studies [64–66] adopt a scaled tanh function $\tanh(\beta x)$, where β is a hyper-parameter controlling the trade-off between the function smoothness and the binary quantization error. Therefore, choosing optimal β is vital yet difficult [64–66] because a small value for β tends to a large quantization loss, while a large value for β tends to an almost vanishing gradient problem similar to the sign function. Hence, we adopt a novel way to solve the binary constraint.

According to the verification of recent works [67,68], the complex problem of non-smooth optimization can be simplified. Specifically, as shown in Figure 3b, we gradually increase the non-smoothness level β in $\tanh(\beta z)$, which would converge to the sign function optimization problem. This method is effective due to a nice relationship between the $sgn(.)$ and $\tanh(.)$ function:

$$sgn(z) = \lim_{\beta \to \infty} \tanh(\beta z) \tag{9}$$

Therefore, in the training process, to use standard back-propagation to train our deep hashing network in an end-to-end manner, we adopt a sequence of scaled tanh as our activation function instead of sign function, that is:

$$\boldsymbol{h} = \tanh(\beta z) \tag{10}$$

where $\boldsymbol{h}$ denotes a K-dimensional real-valued feature vector before converting to binary (termed as a hash-like feature) and β changes gradually.

$$sgn(z) = \lim_{\beta \to \infty} \tanh(\beta z) \tag{11}$$

The details of the training process are as follows. We start with $\beta = 1$ in Equation (10). For each stage, after the network converges in the last stage, we enlarge β and initialize the current stage parameters with the parameters of the last converged network. By involving $\tanh(\beta x)$ with $\beta \to \infty$, the network obtains the same results as adopting $sgn(x)$, which can produce exact binary hash codes as we promise. When β is increased to 10, we can already obtain expectant convergence. Therefore, for each stage, we use the hash-like feature h instead of binary hash code b, and Equation (7) can be rewritten as:

$$\hat{l}_{NT-Xent}(i) = -\frac{1}{2}\sum_{c=1,2}\log\frac{\exp\left(sim\left(\boldsymbol{h}_i^{(1)},\boldsymbol{h}_i^{(2)}\right)/\tau\right)}{\exp\left(sim\left(\boldsymbol{h}_i^{(1)},\boldsymbol{h}_i^{(2)}\right)/\tau\right)+\sum_{j\neq i}\sum_{k=1,2}\exp\left(sim\left(\boldsymbol{h}_i^{(c)},\boldsymbol{h}_j^{(k)}\right)/\tau\right)} \tag{12}$$

Quantization Loss: In order to accelerate the approximation process between the hash-like code $\boldsymbol{h}$ and the binary code $\boldsymbol{b}$, we add quantization loss:

$$l_Q(i) = \frac{1}{2}\left(\left\|\left|\boldsymbol{h}_i^{(1)}\right| - \mathbf{1}\right\|^2 + \left\|\left|\boldsymbol{h}_i^{(2)}\right| - \mathbf{1}\right\|^2\right) \tag{13}$$

where $\|\cdot\|$ and $|\cdot|$, respectively, denote the L2-norm and element-wise absolute value operation. The quantization loss on the i-th positive pair aims at pulling the element of hash-like features to desired discrete values $\{-1,+1\}$.

Total Loss: We combine NT-Xent loss (Equation (12)) and quantization loss (Equation (13)) to get the total loss function:

$$L_{total} = \frac{1}{M}\sum_{i=1}^{M}\left(\hat{l}_{NT-Xent}(i) + \alpha l_Q(i)\right) \tag{14}$$

where α is a weighting parameter to balance the two terms. Algorithm 1 is the main learning procedure of our proposed method.

Algorithm 1 DCSH's main learning algorithm.

Require: batch size M, initial and trainable encoder parameters θ_e, initial and trainable hash layer parameters $\boldsymbol{W}$, hyper-parameter of temperature τ and weighting parameter α

for sampled mini-batch $\{\boldsymbol{x}_i\}_{i=1}^{M}$ **do**
  for i in $\{1,2,\cdots,M\}$ **do**
    draw two different augmented transformation ϕ and $\hat{\phi}$, then get a pair of two related images from the same image x_i:
    $\boldsymbol{v}_i^{(1)} = \phi(\boldsymbol{x}_i)$
    $\boldsymbol{v}_i^{(2)} = \hat{\phi}(\boldsymbol{x}_i)$
    learn deep image representation by Encoder:
    $\boldsymbol{e}_i^{(1)} = E_n\left(\boldsymbol{v}_i^{(1)};\theta_e\right)$
    $\boldsymbol{e}_i^{(2)} = E_n\left(\boldsymbol{v}_i^{(2)};\theta_e\right)$
    get hash-like feature $\boldsymbol{h}_i^{(1)}$ and $\boldsymbol{h}_i^{(2)}$ via hashing layer of Equation (3) and activation function of *tanh* described in Equation (10):
    $\boldsymbol{z}_i^{(1)}, \boldsymbol{z}_i^{(2)} = f_h\left(\boldsymbol{e}_i^{(1)};W\right), f_h\left(\boldsymbol{e}_i^{(2)};W\right)$
    $\boldsymbol{h}_i^{(1)}, \boldsymbol{h}_i^{(2)} = \tanh\left(\beta \boldsymbol{z}_i^{(1)}\right), \tanh\left(\beta \boldsymbol{z}_i^{(2)}\right)$
  end for
  for i in $\{1,2,\cdots,M\}$ **do**
    compute NT-Xent loss on i-th related pair $\hat{l}_{NT-Xent}(i)$ via Equation (12) and quantization loss on i-th related pair $l_Q(i)$ via Equation (13)
  end for
  total M pairs loss: $L_{total} = \frac{1}{M}\sum_{i=1}^{M}\left(\hat{l}_{NT-Xent}(i) + \alpha l_Q(i)\right)$
end for

Inference Process: Note that, in the inference process, we get the desired discrete code by binarizing the hash-like feature h produced by Equation (10), i.e.,

$$\boldsymbol{b} = sgn(\boldsymbol{h}) \tag{15}$$

where sgn is a point-wise function defined by Equation (5).

4. Experiments

To verify the effectiveness of our proposed DCSH, we perform extensive retrieval experiments on two popular benchmark databases, the UC Merced Land Use Database (UCMD) and the Aerial Image Dataset (AID).

4.1. Datasets

The performance of our proposed method is evaluated on two remote sensing benchmark datasets. We briefly introduce these datasets as follows.

