information

Article

Deep Supervised Hashing by Fusing Multiscale Deep Features for Image Retrieval [†]

Adil Redaoui [1], Amina Belalia [2] and Kamel Belloulata [1,*]

[1] RCAM Laboratory, Telecommunications Department, Sidi Bel Abbes University, Sidi bel Abbes 22000, Algeria; redaoui.adil@univ-sba.dz
[2] School of Computer Sciences, Sidi bel Abbes 22000, Algeria; amina.belalia@esi-sba.dz
[*] Correspondence: k_belloula@yahoo.fr; Tel.: +213-7-7371-5910
[†] This paper is an extended version of our paper published in the 1st National Conference on New Educational Technologies and Informatics, NCNETi'23, Guelma, Algeria, 3–4 October 2023.

Abstract: Deep network-based hashing has gained significant popularity in recent years, particularly in the field of image retrieval. However, most existing methods only focus on extracting semantic information from the final layer, disregarding valuable structural information that contains important semantic details, which are crucial for effective hash learning. On the one hand, structural information is important for capturing the spatial relationships between objects in an image. On the other hand, image retrieval tasks often require a more holistic representation of the image, which can be achieved by focusing on the semantic content. The trade-off between structural information and image retrieval accuracy in the context of image hashing and retrieval is a crucial consideration. Balancing these aspects is essential to ensure both accurate retrieval results and meaningful representation of the underlying image structure. To address this limitation and improve image retrieval accuracy, we propose a novel deep hashing method called Deep Supervised Hashing by Fusing Multiscale Deep Features (DSHFMDF). Our approach involves extracting multiscale features from multiple convolutional layers and fusing them to generate more robust representations for efficient image retrieval. The experimental results demonstrate that our method surpasses the performance of state-of-the-art hashing techniques, with absolute increases of 11.1% and 8.3% in Mean Average Precision (MAP) on the CIFAR-10 and NUS-WIDE datasets, respectively.

Keywords: image retrieval; deep learning; multiscale feature; deep supervised hashing

check for updates

Citation: Redaoui, A.; Belalia, A.; Belloulata, K. Deep Supervised Hashing by Fusing Multiscale Deep Features for Image Retrieval. *Information* **2024**, *15*, 143. https://doi.org/10.3390/info15030143

Academic Editors: Mohamed Hammad and Paweł Pławiak

Received: 16 January 2024
Revised: 13 February 2024
Accepted: 22 February 2024
Published: 5 March 2024

1. Introduction

The internet and communication advancements have led to an overwhelming influx of images on the web [1–3], creating a challenge for accurate and efficient large-scale data retrieval. To address this, hash-based image retrieval techniques [4] have gained attention due to their ability to generate compact binary codes, offering computational efficiency and storage advantages.

These techniques can be classified as data-independent and data-dependent approaches. Data-independent methods, such as locality-sensitive hashing (LSH) [5], use random projections as hash functions but suffer from limitations. They do not utilize auxiliary information, leading to poor retrieval accuracy, and require longer codes, consuming more storage [5–7]. In contrast, data-dependent methods leverage training information to learn hashing functions, resulting in shorter codes with improved performance. They can be further categorized as unsupervised [8–11] or supervised hashing methods [12–18].

Deep hashing techniques [19–22] have arisen as a result of the achievements made by deep neural networks in computer vision tasks. These methods, as opposed to traditional hashing methods, possess the ability to effectively extract high-level semantic features and facilitate end-to-end frameworks for generating binary codes. Nevertheless, a drawback of numerous existing deep hashing techniques [23–25] is their reliance on features from the penultimate layer in fully connected networks, which serve as global image descriptors but fail to capture local characteristics.

To overcome these challenges, this paper proposes a novel deep hashing method, called Deep Supervised Hashing by Fusing Multiscale Deep Features (DSHFMDF), which effectively captures multiscale object information. Specifically, it extracts features from different network stages, fuses them at the fusion layer, and encodes them into robust hash codes. The network uses various hashing results based on different scale features, enhancing retrieval recall without sacrificing precision. The key contributions of this paper are as follows:

1. **Integration of Structural Information:** Multiscale deep hashing allows for the integration of structural information from images at multiple levels of granularity. Different scales capture varying levels of structural details in the image. By considering these multiple scales, the model can learn to encode both low-level details and high-level structural features, resulting in more informative hash codes.
2. **Feature Hierarchy:** Deep neural networks used in multiscale hashing often have multiple layers, each capturing features at different abstraction levels. Lower layers capture finer details, while higher layers capture more abstract features or structures. This hierarchical feature representation helps strike a balance by encoding both fine-grained structural details and higher-level abstract information, addressing the trade-off.
3. **Adaptability to Task-Specific Needs:** Multiscale deep hashing can be tailored to the specific requirements of the retrieval task. For tasks where preserving structural information is crucial, appropriate design choices can be made to prioritize encoding such information in the hash codes.
4. **Quantization and Bit Allocation:** Hashing involves quantization of continuous features into binary codes. The bit allocation strategy can be optimized to balance between capturing structural information and achieving high retrieval accuracy. Techniques like adaptive bit allocation can be employed to allocate more bits to preserve crucial structural details while still maintaining retrieval accuracy.
5. **Learning Objectives and Loss Functions:** The design of learning objectives and loss functions can be customized to emphasize the preservation of structural information. Including terms in the loss function that encourage the preservation of certain structural features can guide the learning process.

2. Related Works

In recent years, hashing methods have gained significant attention in the field of image retrieval due to their ability to efficiently store large amounts of data and process the data quickly [6,26]. Like Principal Component Analysis (PCA) [27] and Linear Discriminant Analysis (LDA) [28], which are widely used for dimensionality reduction, the fundamental objective of hashing is to transform high-dimensional input data, such as images, into low-dimensional hash codes. Through this, hashing methods aim to reduce the Hamming distance between similar image pairs while maximizing it for dissimilar pairs, enabling efficient and accurate image retrieval.

The existing literature on hashing methods can be broadly categorized into two types: supervised and unsupervised approaches. Supervised hashing methods utilize labeled data, whereas unsupervised methods operate without the use of any supervision. Unsupervised hashing methods, such as Locality Sensitive Hashing (LSH) [29], Spectral Hashing (SH) [30], and Iterative Quantization (ITQ) [8], are techniques that seek to learn hash functions through unlabeled training samples. These approaches convert input images into binary codes, enabling efficient storage and retrieval. While LSH has been one of the most widely used unsupervised hashing approaches, other methods like SH and ITQ have also been successfully employed in subsequent studies.

Supervised hashing techniques, on the other hand, leverage the availability of labeled data to improve the accuracy of the generated hash codes. These methods outperform unsupervised approaches in terms of retrieval performance. Some notable supervised hashing methods include Supervised Hashing with Kernels (KSH) [31], Minimal Loss Hashing (MLH) [32], and Supervised Discrete Hashing (SDH) [33]. KSH introduces a nonlinear hash function in kernel space to capture complex relationships between image features and their corresponding labels. Instead of directly optimizing hash functions, MLH employs structured Support Vector Machines (SVMs) to create an objective function for learning hash functions. In contrast, SDH prioritizes the generation of top-notch hash codes without any relaxation by redefining the optimization objective.