- **UCMD [69]:** It is publicly free and provided by the University of California, which collected surface images from the United States national city map produced by the United States Geological Survey. This dataset comprises 21 challenging land cover concepts, where each concept consists of 100 images of size 256 × 256 with a spatial resolution of 0.3 m per pixel. We show images from UCMD in Figure 4.
- **AID [70]:** It is a large-scale remote sensing publicly available dataset gathered by Wuhan University from Google Earth imagery. The images are completely annotated by specialists in the remote sensing image interpretation field. By contrast with the UC Merced dataset, AID is significantly more challenging. Specifically, the dataset comprises a total of 10,000 aerial images with a fixed size of 600 × 600 pixels and a spatial resolution varying from 8 m to about 0.5 m. It is categorized into 30 land-use scene classes such as airport, bare land, dense residential, desert, and so on; each class contains a number of images ranging from 220 to 420. Furthermore, the AID is derived from diverse remote sensing imaging sensors and chosen from different countries, under various times and distinct seasons, which makes the intra-class diversity larger and inter-class dissimilarity smaller as well. We show images from AID in Figure 5.

Figure 4. Sample images of the UC Merced Land Use Database (UCMD).

Figure 5. Sample images of the Aerial Image Dataset (AID).

4.2. Evaluation Protocols

To perform the quantitative evaluation of image retrieval, we adopt three standard evaluation metrics: precision-recall curves (PR-curves) [71], mean average precision (*MAP*), and precision within different numbers of top retrieved examples. For the query database, the *MAP* is computed as follows:

$$MAP = \frac{1}{|Q|}\sum_{i=1}^{|Q|} AP_i$$

where $|Q|$ denotes the total number of query databases, and AP_i is the average precision of query x_i based on ranking, i.e.,

$$AP_i = \frac{1}{N_+}\sum_{k=1}^{n}\frac{N_+^k}{k} \times pos(k)$$

where N_+ is the total number of images related to query x_i in the retrieval database, n denotes the number of images in the retrieval database, N_+^k is the number of related examples within the top k retrieved examples, and $pos(k)$ is an indicator function and indicates if the returned image at position k is related to the query. $pos(k) = 1$ if the relationship is relevant, otherwise $pos(k) = 0$.

4.3. Implementation Details

We implement our DCSH method using the open source Pytorch (https://pytorch.org/) on Ubuntu 16.04 with 2 NVIDIA GeForce GTX TITAN X GPUs. The

DCSH method contains four main components: data augmentation, encoder, hashing layer, and contrastive loss function. We conduct the experiment on two benchmark datasets, UCMD and AID. We randomly select a mini-batch of images from the training dataset; the selected images are first resized into $224 \times 224 \times 3$ as the input. The data augmentation module randomly transforms each image into two different views with the size of $224 \times 224 \times 3$. Then the augmented mini-batch training set will be fed into our encoder network. The architecture of the encoder is shown in Table 1; we construct the encoder network from the original ResNet-50 by removing all layers after the global average pooling layer. We initialize it with the pre-trained ResNet50, which has been well-trained on the ImageNet dataset [72]. The hash layer contains two fully connected (FC) layers; the first FC layer has 1024 neurons with ReLU activation function, and the second FC layer has K neurons (the length of hash code) and deploys scaled tanh function as the activation function. In addition, the hash layer is randomly initialized with zero-mean Gaussian distribution with a standard deviation of 0.01. We use Adam (Adaptive moment estimation) [73] as the optimizer in our experiment; the learning rates of the encoder network and the hash layer are set to 10^{-5} and 10^{-4}, respectively. The training epochs for UCMD and AID are set to 100 and 120, respectively. The hyper-parameters of batch size and the temperature τ are set to 64 and 0.3, respectively. The detailed analysis of these settings will be further investigated in Section 5. The weighting parameter α is set to 1.

4.4. Comparative Experiments with State-of-the-Art Methods

We compare our proposed unsupervised deep hashing framework for remote sensing image retrieval with six state-of-the-art methods, including three unsupervised hashing methods specifically designed for remote sensing, i.e., kernel-based unsupervised hashing (KULSH) [36], partial randomness hashing (PRH) [37] and probabilistic latent semantic hashing (PLSH) [26]; and three representative hashing methods for natural images widely used in computer vision, i.e., density sensitive hashing (DSH) [74], locality sensitive hashing (LSH) [75] and iterative quantization (ITQ) [59]. For a fair comparison, all approaches are unsupervised hashing without utilizing any label information in the training stage, which is vital in practical application since the label annotation is highly labor-intensive and prone to errors.

According to the settings in Shan's recent paper [14], we take both hand-crafted features of 512-dimensional GIST descriptors [38] and fully-connected features with 4096 dimensions of a pre-trained VGG16 as input for all these baseline methods and use '-CNN' and '-GIST' as the notation, respectively. Then, we demonstrate the experimental results of all the compared methods and our proposed method under the indicators above on UCMD and AID datasets. Moreover, we conduct several ablation analyses and discussions on two significant hyper-parameters appearing in our proposed method in Section 5.

4.4.1. Results on UCMD

For the UCMD dataset, there are 21 classes and each class consist of 100 images. We randomly choose 80 images for every class as training data (1680 images), and in the retrieval database, the remaining images (420 images in total) are set as the test queries.

Table 2 shows the *MAP* scores obtained by our approach and other baseline methods for hash codes with different lengths on the UCMD dataset. First, we can observe a significant superiority of our proposed method over the baseline methods in all hash code lengths. To be specific, the *MAP* score of our method shows a large improvement of 7.02–13.52% compared to the second-best competitor, PLSH-CNN. In addition, the methods specially designed for remote sensing images, such as KULSH, PRH and PLSH, outperform the methods for natural images; this phenomenon is also observed in a recent paper [14]. Third, all the 'CNN'-based hash methods outperform their 'GIST'-based versions, which indicates the CNN feature is more discriminative than the hand-crafted feature of GIST descriptors under the same hash method. Finally, even in general, the *MAP* score increases

when the length of the hash code increases; the *MAP* score will sometimes decline when the length of the hash code reaches a certain value, such as 64-bits, as shown in Table 2.

Table 2. The mean average precision (*MAP*) retrieval results on the UCMD dataset; we give separate hash retrieval results for 16, 32, 48, 64 bits. The testing set has 420 images (each category randomly takes 20% of the images).

Method	Hash Code Length			
	16-bits	**32-bits**	**48-bits**	**64-bits**
DSH-GIST [74]	16.38	17.09	19.24	19.18
LSH-GIST [75]	17.55	18.25	19.78	21.46
ITQ-GIST [59]	19.35	20.45	20.89	20.78
KULSH-GIST [36]	28.37	33.56	34.98	34.16
PRH-GIST [37]	31.77	33.38	35.76	36.92
PLSH-GIST [26]	40.49	44.17	47.32	46.37
DSH-CNN	28.35	33.90	34.24	34.66
LSH-CNN	32.44	38.58	45.48	51.67
ITQ-CNN	42.38	45.99	47.28	47.49
KULSH-CNN	53.68	55.37	58.23	64.58
PRH-CNN	55.39	59.45	61.89	67.77
PLSH-CNN	62.28	65.35	70.44	73.07
OUR	**75.80**	**78.17**	**80.49**	**80.09**

In addition to the *MAP* indicator, we demonstrate two important evaluation metrics to further validate the effectiveness of our proposed method, i.e., precision curves and precision-recall curves. Figure 6 shows the UCMD retrieval results of precision curves with respect to a different number of retrieved images for different methods. As shown in Figure 6a,b, our proposed method outperforms all baseline methods by significant margins on the precision curve with 32 and 64 bits, respectively. Compared to methods with 32 bits hash codes, the same methods with 64 bits hash codes have small gains in precision with the same number of returned images, which indicates that appropriately increasing the length of hash codes can contribute to improving the retrieval performance. Figure 7 reveals the precision-recall curves by ranking on Hamming distance. From Figure 7a,b, we can find that our method obtains the best performance on both 32 and 64 bits, respectively. Therefore, compared to the state-of-art algorithm of PLSH, we obtain higher precision with the same recall, which is desirable for the remote sensing image retrieval task.

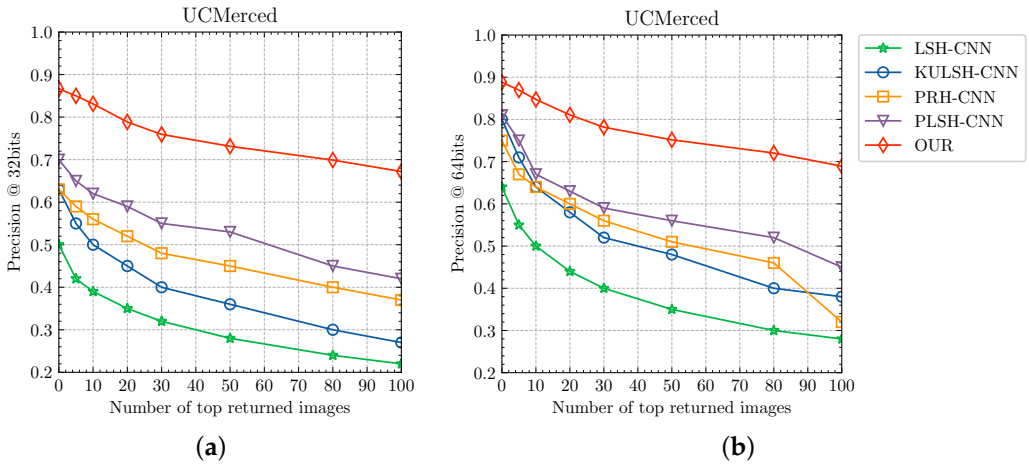


Figure 6. Precision curves with respect to a different number of retrieved images on the UCMD dataset. (**a**) Precision curve with regard to top-n@32 bits. (**b**) Precious curve with regard to top-n@64 bits.

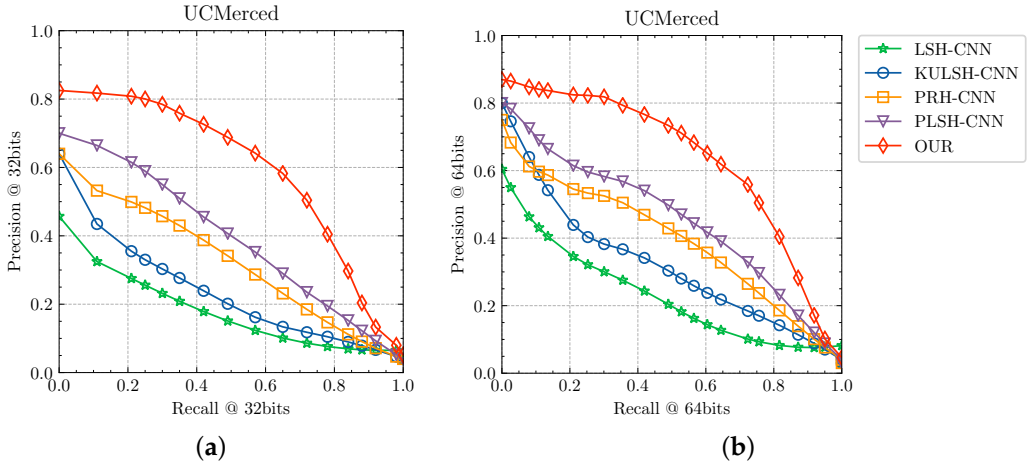


Figure 7. Precision-recall curves on the UCMD dataset. (**a**) Precision-recall curve@32 bits. (**b**) Precision-recall curve@48 bits.

4.4.2. Results on AID

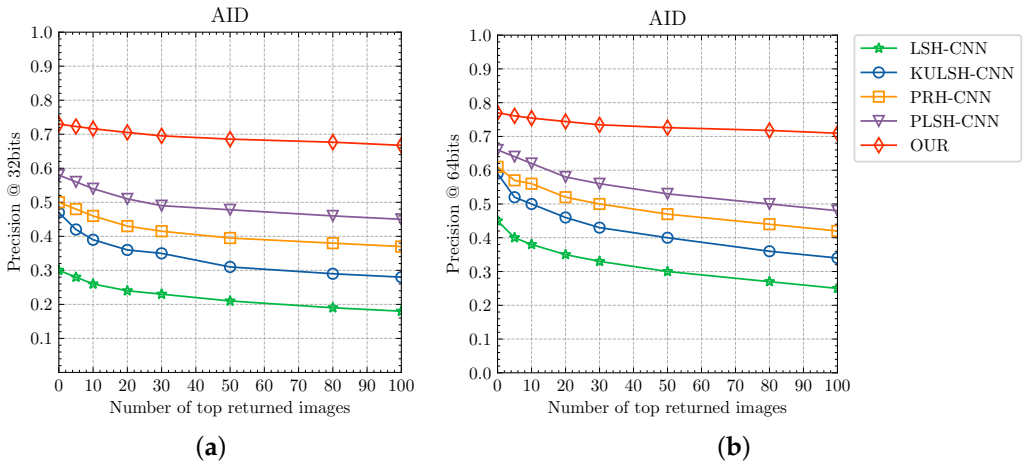
The AID dataset consists of 30 classes, and each class contains a number of images ranging from 220 to 420. We randomly choose 50% of the samples from every class as training data (5000 images) and retrieval database, and the remaining images (500 images in total) are set as the test queries. Each image has a fixed size of 600 × 600, and we resize it to 256 × 256.