Due to the swift progress in deep neural networks, deep hashing methods have emerged as a powerful approach in image retrieval. These methods leverage the extensive feature representations provided by deep neural networks to achieve superior performance compared to traditional hand-crafted feature-based approaches. Various deep hashing algorithms have been proposed, including Convolutional Neural Network Hashing (CNNH) [32], Deep Pairwise-Supervised Hashing (DPSH) [34], HashGAN [35], Zhuang et al. [36], Deep Triplet Quantization (DTQ) [37], Supervised Learning of Semantic-Preserving Hash (SSDH) [38], Wang et al. [39], and Similarity-Adaptive Deep Hashing (SADH) [40].

CNNH is an algorithm that focuses on learning hash codes by utilizing features extracted from Convolutional Neural Networks (CNNs). It follows a two-step process, where the hash function learning and feature representation learning are performed independently. By leveraging the rich and high-level representations captured by CNNs, CNNH aims to generate effective hash codes for image retrieval tasks. In contrast, DPSH takes a Bayesian approach to establish a relationship between hash codes and pairwise labels. It optimizes this relationship to learn hash functions that can effectively preserve the pairwise similarities among the data samples. By considering the pairwise label information, DPSH aims to learn more discriminative hash codes that can improve the retrieval performance. HashGAN, on the other hand, introduces the use of Wasserstein Generative Adversarial Networks (GANs) to enhance the training process. It exploits pairwise similarity or dissimilarity information to generate hash codes within a Bayesian framework. By leveraging the power of GANs, HashGAN can effectively increase the amount of training data and generate high-quality hash codes. In the study by Zhuang et al. [36], they propose a binary CNN classifier that incorporates a triplet-based loss. This loss function is designed to learn both semantic links and hashing functions simultaneously. By considering triplets of samples with their corresponding similarities or dissimilarities, the binary CNN classifier aims to learn hash codes that not only preserve the semantic relationships among data samples but also enhance the retrieval accuracy. DTQ takes a different approach by combining a triplet quantization strategy within a supervised deep learning framework. It jointly optimizes the quantization and feature-learning processes to generate discriminative hash codes. By considering the relationships among triplets of samples, DTQ aims to learn hash codes that can preserve the relative distances between data samples and improve retrieval performance. SSDH introduces a novel approach where hash functions are built as new fully connected layers (FC Layers). The learning of hash codes is achieved by minimizing the specified classification error. By incorporating the hash functions as additional layers within the network architecture, SSDH aims to learn compact and discriminative hash codes that can effectively preserve the semantic information in the data. Wang et al. provide a general framework for distance-preserving linear hashing that incorporates deep hashing approaches. This framework aims to learn hash functions that can preserve the pairwise distances between data samples, enabling efficient similarity search. By integrating deep hashing algorithms into the framework, Wang et al. propose an effective way to learn discriminative hash codes for image retrieval tasks. SADH is a two-step hashing algorithm that leverages the output representations from the fully connected layers (FC Layers) to update the similarity graph matrix. By utilizing the FC Layer outputs, SADH aims to improve the optimization process of hash codes. This approach focuses on capturing the intrinsic structure of the data and refining the hash codes accordingly, leading to enhanced retrieval performance. Overall, these approaches showcase various strategies to leverage deep learning techniques for the task of image hashing. They aim to learn effective hash codes by exploiting the power of deep neural networks, considering pairwise relationships, incorporating Bayesian frameworks, and optimizing the hash functions based on different objectives and constraints.

In addition to these methods, recent research has introduced further advancements in image hashing techniques: Deep-Feature Enhancing and Semantic-Preserving Hashing for Image Retrieval (DFEH) [41]. DFEH addresses issues in traditional hashing methods by introducing a feature enhancement layer to improve feature extraction, remove redundant features, and better preserve semantic relationships. It uses contrastive and balance losses to produce compact binary codes. Deep Hashing via Weight Pruning (DHWP) [42] focuses on obtaining short, high-quality hash codes by training models with relatively long hash codes and gradually obtaining shorter codes via weight pruning. It outperforms existing state-of-the-art methods, especially for short hash codes. Deep Momentum Uncertainty Hashing (DMUH) [43] addresses the challenges in deep hashing by explicitly estimating and leveraging uncertainty during training. It models bit-level uncertainty and aims to improve retrieval performance by reducing uncertainty.

While many hashing methods focus on using features from the last fully connected layer (FC), it has been recognized that extracting different types of features can lead to a more comprehensive image description and enhance retrieval performance. Several approaches have been suggested for multiple-level image retrieval. For instance, Lin et al. propose DDH [44], which combines end-to-end learning, divide-and-encode, and hash code learning into a unified framework. DDH employs a stack of convolutional pooling (conv-pool) layers to obtain multiscale features by combining the outputs of the third pooling layer and the fourth convolutional layer. In their work, Yang et al. present Feature Pyramid Hashing (FPH) [45], a novel architecture for image hashing that incorporates two pyramids, namely, vertical and horizontal pyramids. FPH is designed to effectively capture intricate visual details and semantic information, thereby enabling the retrieval of fine-grained images. In [46], Redaoui and Belloulata have proposed Deep Feature Pyramid Hashing (DFPH), which can fully utilize images' multi-level visual and semantic information. Ng et al. introduce a ground-breaking method called multi-level supervised hashing (MLSH) [47] for image retrieval. MLSH focuses on constructing and training distinct hash tables that utilize various levels of features, such as semantic and structural information. By incorporating multiple levels of information, MLSH aims to enhance the accuracy of image retrieval.

In summary, hashing methods in image retrieval have witnessed significant advancements in recent years. From unsupervised approaches to supervised and deep hashing techniques, researchers have explored various methods to learn effective hash functions and generate compact hash codes for efficient image retrieval. Furthermore, the incorporation of multi-level features has shown promise in improving retrieval performance by capturing both fine-grained details and high-level semantics.

3. Proposed Method

In this section, we provide a comprehensive explanation of our proposed approach, Deep Supervised Hashing by Fusing Multiscale Deep Features (DSHFMDF). We start by defining the problem of learning hash codes and subsequently introduce the architecture of our model. Finally, we describe the objective function of our proposed DSHFMDF method.