Table 3 shows the *MAP* scores for hash codes with different lengths ranging from 16 to 64 bits on the AID dataset. First, as shown in Table 3, our method surpasses all baseline methods. Compared to the second-best competitor PLSH-CNN, *MAP* indicates a large advantage of 13.69–17.69%. Second, the methods specifically designed for remote sensing images are significantly superior to the methods for natural images such as DSH, LSH and ITQ. Third, all the 'CNN'-based hash methods outperform their 'GIST'-based versions, which indicates that the CNN feature is a more discriminant feature than hand-crafted features of GIST descriptors under the same hash method on the AID dataset. For example, KULSH-CNN gets more than 27% than KULSH-GIST at all hash code lengths. Finally, even in general, the *MAP* score increases when the length of the hash code increases, the *MAP* score will sometimes decline when the length of the hash code reaches a certain value, such as 64-bits, as shown in Table 3.

Table 3. *MAP* retrieval results on the AID dataset; we give separate hash retrieval results for 16, 32, 48, 64 bits. The testing set has 5000 images (each category randomly takes 50% of the images).

Method	Hash Code Length			
	16-bits	**32-bits**	**48-bits**	**64-bits**
DSH-GIST [74]	9.42	9.87	10.06	10.35
LSH-GIST [75]	10.53	10.89	12.74	13.77
ITQ-GIST [59]	9.67	10.49	11.74	11.91
KULSH-GIST [36]	11.43	13.46	14.63	15.86
PRH-GIST [37]	13.42	15.37	15.74	17.26
PLSH-GIST [26]	13.26	17.32	18.64	18.99
DSH-CNN	16.25	18.74	19.63	19.57
LSH-CNN	25.44	29.79	35.58	40.66
ITQ-CNN	23.28	27.29	28.74	29.65
KULSH-CNN	38.47	41.37	46.64	50.62
PRH-CNN	43.89	47.58	50.27	55.74
PLSH-CNN	48.72	52.35	55.83	60.14
OUR	**66.17**	**70.04**	**71.48**	**73.83**

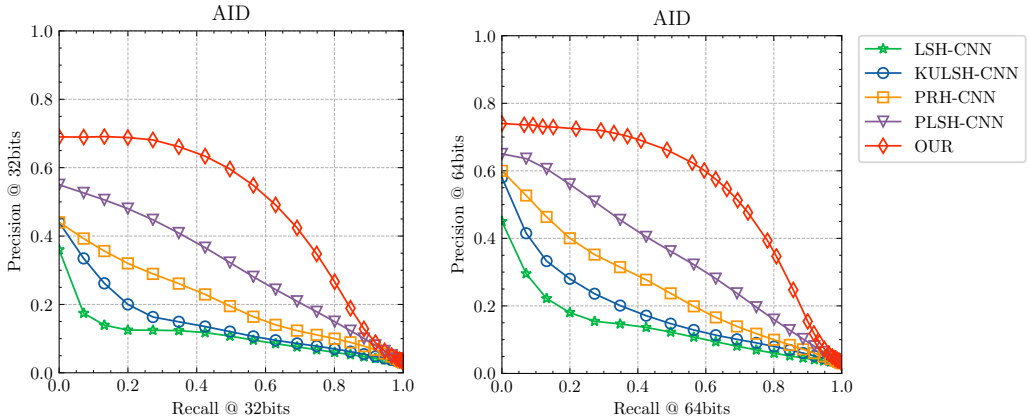
Figure 8 shows the AID retrieval results of precision curves with respect to the different numbers of retrieved images for different methods. As shown in Figure 8a,b, the proposed method outperforms all baseline methods by significant margins on precision curves with 32 and 64 bits, respectively. Compared to methods with 32-bit hash codes, the same methods with 64 bits hash codes have a small gain in the precision with the same number of returned images, which indicates that an appropriate increase in the length of hash codes would contribute to the promotion of the retrieval performance.



(**a**) (**b**)

Figure 8. Precision curves with respect to a different number of retrieved images on the AID dataset. (**a**) Precision curve with regard to the top-n@32 bits. (**b**) Precious curve with regard to the top-n@64 bits.

Figure 9a,b reveal the precision-recall curves by ranking the Hamming distance. Since high precision corresponding to high recall has been a sign of a desirable retrieval algorithms, our method is advantageous over other compared methods on both 32 bits (red line) and 64 bits (red line), respectively.



(**a**) (**b**)

Figure 9. Precision-recall curves on the AID dataset. (**a**) Precision-recall curve@32 bits. (**b**) Precision-recall curv@64 bits.

5. Discussion

5.1. The Analysis and Setting of Temperature τ

Similar to previous work [46,76], τ in Equation (7) is used to scale the sensitivity of the loss function, which plays an important role in our unsupervised contrastive learning on remote sensing image retrieval. In view of this, we further make an ablation analysis to examine the influence of hyperparameter τ on *MAP* scores on the UCMD dataset. We fix the

batch size to 64 and evaluate the *MAP* score by varying τ from 0.1 to 1. From Figure 10, it is inspiring that the model trained with the optimal temperature can improve performance by nearly 10%. More specifically, with the increase in τ, the *MAP* score first shows a sharp increase before $\tau = 0.3$ and then remains at the top level from $\tau = 0.3$ to $\tau = 0.6$, then is followed by a moderate drop afterward. The best two values of the *MAP* score (0.8049 and 0.8051) are almost equal. Since recent research shows that a smaller temperature can increase the model's penalty on difficult negative examples [77] and then generate more discriminative image features and hash codes, we set $\tau = 0.3$.

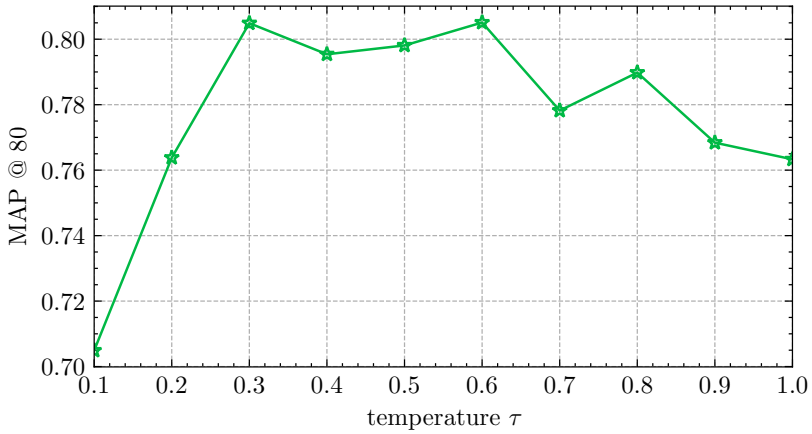


Figure 10. Model performance with various temperatures τ at 48 bits on UCMD.

5.2. The Analysis and Setting of Training Batch Size

As can be seen in the detail of training implements, our proposed method needs to set the proper batch size to train an optimal remote sensing retrieval model. In order to evaluate the effect of the batch size in the training process, we conduct several experiments to demonstrate the retrieval *MAP* metric under various batch sizes. Due to the conclusion that in the original feature space larger batch size can benefit the contrastive learning [46], we also design the batch size from small to large to study whether it still has this property in the Hamming space. Figure 11 shows the influence of batch size when the model is trained for different numbers of epochs, and this parameter setting makes a vast difference in getting excellent retrieval performance. We can observe that when the batch size is too small or too large, the performance is not optimal. On the other hand, as the batch size and epochs increase to a proper value, the performance rises obviously and then converges to the maximum *MAP*.

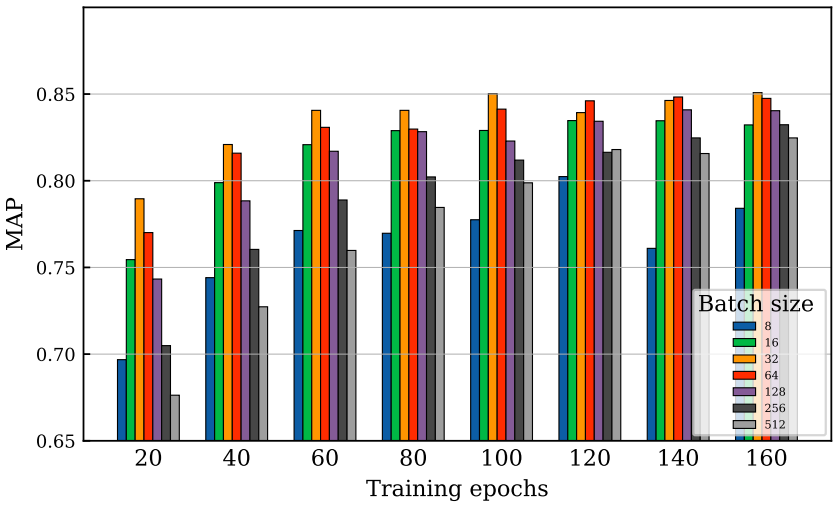


Figure 11. Our proposed model trained with different batch size and epochs.

5.3. Visualization Study

To further emphasize the effectiveness of our proposed method intuitively, we illustrate the quantitative top-10 matching results and the number of correctly returned images out of the top-60 results of the query samples, respectively, from the UCMD dataset and AID dataset in Figures 12 and 13. The query images are all randomly sampled in the related dataset, and the returned retrieval results are ranked by the Hamming distances between the query image and the images in the retrieval database. We select three competitive baseline methods, KULSH-CNN, PRH-CNN and PLSH-CNN, to compare with our proposed method in terms of the visual, quantitative retrieval results.

As shown in Figures 12 and 13, the first column displays the query image, and the remaining columns present the retrieval results for the different baseline methods. The red boxes indicate the false positive data points. Particularly, in view of the *MAP* results obtained by different hash bits presented in Tables 2 and 3, we can observe that the 48-bit hash codes can achieve the best result of 80.49% in the UCMD dataset and the 64-bit hash codes perform best in AID dataset. Therefore, we select 48-bit and 64-bit hash codes to demonstrate the performance of the two datasets, respectively. From the two figures, it can be seen that our proposed method can obtain all true positive results for two query samples (baseballdiamond from UCMD dataset and bridge from AID dataset) in the top-10 and top-60. The other three methods also obtain good results, but they have several wrong retrieval images. In addition, it is worth noting that our method can obtain more discriminative hash codes, which can distinguish confusing remote sensing images. For instance, the KULSH-CNN and PRH-CNN have an error in identifying the "tenniscourt" as the "baseballdiamond", and the PLSH-CNN misclassifies the residential as the bridge. Our method demonstrates obviously superior performance over other competitive baseline methods.

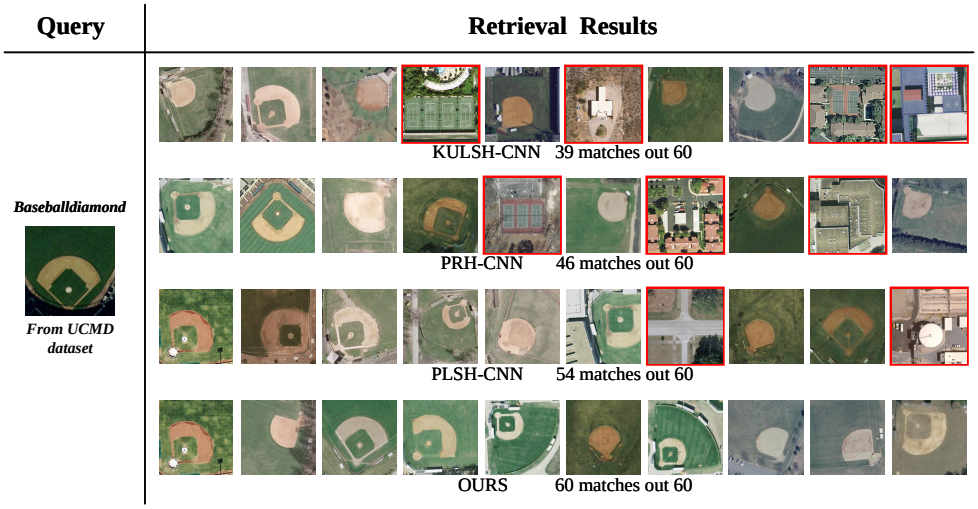


Figure 12. Retrieval results of "baseballdiamond" in the UCMD dataset by different competitive baseline methods. The single image in the left column is the query sample. Each row in the rest of the columns is the top-10 retrieval results from each comparable method. The results with red rectangles are incorrect, and the number of true positive results out of the top-60 retrieved images is also provided.