3.1. Problem Definition

Let $X = \{x_i\}_{i=1}^{N} \in \mathbb{R}^{d \times N}$ represent a training dataset consisting of N images. Here, $Y = \{y_i\}_{i=1}^{N} \in \mathbb{R}^{K \times N}$ denotes the ground truth labels for the x_i samples, where K represents the number of classes. The pairwise label matrix $S = \{s_{ij}\}$ indicates the semantic similarity between training image samples, with $s_{ij} \in \{0, 1\}$. If $s_{ij} = 1$, this signifies that samples x_i and x_j are semantically similar, whereas $s_{ij} = 0$ indicates that they are not. The objective of deep hashing methods is to learn a deep hash function $f : x \mapsto B \in \{-1, 1\}^L$ that encodes each input x_i into $b_i \in \{-1, 1\}^L$. Here, L represents the length of the binary codes.

3.2. Model Architecture

Figure 1 shows the network's architecture of the proposed deep hashing for image retrieval. Our proposed approach for the model architecture is designed to be comprehensive and effective, consisting of five main components: (1) feature extraction; (2) feature reduction; (3) feature fusion; (4) hash coding; (5) and classification. By providing a more detailed and bulky description, we can better understand the intricacies of each component and how they contribute to the overall functionality of the model.

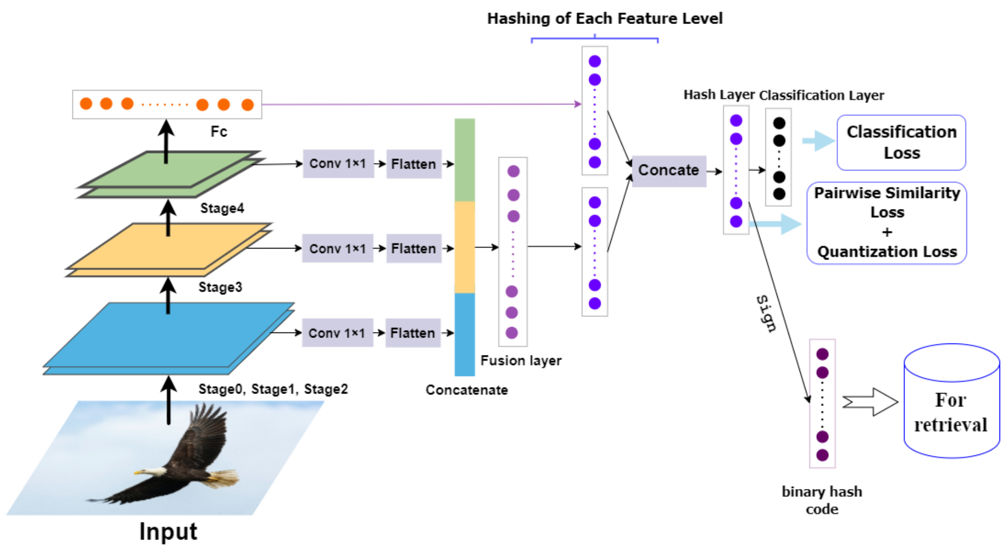


Figure 1. Deep Supervised Hashing by Fusing Multiscale Deep Features (DSHFMDF).

To begin with, the feature extraction stage plays a crucial role in capturing relevant information from the input image. In our approach, we leverage the VGG-19 network as the backbone, which is a deep convolutional neural network known for its ability to extract rich and discriminative features. The VGG-19 architecture comprises several layers, each responsible for extracting features at different levels of abstraction.

During the feature extraction process, we extract features from multiple levels of the VGG-19 network. This includes extracting low-level features that capture structural information at a local level, such as edges and corners, as well as high-level features that capture more abstract and semantic information about the image. By considering features from multiple levels, we aim to capture both fine-grained details and high-level semantic concepts in the image representation.

During the feature extraction process, we adopt a comprehensive approach to extract features from various levels of the VGG-19 network. Specifically, we focus on the feature output of the last convolutional layer within each convolutional block ('conv3', 'conv4', and 'conv5'), as well as the fully connected layer '$fc1$'. We have deliberately chosen to exclude 'conv1' and 'conv2' from this process due to their substantial memory footprint and relatively low semantic information content.

Table 1 provides a detailed overview of the convolutional layers' parameters and the corresponding feature sizes of different convolutional blocks. This multi-level feature-extraction strategy enables us to capture a wide spectrum of image characteristics, ranging from low-level structural details, such as edges and corners, to high-level semantic concepts. By considering features from multiple levels of abstraction, our goal is to create a comprehensive image representation that encompasses fine-grained details and high-level semantic information. This approach allows us to effectively balance local and global information, leading to a more robust and discriminative feature representation for our specific task.

Table 1. Specifics of the feature extraction network are as follows. It is important to note that we utilize the features from layers marked with '#'. For the sake of simplicity, we have omitted the ReLU and Batch Normalization layers.

conv Block	Layers	Kernel Size	Feature Size
1	*conv*2D *conv*2D# MaxPooling	$64 \times 3 \times 3$, $64 \times 3 \times 3$	224×224
2	*conv*2D *conv*2D# MaxPooling	$128 \times 3 \times 3$, $128 \times 3 \times 3$	112×112
3	*conv*2D *conv*2D *conv*2D *conv*2D# MaxPooling	$256 \times 3 \times 3$, $256 \times 3 \times 3$ $256 \times 3 \times 3$, $256 \times 3 \times 3$	56×56
4	*conv*2D *conv*2D *conv*2D *conv*2D# MaxPooling	$512 \times 3 \times 3$, $512 \times 3 \times 3$ $512 \times 3 \times 3$, $512 \times 3 \times 3$	28×28
5	*conv*2D *conv*2D *conv*2D *conv*2D# MaxPooling	$512 \times 3 \times 3$, $512 \times 3 \times 3$ $512 \times 3 \times 3$, $512 \times 3 \times 3$	14×14

After feature extraction, we move on to the feature reduction stage, where we aim to reduce the dimensionality of the extracted features while preserving their discriminative power. To achieve this, we employ a 1×1 convolutional kernel, which acts as a linear combination of the features from different levels. This process helps enhance the depth and robustness of the extracted features while reducing redundancy.

Next, we proceed to the feature fusion layer, which consists of 1024 nodes. At this layer, we connect and combine the different feature levels, allowing for the integration of both low-level and high-level information. This fusion of features from multiple levels helps capture a comprehensive representation of the image, combining both local structural details and global semantic information.

In order to approximate hash codes, we perform a nonlinear mapping of the features from the fusion layer and the fully connected layer (FC). This mapping is accomplished using hash layers, which consist of L nodes representing the desired length of the hash codes. The nonlinear mapping ensures that the generated hash codes capture the essential characteristics of the image representation in a compact and efficient manner.

Moving forward, we concatenate the two hashing layers, resulting in a consolidated representation of the hash codes. This concatenated layer is then connected to the final hashing layer, which further refines the representation and prepares it for classification. By arranging the architecture in this manner, we aim to enhance the preservation of semantic information in the generated hash codes, ensuring that they possess meaningful and discriminative properties.

The classification layer is the last component of our model architecture. It contains neurons equal to the number of classes in the dataset, allowing the network to classify the images based on the learned representations. The classification layer takes advantage of the discriminative power of the hash codes to accurately assign images to their respective classes.