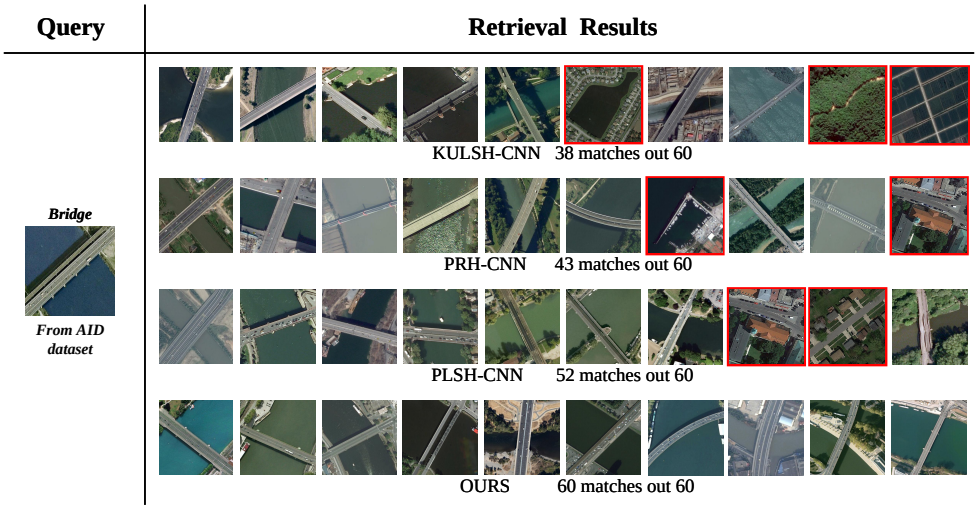


Figure 13. Retrieval results of "bridge" in the AID dataset by different competitive baseline methods. The single image in the left column is the query sample. Each row in the rest of the columns is the top-10 retrieval results from each comparable method. The results with red rectangles are incorrect, and the number of true positive results out of the top-60 retrieved images is also provided.

6. Conclusions

In this paper, we propose an unsupervised deep hash learning method for remote sensing image retrieval, namely deep contrastive self-supervised hashing. By creating a pretext task of preserving data visual augmentation invariance, we solve the problem of lacking supervised information, which is a significant challenge in unsupervised retrieval research. More specially, based on the task of pulling two augmented views generated from the same image and simultaneously pushing two views produced from different images, we develop the first deep unsupervised hash learning framework by introducing the normalized temperature-scaled cross-entropy loss and quantization loss. Extensive experiments on two popular benchmark datasets of UCMD and AID have demonstrated that our DCSH has overwhelming superiority in remote sensing image retrieval compared with state-of-art unsupervised hashing methods. For instance, on AID, the *MAP* performance for 16, 32, 48 and 64 bits of our DCSH method is 66.17%, 70.04%, 71.48% and 73.83%, respectively, which enables it to outperform KUSH, PRH and PLSH by at least 13.69%.

The paper only focuses on unimodal hash retrieval; that is, a query image must be provided to find the desired image. Nevertheless, retrieval system users sometimes prefer to enter text rather than images, so we plan to extend our method to the problem of retrieving images via text in the future, i.e., cross-modal hash retrieval.

Author Contributions: Conceptualization, X.T. and Y.Z.; methodology, X.T. and Y.Z.; data curation, X.T.; formal analysis, X.T.; investigation, X.T. and Y.Z.; software, X.T.; validation, X.T.; visualization, X.T. and Y.Z.; writing—original draft, X.T. and Y.Z.; writing—review and editing, X.T., Z.G. and Y.Z.; funding acquisition, K.Z.; supervision, Q.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the Innovation Group Project of National Natural Science Foundation of China No. 61821003.

Data Availability Statement: Not applicable.

Acknowledgments: We are grateful to the editor and the anonymous reviewers for their detailed review, valuable comments and constructive suggestions.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Lu, H.; Su, H.; Hu, J.; Du, Q. Dynamic Ensemble Learning with Multi-View Kernel Collaborative Subspace Clustering for Hyperspectral Image Classification. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2022**, *15*, 2681–2695. [CrossRef]
2. Li, Y.; Ma, J.; Zhang, Y. Image retrieval from remote sensing big data: A survey. *Inf. Fusion* **2021**, *67*, 94–115. [CrossRef]
3. Tong, X.Y.; Xia, G.S.; Hu, F.; Zhong, Y.; Datcu, M.; Zhang, L. Exploiting deep features for remote sensing image retrieval: A systematic investigation. *IEEE Trans. Big Data* **2019**, *6*, 507–521. [CrossRef]
4. Wolfmuller, M.; Dietrich, D.; Sireteanu, E.; Kiemle, S.; Mikusch, E.; Bottcher, M. Data flow and workflow organization—The data management for the TerraSAR-X payload ground segment. *IEEE Trans. Geosci. Remote Sens.* **2008**, *47*, 44–50. [CrossRef]
5. Wang, X.; Chen, N.; Chen, Z.; Yang, X.; Li, J. Earth observation metadata ontology model for spatiotemporal-spectral semantic-enhanced satellite observation discovery: A case study of soil moisture monitoring. *GISci. Remote Sens.* **2016**, *53*, 22–44. [CrossRef]
6. Peijun, D.; Yunhao, C.; Hong, T.; Tao, F. Study on content-based remote sensing image retrieval. In Proceedings of the IGARSS'05. 2005 IEEE International Geoscience and Remote Sensing Symposium, Seoul, Korea, 25–29 July 2005; Volume 2, p. 4.
7. Datta, R.; Li, J.; Wang, J.Z. Content-based image retrieval: Approaches and trends of the new age. In Proceedings of the 7th ACM SIGMM International Workshop on Multimedia Information Retrieval, Singapore, 10–11 November 2005; pp. 253–262.
8. Bretschneider, T.; Cavet, R.; Kao, O. Retrieval of remotely sensed imagery using spectral information content. In Proceedings of the IEEE International Geoscience and Remote Sensing Symposium, Toronto, ON, Canada, 24–28 June 2002; Volume 4, pp. 2253–2255.
9. Shao, Z.; Zhou, W.; Zhang, L.; Hou, J. Improved color texture descriptors for remote sensing image retrieval. *J. Appl. Remote Sens.* **2014**, *8*, 083584. [CrossRef]
10. Byju, A.P.; Demir, B.; Bruzzone, L. A progressive content-based image retrieval in JPEG 2000 compressed remote sensing archives. *IEEE Trans. Geosci. Remote Sens.* **2020**, *58*, 5739–5751. [CrossRef]
11. Dell'Acqua, F.; Gamba, P. Query-by-shape in meteorological image archives using the point diffusion technique. *IEEE Trans. Geosci. Remote Sens.* **2001**, *39*, 1834–1843. [CrossRef]
12. Jégou, H.; Douze, M.; Schmid, C.; Pérez, P. Aggregating local descriptors into a compact image representation. In Proceedings of the 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, San Francisco, CA, USA, 13–18 June 2010; pp. 3304–3311.
13. Yang, Y.; Newsam, S. Geographic image retrieval using local invariant features. *IEEE Trans. Geosci. Remote Sens.* **2012**, *51*, 818–832. [CrossRef]
14. Shan, X.; Liu, P.; Wang, Y.; Zhou, Q.; Wang, Z. Deep Hashing Using Proxy Loss on Remote Sensing Image Retrieval. *Remote Sens.* **2021**, *13*, 2924. [CrossRef]
15. Kang, J.; Fernandez-Beltran, R.; Ye, Z.; Tong, X.; Plaza, A. Deep hashing based on class-discriminated neighborhood embedding. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2020**, *13*, 5998–6007. [CrossRef]
16. Liu, C.; Ma, J.; Tang, X.; Zhang, X.; Jiao, L. Adversarial hash-code learning for remote sensing image retrieval. In Proceedings of the IGARSS 2019—2019 IEEE International Geoscience and Remote Sensing Symposium, Yokohama, Japan, 28 July–2 August 2019; pp. 4324–4327.
17. Demir, B.; Bruzzo, L. Kernel-based hashing for content-based image retrval in large remote sensing data archive. In Proceedings of the 2014 IEEE Geoscience and Remote Sensing Symposium, Quebec City, QC, Canada, 13–18 July 2014; pp. 3542–3545.
18. Kong, J.; Sun, Q.; Mukherjee, M.; Lloret, J. Low-Rank Hypergraph Hashing for Large-Scale Remote Sensing Image Retrieval. *Remote Sens.* **2020**, *12*, 1164. [CrossRef]
19. Liu, C.; Ma, J.; Tang, X.; Liu, F.; Zhang, X.; Jiao, L. Deep hash learning for remote sensing image retrieval. *IEEE Trans. Geosci. Remote Sens.* **2020**, *59*, 3420–3443. [CrossRef]
20. Li, P.; Zhang, X.; Zhu, X.; Ren, P. Online hashing for scalable remote sensing image retrieval. *Remote Sens.* **2018**, *10*, 709. [CrossRef]
21. Li, P.; Han, L.; Tao, X.; Zhang, X.; Grecos, C.; Plaza, A.; Ren, P. Hashing nets for hashing: A quantized deep learning to hash framework for remote sensing image retrieval. *IEEE Trans. Geosci. Remote Sens.* **2020**, *58*, 7331–7345. [CrossRef]
22. Song, W.; Li, S.; Benediktsson, J.A. Deep hashing learning for visual and semantic retrieval of remote sensing images. *IEEE Trans. Geosci. Remote Sens.* **2020**, *59*, 9661–9672. [CrossRef]
23. Tang, X.; Zhang, X.; Liu, F.; Jiao, L. Unsupervised deep feature learning for remote sensing image retrieval. *Remote Sens.* **2018**, *10*, 1243. [CrossRef]
24. Jin, S.; Yao, H.; Sun, X.; Zhou, S. Unsupervised semantic deep hashing. *Neurocomputing* **2019**, *351*, 19–25. [CrossRef]
25. Reato, T.; Demir, B.; Bruzzone, L. An unsupervised multicode hashing method for accurate and scalable remote sensing image retrieval. *IEEE Geosci. Remote Sens. Lett.* **2018**, *16*, 276–280. [CrossRef]
26. Fernandez-Beltran, R.; Demir, B.; Pla, F.; Plaza, A. Unsupervised remote sensing image retrieval using probabilistic latent semantic hashing. *IEEE Geosci. Remote Sens. Lett.* **2020**, *18*, 256–260. [CrossRef]
27. Huang, Y.; Peng, J.; Ning, Y.; Sun, W.; Du, Q. Graph embedding and distribution alignment for domain adaptation in hyperspectral image classification. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2021**, *14*, 7654–7666. [CrossRef]
28. Yang, B.; Li, H.; Guo, Z. Learning a deep similarity network for hyperspectral image classification. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2020**, *14*, 1482–1496. [CrossRef]