Through the comprehensive approach outlined above, our model utilizes different hashing outcomes based on the various feature levels. This leads to improved image retrieval performance, as the hash codes capture both local and global information. Furthermore, the learning process of the hash codes ensures the preservation of pairwise similarity and the maintenance of semantic information. This results in more meaningful and effective image retrieval based on the learned representations.

3.3. Objective Function

To ensure the learning of similarity-preserving hash codes, our DSHFMDF approach employs three loss functions: pairwise similarity loss, quantization loss, and classification loss. These losses are combined to train our model effectively.

3.3.1. Pairwise Similarity Loss

Our DSHFMDF strives to maintain the resemblances between pairs of input data in a Hamming space. We evaluate pairwise similarity by utilizing the inner product. In particular, the inner product $\langle .,. \rangle$ between hash codes b_i and b_j is precisely defined as $dist_H(b_i, b_j) = \frac{1}{2} b_i^T b_j$.

Given the binary codes $B = \{b_i\}_{i=1}^N$ and the pairwise labels $S = \{s_{ij}\}$, the formulation of the likelihood of pairwise labels is expressed in the following manner:

$$p(s_{ij}|B) = \begin{cases} \sigma(w_{ij}) & s_{ij} = 1 \\ \\ 1 - \sigma(w_{ij}) & s_{ij} = 0 \end{cases} \tag{1}$$

where $\sigma(w_{ij}) = \frac{1}{1+e^{-w_{ij}}}$ and $w_{ij} = \frac{1}{2} b_i^T b_j$.

This formulation implies that a larger inner product $\langle b_i, b_j \rangle$ corresponds to a smaller $dist_H(b_i, b_j)$ and a higher value of $p(1|b_i, b_j)$. Thus, when $s_{ij} = 1$, the binary codes b_i and b_j are considered similar.

Upon calculating the negative log-likelihood of labels on S, we encounter the subsequent optimization problem:

$$J_1 = -\log p(S|B) = -\sum_{s_{ij} \in S} (s_{ij} w_{ij} - \log(1 + e^{w_{ij}})) \tag{2}$$

The optimization problem described above seeks to minimize Hamming distance between similar samples while maximizing the distance between dissimilar points. This objective is in line with the goals of pairwise similarity-based hashing techniques.

3.3.2. Quantization Loss

In practical applications, binary hash codes are commonly used to measure similarity. However, optimizing discrete hash codes within a CNN presents challenges. To overcome this, we propose a continuous form of hash coding. The output of the hash layer is defined as u_i and we set $b_i = \text{sgn}(u_i)$.

To minimize the discrepancy between continuous and discrete hash codes, we introduce the quantization loss as the second objective:

$$J_2 = \sum_{i=1}^{Q} \| b_i - u_i \|_2^2 \tag{3}$$

Here, Q represents the mini-batch size.

3.3.3. Classification Loss

To ensure robust learning of multiscale features throughout the deep network, we employ cross-entropy loss (classification loss) to classify the classes. The formulation of the classification loss is given by

$$J_3 = -\sum_{i=1}^{Q} \sum_{k=1}^{K} y_{i,k} \log(p_{i,k}), \tag{4}$$

In this context, $y_{i,k}$ denotes the true label and $p_{i,k}$ represents the softmax output of the i-th training sample belonging to the k-th class.

To summarize, the overall loss function is obtained by combining the losses from pairwise similarity, pairwise quantization, and classification:

$$J = J_1 + \beta J_2 + \gamma J_3 \tag{5}$$

4. Experiments

We validate the effectiveness of our approach using two publicly available datasets: NUS-WIDE and CIFAR-10. Firstly, we provide a concise overview of these datasets, followed by an exploration of our experimental configurations. Section 4.3 presents the evaluation metrics and baseline methods. Finally, in the concluding section, we present the results of our method, including validations and comparisons with several state-of-the-art hashing techniques.

4.1. Datasets

The CIFAR-10 [48] database, as described in the work by Krizhevsky et al. (2009), comprises a collection of 60,000 images categorized into 10 classes. Each image has a dimension of 32 × 32 pixels. Following the approach outlined in [49], we randomly choose 100 images per class to serve as queries, resulting in a total of 1000 test instances. The remaining images form the database set. Additionally, we randomly select 500 images per category (totaling 5000) from the database to create the training set.

The NUS-WIDE [50] database, introduced by Chua et al. (2009), is a comprehensive collection of approximately 270,000 images sourced from Flickr. It consists of 81 different labels or concepts. For our experiment, we randomly choose 2100 images from 21 classes to form the query database set, while the remaining images serve as the database. Furthermore, we randomly select 10,000 images from the database set to construct the training dataset.

4.2. Experimental Settings

To implement DSHFMDF, we utilize PyTorch as our framework. As a base network, we employ a VGG-19 convolutional network that has been pre-trained on the ImageNet dataset [51]. Throughout our experiments, we train our network using the Adam algorithm [52] with a learning rate of 1×10^{-5}. As for the hyperparameters of the cost function, we assign a value of 0.01 to alpha and 0.1 to beta.

4.3. Evaluation Metrics

In order to assess the performance of various approaches, we employ four evaluation metrics: (MAP) Mean Average Precision, (PR) Precision–Recall curves, Precision curve within Hamming radius 2, and Precision curves with top N returned results (P@N).

We conduct a comparison between our proposed DSHFMDF method and several classical or state-of-the-art methods, which encompass five unsupervised shallow methods, two traditional supervised hashing techniques, and eight deep supervised hashing techniques. In the case of the multi-label CIFAR-10 and NUS-WIDE datasets, samples are considered similar if they share the same semantic labels. Conversely, if the samples have different semantic labels, they are considered dissimilar.

4.4. Results Discussion

The results obtained from our experiments on the CIFAR-10 and NUS-WIDE datasets, evaluating the performance of various hash code lengths, are presented in Table 2. These results are, partially, presented in our preliminary paper [53]. The table clearly shows that our proposed DSHFMDF (Deep Supervised Hashing by Fusing Multiscale Deep Feature) method outperforms all the compared methods by a significant margin. Specifically, when compared to SDH (Supervised Discrete Hashing), which is considered one of the top shallow hashing methods, DSHFMDF demonstrates substantial improvements, with absolute increases of 49.4% and 24% in average Mean Average Precision (MAP) on the

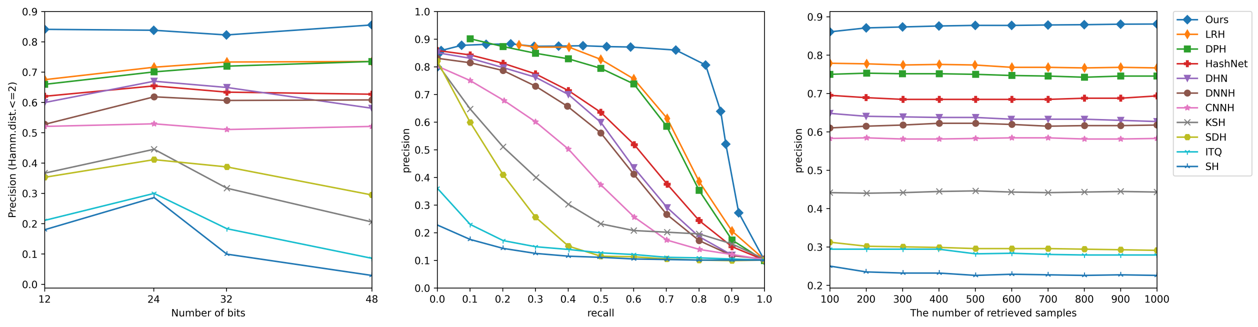
CIFAR-10 and NUS-WIDE datasets, respectively. This highlights the effectiveness of our approach in generating high-quality hash codes for image retrieval tasks.

Table 2. The Mean Average Precision (MAP) scores for Hamming ranking on CIFAR-10 and NUS-WIDE datasets with different numbers of bits. The MAP values are computed based on the top 5000 retrieved images for the NUS-WIDE dataset.

Method	**CIFAR-10 (MAP)**				**NUS-WIDE (MAP)**			
	12 Bits	**24 Bits**	**32 Bits**	**48 Bits**	**12 Bits**	**24 Bits**	**32 Bits**	**48 Bits**
SH [30]	0.127	0.128	0.126	0.129	0.454	0.406	0.405	0.400
ITQ [8]	0.162	0.169	0.172	0.175	0.452	0.468	0.472	0.477
KSH [31]	0.303	0.337	0.346	0.356	0.556	0.572	0.581	0.588
SDH [33]	0.285	0.329	0.341	0.356	0.568	0.600	0.608	0.637
CNNH [54]	0.439	0.511	0.509	0.522	0.611	0.618	0.625	0.608
DNNH [21]	0.552	0.566	0.558	0.581	0.674	0.697	0.713	0.715
DHN [20]	0.555	0.594	0.603	0.621	0.708	0.735	0.748	0.758
HashNet [55]	0.609	0.644	0.632	0.646	0.643	0.694	0.737	0.750
DPH [56]	0.698	0.729	0.749	0.755	0.770	0.784	0.790	0.786
LRH [49]	0.684	0.700	0.727	0.730	0.726	0.775	0.774	0.780
DFEH [41]	0.6753	0.7216	-	0.7864	0.5674	0.5788	0.5863	0.5921
DHWP [42]	0.730	0.729	0.735	0.752	0.796	0.813	0.818	0.822
DMUH [43]	0.772	0.815	0.822	0.826	0.792	0.818	0.825	0.829
Ours	0.779	0.827	0.835	0.845	0.823	0.851	0.851	0.863

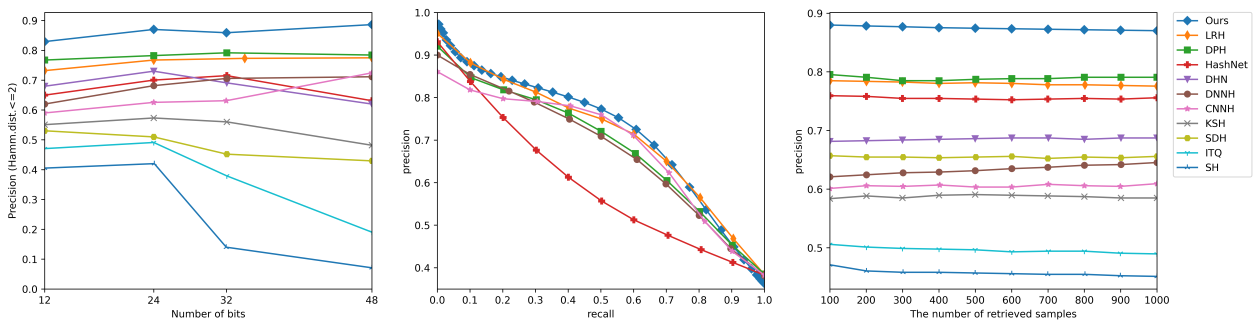
Furthermore, our results indicate that deep hashing methods, including DSHFMDF, outperform traditional hashing techniques. This can be attributed to the ability of deep hashing methods to generate more robust feature representations, leveraging the power of deep neural networks. Among the deep hashing techniques, DSHFMDF surpasses the second-best method, DMUH (Deep Momentum Uncertainty Hashing), achieving average MAP increases of 1.275% and 3.1% on the CIFAR-10 and NUS-WIDE datasets, respectively. These findings highlight the superiority of DSHFMDF in capturing and preserving semantic information in the hash codes, leading to improved retrieval performance.

In order to provide a more detailed analysis of our results, we present Precision curves (P@H = 2) in Figures 2a and 3a, which illustrate the retrieval performance of different methods. Notably, the Precision curves clearly demonstrate that our proposed DSHFMDF model consistently outperforms the other methods as the code length increases. This indicates that our method maintains the highest precision rate, even when longer hash codes are used, showcasing its effectiveness in producing accurate and reliable retrieval results.



(a) Precision within Hamming radius 2 (b) Precision Recall Curve on 48 bits (c) Precision curve with respect to top-N @48 bits

Figure 2. The results obtained by comparing various methods on the CIFAR-10 dataset using three evaluation metrics.



(a) Precision within Hamming radius 2 (b) Precision Recall Curve on 48 bits (c) Precision curve with respect to top-N @48 bits

Figure 3. The results of comparing different approaches on the NUS-WIDE dataset using three evaluation metrics.

To further evaluate the performance of DSHFMDF, we analyze its Precision–Recall (PR) performance and Precision at N (P@N) measures compared to other approaches. Figures 2b,c and 3b,c present the PR performance and P@N results for the CIFAR-10 and NUS-WIDE datasets, respectively. In Figures 2c and 3c, it is evident that the proposed DSHFMDF method achieves the highest precision when using 48-bit hash codes. This demonstrates its effectiveness in generating precise retrieval results. Additionally, Figures 2b and 3b show consistently high precision levels at low recall, which is of great importance in precision-oriented retrieval tasks and has practical application in various systems.

In conclusion, our DSHFMDF method surpasses the compared methods across multiple evaluation aspects, highlighting its superiority in image retrieval tasks. To provide visual evidence of its effectiveness in removing irrelevant images, we present Figure 4, which showcases the retrieval accuracy of various image categories in the CIFAR-10 dataset using DSHFMDF with 48-bit binary codes. The figure includes the query images in the first column, while the subsequent columns display the retrieved images using DSHFMDF. This example further emphasizes the ability of our approach to accurately retrieve relevant images and demonstrates its practical utility.

Overall, our extensive experiments and detailed analysis validate the superiority of the proposed DSHFMDF method in generating high-quality hash codes, improving retrieval performance, and achieving accurate and precise image retrieval results.