29. Deng, W.; Shi, Q.; Li, J. Attention-Gate-Based Encoder–Decoder Network for Automatical Building Extraction. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2021**, *14*, 2611–2620. [CrossRef]
30. Li, Y.; Zhang, Y.; Huang, X.; Zhu, H.; Ma, J. Large-scale remote sensing image retrieval by deep hashing neural networks. *IEEE Trans. Geosci. Remote Sens.* **2017**, *56*, 950–965. [CrossRef]
31. Han, L.; Li, P.; Bai, X.; Grecos, C.; Zhang, X.; Ren, P. Cohesion intensive deep hashing for remote sensing image retrieval. *Remote Sens.* **2020**, *12*, 101. [CrossRef]
32. Roy, S.; Sangineto, E.; Demir, B.; Sebe, N. Metric-learning-based deep hashing network for content-based retrieval of remote sensing images. *IEEE Geosci. Remote Sens. Lett.* **2020**, *18*, 226–230. [CrossRef]
33. Cao, R.; Zhang, Q.; Zhu, J.; Li, Q.; Li, Q.; Liu, B.; Qiu, G. Enhancing remote sensing image retrieval using a triplet deep metric learning network. *Int. J. Remote Sens.* **2020**, *41*, 740–751. [CrossRef]
34. Sumbul, G.; Ravanbakhsh, M.; Demir, B. Informative and Representative Triplet Selection for Multilabel Remote Sensing Image Retrieval. *IEEE Trans. Geosci. Remote Sens.* **2022**, *60*, 1–11. [CrossRef]
35. Cheng, Q.; Huang, H.; Ye, L.; Fu, P.; Gan, D.; Zhou, Y. A Semantic-Preserving Deep Hashing Model for Multi-Label Remote Sensing Image Retrieval. *Remote Sens.* **2021**, *13*, 4965. [CrossRef]
36. Demir, B.; Bruzzone, L. Hashing-based scalable remote sensing image search and retrieval in large archives. *IEEE Trans. Geosci. Remote Sens.* **2015**, *54*, 892–904. [CrossRef]
37. Li, P.; Ren, P. Partial randomness hashing for large-scale remote sensing image retrieval. *IEEE Geosci. Remote Sens. Lett.* **2017**, *14*, 464–468. [CrossRef]
38. Oliva, A.; Torralba, A. Modeling the shape of the scene: A holistic representation of the spatial envelope. *Int. J. Comput. Vis.* **2001**, *42*, 145–175. [CrossRef]
39. Wang, Y.; Yao, H.; Zhao, S. Auto-encoder based dimensionality reduction. *Neurocomputing* **2016**, *184*, 232–242. [CrossRef]
40. Blei, D.M. Probabilistic topic models. *Commun. ACM* **2012**, *55*, 77–84. [CrossRef]
41. Li, W.J.; Wang, S.; Kang, W.C. Feature learning based deep supervised hashing with pairwise labels. *arXiv* **2015**, arXiv:1511.03855.
42. Xia, R.; Pan, Y.; Lai, H.; Liu, C.; Yan, S. Supervised hashing for image retrieval via image representation learning. In Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, Quebec City, QC, Canada, 27–31 July 2014.
43. Chen, Y.; Zhao, D.; Lu, X.; Xiong, S.; Wang, H. Unsupervised Balanced Hash Codes Learning With Multichannel Feature Fusion. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2022**, *15*, 2816–2825. [CrossRef]
44. Ye, D.; Li, Y.; Tao, C.; Xie, X.; Wang, X. Multiple feature hashing learning for large-scale remote sensing image retrieval. *ISPRS Int. J. Geo-Inf.* **2017**, *6*, 364. [CrossRef]
45. Reato, T.; Demir, B.; Bruzzone, L. Primitive cluster sensitive hashing for scalable content-based image retrieval in remote sensing archives. In Proceedings of the 2017 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), Fort Worth, TX, USA, 23–28 July 2017; pp. 2199–2202.
46. Chen, T.; Kornblith, S.; Norouzi, M.; Hinton, G. A simple framework for contrastive learning of visual representations. In Proceedings of the International Conference on Machine Learning, Virtual Event, 13–18 July 2020; pp. 1597–1607.
47. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. *Adv. Neural Inf. Process. Syst.* **2012**, *25*, 1097–1105. [CrossRef]
48. Conneau, A.; Schwenk, H.; Barrault, L.; Lecun, Y. Very deep convolutional networks for text classification. *arXiv* **2016**, arXiv:1606.01781.
49. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1–9.
50. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
51. Yang, Y.; Geng, L.; Lai, H.; Pan, Y.; Yin, J. Feature pyramid hashing. In Proceedings of the 2019 on International Conference on Multimedia Retrieval, Ottawa, ON, Canada, 10–13 June 2019; pp. 114–122.
52. Sohn, K. Improved deep metric learning with multi-class n-pair loss objective. In Proceedings of the Advances in Neural Information Processing Systems, Barcelona, Spain, 5–10 December 2016; pp. 1857–1865.
53. Rumelhart, D.E.; Hinton, G.E.; Williams, R.J. Learning representations by back-propagating errors. *Nature* **1986**, *323*, 533–536. [CrossRef]
54. Hinton, G.E.; Osindero, S.; Teh, Y.W. A fast learning algorithm for deep belief nets. *Neural Comput.* **2006**, *18*, 1527–1554. [CrossRef]
55. Lillicrap, T.P.; Santoro, A.; Marris, L.; Akerman, C.J.; Hinton, G. Backpropagation and the brain. *Nat. Rev. Neurosci.* **2020**, *21*, 335–346. [CrossRef]
56. Lai, H.; Pan, Y.; Liu, Y.; Yan, S. Simultaneous feature learning and hash coding with deep neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 3270–3278.
57. Cao, Y.; Long, M.; Liu, B.; Wang, J. Deep cauchy hashing for hamming space retrieval. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 1229–1237.
58. Shen, Y.; Qin, J.; Chen, J.; Yu, M.; Liu, L.; Zhu, F.; Shen, F.; Shao, L. Auto-encoding twin-bottleneck hashing. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 2818–2827.