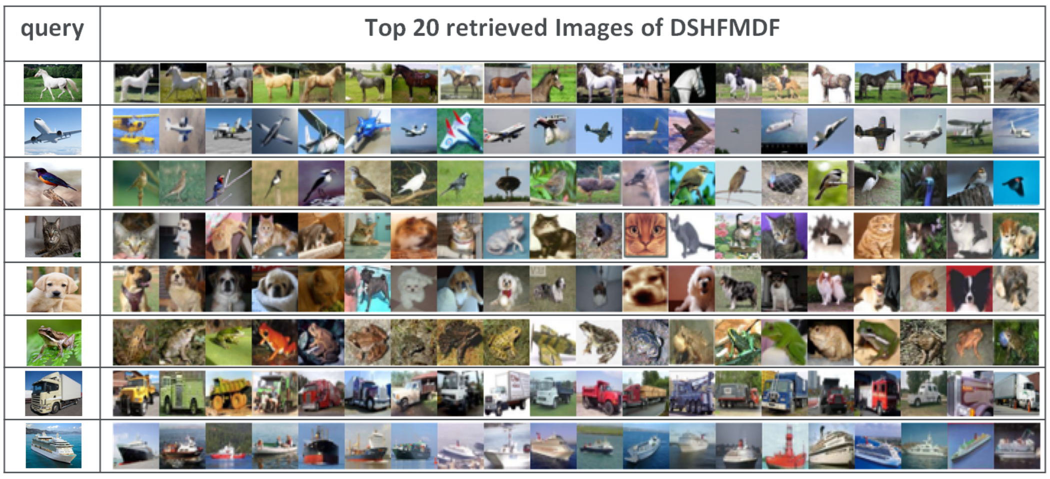


Figure 4. The top 20 retrieved results from the CIFAR-10 dataset using DSHFMDF with 48-bit hash codes are presented. The query images are displayed in the first column, while the subsequent columns showcase the retrieval results obtained by DSHFMDF.

4.5. Ablation Studies

(1) In our ablation studies, we explored different fundamental feature extractors. Specifically, we replaced VGG19 with VGG13 and VGG16, and the performance of these models on the CIFAR-10 dataset is presented in Table 3. The table illustrates that using deeper neural networks can improve image retrieval performance, leading us to choose VGG19 as our primary feature extractor.

Table 3. Mean Average Precision (MAP) of Hamming ranking for different number of bits using VGG13, VGG16, and VGG19 as the fundamental feature extractors on CIFAR-10 dataset. The best results are highlighted in bold.

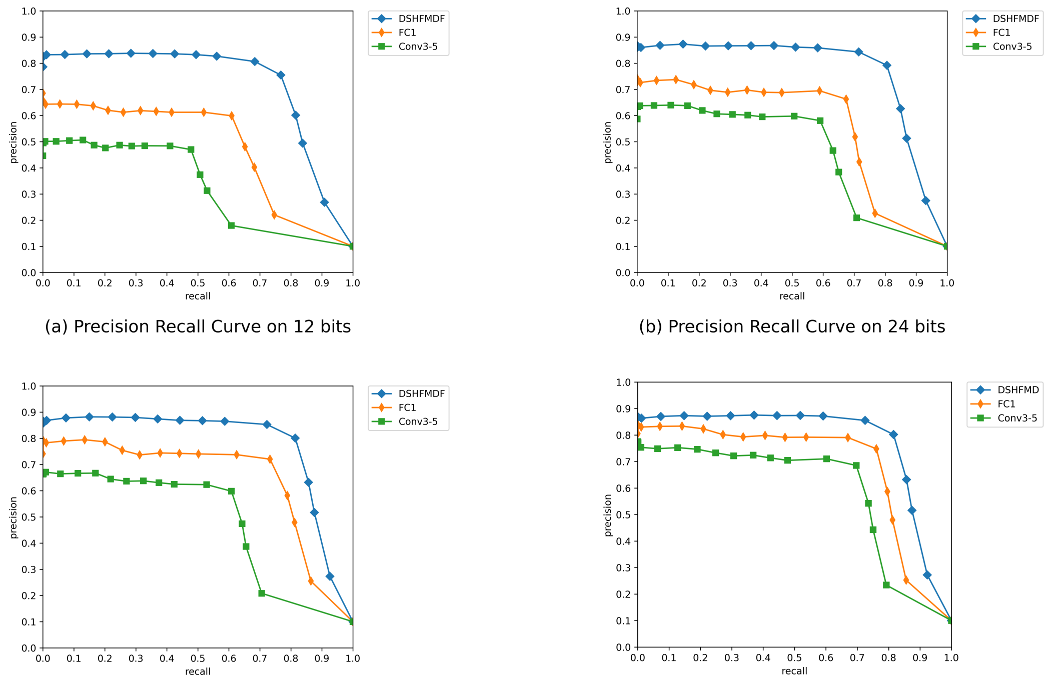
Method	12 Bits	24 Bits	32 Bits	48 Bits	*conv* Layers Num
VGG13	0.726	0.788	0.792	0.806	10
VGG16	0.770	**0.831**	0.819	0.826	13
VGG19	**0.779**	0.827	**0.835**	**0.845**	16

(2) Ablation studies on multi-level image representations for improved hash learning: In our ablation studies, we conducted a thorough investigation into the impact of different levels of image representations on the performance of our DSHFMDF method. Unlike many existing approaches that primarily focus on extracting semantic information from the penultimate layer in fully connected networks, we recognized the importance of structural information that contains crucial semantic details for effective hash learning. To address this, we considered using feature maps from various layers, ranging from low-level to high-level features.
Table 4 presents the results of these experiments, focusing on the retrieval performance using different feature maps on CIFAR-10. Notably, we observed that utilizing features from '$fc1$' led to the most significant average mAP score of 69%, emphasizing the importance of high-level features in capturing semantic information. In contrast, using features from '*conv* 3–5' yielded an average mAP score of only 59.4%, which reflects the significance of low-level and structural details. Furthermore, our proposed DSHFMDF method outperformed other methods, achieving an average mAP score of 82.15%. This result demonstrated that combining features from different scales, including both low-level and high-level features, significantly enhanced the performance of our method.

Table 4. Mean Average Precision (MAP) of Hamming ranking of different scales for different numbers of bits on CIFAR-10.

Method	CIFAR-10 (MAP)			
	12 Bits	24 Bits	32 Bits	48 Bits
$fc1$	0.610	0.667	0.725	0.758
*conv*3–5	0.493	0.595	0.619	0.669
DSHFMDF	0.779	0.827	0.835	0.845

In Figure 5, we display the Precision–Recall curves of DSHFMDF in the case of various scale features. Our DSHFMDF retains over 80% Precision and nearly identical Precision–Recall curves at 12, 24, 36, and 48 hash bits. DSHFMDF achieves superior Precision and Recall with the same hash code length compared to other methods. The binary hash codes perform best when all feature scales are used. This proves that high-level characteristics are more effective in carrying information when creating hash codes. While low-level features can contribute supplementary information to the high-level features information, low-level features cannot entirely take the place of high-level characteristics. The information contained in each scale's features is essential. This further demonstrates how well DSHFMDF makes use of all scales' features.



(a) Precision Recall Curve on 12 bits (b) Precision Recall Curve on 24 bits

(c) Precision Recall Curve on 32 bits (d) Precision Recall Curve on 48 bits

Figure 5. Precision–Recall curve of different scale features on CIFAR-10 with 12, 24, 32, and 48 hash bits.

(3) Ablation studies of the objective function: We conducted ablation studies on our objective function to assess the influence of Pairwise Quantization Loss and Classification Loss constraints, which examine the impact of hash coding and classification on CIFAR-10. We based our experiments on the proposed DSHFMDF method, where β and γ are the key parameters for $J2$ and $J3$. When $\beta = 0$, the model is referred to as DSHFMDF-J3, and when $\gamma = 0$, it is DSHFMDF-J2. As shown in Table 5, if β and γ are not set to zero, each component in the suggested loss function contributes to hash code creation, resulting in a 6.55% performance improvement. Both $J2$ and $J3$ yield similar enhancements because they reduce quantization errors and preserve semantics in the overall model. Eliminating either $J2$ or $J3$ may reduce the model's performance.

Table 5. mAP values of different variants of our objective function for CIFAR-10 dataset.

Method	CIFAR-10 (MAP)			
	12 Bits	**24 Bits**	**32 Bits**	**48 Bits**
DSHFMDF-J2	0.652	0.792	0.790	0.790
DSHFMDF-J3	0.656	0.655	0.783	0.744
DSHFMDF	0.779	0.827	0.835	0.845

4.6. Parameter Sensitivity Analysis

To examine how hyper-parameters β and γ impact retrieval performance, we conducted experiments spanning the values of β and γ from 0.0001 to 0.1, with a tenfold increase, using a 48-bit code on the CIFAR-10 dataset. In Figure 6, we present the changes in mAP curves for DSHFMDF as β and γ vary. Initially, mAP improves as β and γ increase, but performance levels off after reaching their peaks. Notably, our method maintains satisfactory performance over a wide range of β and γ values, demonstrating its robustness to these parameters.

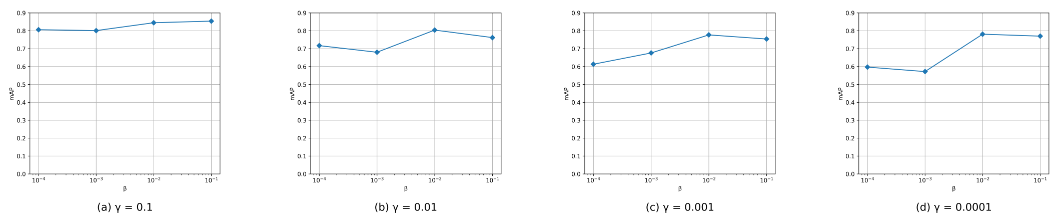


Figure 6. Parameter sensitivity analysis, on CIFAR-10 dataset with 48 bits, of the proposed DSHFMDF.

5. Conclusions and Future Work

This paper presents an end-to-end approach called Deep Supervised Hashing by Fusing Multiscale Deep Features (DSHFMDF) for image retrieval. Our proposed method focuses on generating robust binary codes by optimizing the similarity loss, quantization loss, and semantic loss. Moreover, the network leverages various hashing results based on multiscale features, thereby enhancing retrieval recall while preserving precision. In summary, multiscale deep hashing offers a framework to efficiently incorporate structural information into hash representations, aiding in the balance between structural information and image retrieval accuracy. The careful design of architectures, loss functions, and bit allocation strategies is crucial in achieving this balance based on the specific requirements of the application. Through extensive experiments on two image retrieval datasets, we demonstrate the superiority of our method over other state-of-the-art hashing techniques. As a natural extension of this research, future work will explore the application of our framework to medical image datasets. Medical images often contain objects at various scales, with many being very small. We believe that the use of our method will significantly enhance retrieval performance in the context of medical imaging.

Author Contributions: Conceptualization, A.R. and K.B.; methodology, K.B.; software, A.R.; validation, K.B. and A.R.; formal analysis, A.R., A.B. and K.B.; investigation, K.B. and A.B.; writing—original draft preparation, A.R. and K.B.; writing—review and editing, K.B. and A.R.; visualization, A.R. and A.B.; supervision, K.B. and A.B.; project administration, K.B.; funding acquisition, K.B. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Publicly available datasets were analyzed in this study. These data can be found here: http://www.cs.toronto.edu/~kriz/cifar.html, https://paperswithcode.com/datasets (all accessed on 20 July 2023).

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

Abbreviations

The following abbreviations are used in this manuscript:

DSHFMDF	Deep Supervised Hashing by Fusing Multiscale Deep Features
CBIR	Content-Based Image Retrieval
FPN	Feature Pyramid Network
CNN	Convolutional Neural Network
DCNN	Deep Convolutional Neural Network

References

1. Yan, C.; Shao, B.; Zhao, H.; Ning, R.; Zhang, Y.; Xu, F. 3D room layout estimation from a single RGB image. *IEEE Trans. Multimed.* **2020**, *22*, 3014–3024. [CrossRef]
2. Yan, C.; Li, Z.; Zhang, Y.; Liu, Y.; Ji, X.; Zhang, Y. Depth image denoising using nuclear norm and learning graph model. *ACM Trans. Multimed. Comput. Commun. Appl. (TOMM)* **2020**, *16*, 1–17. [CrossRef]