59. Gong, Y.; Lazebnik, S.; Gordo, A.; Perronnin, F. Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval. *IEEE Trans. Pattern Anal. Mach. Intell.* **2012**, *35*, 2916–2929. [CrossRef] [PubMed]
60. Su, S.; Zhang, C.; Han, K.; Tian, Y. Greedy hash: Towards fast optimization for accurate hash coding in cnn. In Proceedings of the 32nd International Conference on Neural Information Processing Systems, Virtual, 6–14 December 2018; pp. 806–815.
61. Lin, K.; Lu, J.; Chen, C.S.; Zhou, J. Learning compact binary descriptors with unsupervised deep neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 1183–1192.
62. Zhang, P.; Zhang, W.; Li, W.J.; Guo, M. Supervised hashing with latent factor models. In Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval, Gold Coast, Australia, 6–11 July 2014; pp. 173–182.
63. Chen, Z.; Yuan, X.; Lu, J.; Tian, Q.; Zhou, J. Deep hashing via discrepancy minimization. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 6838–6847.
64. Do, T.T.; Le, K.; Hoang, T.; Le, H.; Nguyen, T.V.; Cheung, N.M. Simultaneous feature aggregating and hashing for compact binary code learning. *IEEE Trans. Image Process.* **2019**, *28*, 4954–4969. [CrossRef] [PubMed]
65. Yuan, X.; Ren, L.; Lu, J.; Zhou, J. Relaxation-free deep hashing via policy gradient. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 134–150.
66. Zhang, R.; Lin, L.; Zhang, R.; Zuo, W.; Zhang, L. Bit-scalable deep hashing with regularized similarity learning for image retrieval and person re-identification. *IEEE Trans. Image Process.* **2015**, *24*, 4766–4779. [CrossRef] [PubMed]
67. Cao, Z.; Long, M.; Wang, J.; Yu, P.S. Hashnet: Deep learning to hash by continuation. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 5608–5617.
68. Song, J.; He, T.; Gao, L.; Xu, X.; Hanjalic, A.; Shen, H.T. Unified Binary Generative Adversarial Network for Image Retrieval and Compression. *Int. J. Comput. Vis.* **2020**, *128*, 2243–2264. [CrossRef]
69. Yang, Y.; Newsam, S. Bag-of-visual-words and spatial extensions for land-use classification. In Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems, San Jose, CA, USA, 2–5 November 2010; pp. 270–279.
70. Xia, G.S.; Hu, J.; Hu, F.; Shi, B.; Bai, X.; Zhong, Y.; Zhang, L.; Lu, X. AID: A benchmark data set for performance evaluation of aerial scene classification. *IEEE Trans. Geosci. Remote Sens.* **2017**, *55*, 3965–3981. [CrossRef]
71. Shao, Z.; Yang, K.; Zhou, W. Performance evaluation of single-label and multi-label remote sensing image retrieval using a dense labeling dataset. *Remote Sens.* **2018**, *10*, 964. [CrossRef]
72. Deng, J.; Dong, W.; Socher, R.; Li, L.J.; Li, K.; Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009; pp. 248–255.
73. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
74. Jin, Z.; Li, C.; Lin, Y.; Cai, D. Density sensitive hashing. *IEEE Trans. Cybern.* **2013**, *44*, 1362–1371. [CrossRef]
75. Slaney, M.; Casey, M. Locality-sensitive hashing for finding nearest neighbors [lecture notes]. *IEEE Signal Process. Mag.* **2008**, *25*, 128–131. [CrossRef]
76. Qiu, Z.; Su, Q.; Ou, Z.; Yu, J.; Chen, C. Unsupervised Hashing with Contrastive Information Bottleneck. *arXiv* **2021**, arXiv:2105.06138.
77. Wang, F.; Liu, H. Understanding the behaviour of contrastive loss. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 2495–2504.