3. Li, S.; Chen, Z.; Li, X.; Lu, J.; Zhou, J. Unsupervised variational video hashing with 1d-cnn-lstm networks. *IEEE Trans. Multimed.* **2019**, *22*, 1542–1554. [CrossRef]
4. Wang, J.; Zhang, T.; Song, J.; Sebe, N.; Shen, H.T. A survey on learning to hash. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *40*, 769–790. [CrossRef]
5. Gionis, A.; Indyk, P.; Motwani, R. Similarity search in high dimensions via hashing. In Proceedings of the VLDB, Edinburgh, UK, 7–10 September 1999; Volume 99, pp. 518–529.
6. Andoni, A.; Indyk, P. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. *Commun. ACM* **2008**, *51*, 117–122. [CrossRef]
7. Indyk, P.; Motwani, R. Approximate nearest neighbors: Towards removing the curse of dimensionality. In Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing, Dallas, TX, USA, 24–26 May 1998; pp. 604–613.
8. 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]
9. Liu, W.; Wang, J.; Mu, Y.; Kumar, S.; Chang, S.F. Compact hyperplane hashing with bilinear functions. *arXiv* **2012**, arXiv:1206.4618.
10. Gong, Y.; Kumar, S.; Rowley, H.A.; Lazebnik, S. Learning binary codes for high-dimensional data using bilinear projections. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Portland, OR, USA, 23–28 June 2013; pp. 484–491.
11. Lin, G.; Shen, C.; Wu, J. Optimizing ranking measures for compact binary code learning. In Proceedings of the European Conference on Computer Vision, Zurich, Switzerland, 6–12 September 2014; Springer: Cham, Switzerland, 2014; pp. 613–627.
12. Kulis, B.; Darrell, T. Learning to hash with binary reconstructive embeddings. *Adv. Neural Inf. Process. Syst.* **2009**, *22*, 1042–1050.
13. Strecha, C.; Bronstein, A.; Bronstein, M.; Fua, P. LDAHash: Improved matching with smaller descriptors. *IEEE Trans. Pattern Anal. Mach. Intell.* **2011**, *34*, 66–78. [CrossRef] [PubMed]
14. Lin, G.; Shen, C.; Suter, D.; Van Den Hengel, A. A general two-step approach to learning-based hashing. In Proceedings of the IEEE International Conference on Computer Vision, Sydney, Australia, 1–8 December 2013; pp. 2552–2559.
15. Lin, G.; Shen, C.; Shi, Q.; Van den Hengel, A.; Suter, D. Fast supervised hashing with decision trees for high-dimensional data. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 1963–1970.
16. Chen, Z.; Zhou, J. Collaborative multiview hashing. *Pattern Recognit.* **2018**, *75*, 149–160. [CrossRef]
17. Cui, Y.; Jiang, J.; Lai, Z.; Hu, Z.; Wong, W. Supervised discrete discriminant hashing for image retrieval. *Pattern Recognit.* **2018**, *78*, 79–90. [CrossRef]
18. Song, J.; Gao, L.; Liu, L.; Zhu, X.; Sebe, N. Quantization-based hashing: A general framework for scalable image and video retrieval. *Pattern Recognit.* **2018**, *75*, 175–187. [CrossRef]
19. Erin Liong, V.; Lu, J.; Wang, G.; Moulin, P.; Zhou, J. Deep hashing for compact binary codes learning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 2475–2483.
20. Zhu, H.; Long, M.; Wang, J.; Cao, Y. Deep hashing network for efficient similarity retrieval. In Proceedings of the AAAI Conference on Artificial Intelligence, Phoenix, AZ, USA, 12–17 February 2016; Volume 30.
21. 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.
22. Cakir, F.; He, K.; Bargal, S.A.; Sclaroff, S. Hashing with mutual information. *IEEE Trans. Pattern Anal. Mach. Intell.* **2019**, *41*, 2424–2437. [CrossRef]
23. Gordo, A.; Almazán, J.; Revaud, J.; Larlus, D. Deep image retrieval: Learning global representations for image search. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 11–14 October 2016; Springer: Cham, Switzerland, 2016; pp. 241–257.
24. Jiang, Q.Y.; Li, W.J. Asymmetric deep supervised hashing. In Proceedings of the AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018; Volume 32.
25. Shen, F.; Gao, X.; Liu, L.; Yang, Y.; Shen, H.T. Deep asymmetric pairwise hashing. In Proceedings of the 25th ACM International Conference on Multimedia, Mountain View, CA, USA, 23–27 October 2017; pp. 1522–1530.
26. Jin, Z.; Li, C.; Lin, Y.; Cai, D. Density sensitive hashing. *IEEE Trans. Cybern.* **2013**, *44*, 1362–1371. [CrossRef]
27. Li, X.; Wang, H. Adaptive Principal Component Analysis. In Proceedings of the 2022 SIAM International Conference on Data Mining (SDM), Society for Industrial and Applied Mathematics: Virtual Conference, Originally Scheduled in Alexandria, Alexandria, VA, USA, Virtual, 28–30 April 2022; pp. 486–494. [CrossRef]
28. Li, X.; Wang, H. On Mean-Optimal Robust Linear Discriminant Analysis. In Proceedings of the 2022 IEEE International Conference on Data Mining (ICDM), Orlando, FL, USA, 28 November–1 December 2022; pp. 1047–1052. [CrossRef]
29. Datar, M.; Immorlica, N.; Indyk, P.; Mirrokni, V.S. Locality-sensitive hashing scheme based on p-stable distributions. In Proceedings of the Twentieth Annual Symposium on Computational Geometry, Brooklyn, NY, USA, 8–11 June 2004; pp. 253–262.
30. Weiss, Y.; Torralba, A.; Fergus, R. Spectral hashing. *Adv. Neural Inf. Process. Syst.* **2008**, *21*, 1753–1760.
31. Liu, W.; Wang, J.; Ji, R.; Jiang, Y.G.; Chang, S.F. Supervised hashing with kernels. In Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, 16–21 June 2012; pp. 2074–2081.
32. Norouzi, M.; Fleet, D.J. Minimal loss hashing for compact binary codes. In Proceedings of the ICML, Bellevue, WA, USA, 28 June–2 July 2011.

33. Shen, F.; Shen, C.; Liu, W.; Tao Shen, H. Supervised discrete hashing. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 37–45.
34. Li, W.J.; Wang, S.; Kang, W.C. Feature learning based deep supervised hashing with pairwise labels. *arXiv* **2015**, arXiv:1511.03855.
35. Cao, Y.; Liu, B.; Long, M.; Wang, J. Hashgan: Deep learning to hash with pair conditional wasserstein gan. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 1287–1296.
36. Zhuang, B.; Lin, G.; Shen, C.; Reid, I. Fast training of triplet-based deep binary embedding networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 26 June–1 July 2016; pp. 5955–5964.
37. Liu, B.; Cao, Y.; Long, M.; Wang, J.; Wang, J. Deep triplet quantization. In Proceedings of the 26th ACM International Conference on Multimedia, Seoul, Republic of Korea, 22–26 October 2018; pp. 755–763.
38. Yang, H.F.; Lin, K.; Chen, C.S. Supervised learning of semantics-preserving hash via deep convolutional neural networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *40*, 437–451. [CrossRef]
39. Wang, M.; Zhou, W.; Tian, Q.; Li, H. A general framework for linear distance preserving hashing. *IEEE Trans. Image Process.* **2017**, *27*, 907–922. [CrossRef] [PubMed]
40. Shen, F.; Xu, Y.; Liu, L.; Yang, Y.; Huang, Z.; Shen, H.T. Unsupervised deep hashing with similarity-adaptive and discrete optimization. *IEEE Trans. Pattern Anal. Mach. Intell.* **2018**, *40*, 3034–3044. [CrossRef]
41. Zhao, X.; Liu, J. Leveraging Deep Features Enhance and Semantic-Preserving Hashing for Image Retrieval. *Electronics* **2022**, *11*, 2391. [CrossRef]
42. Ma, Z.; Guo, Y.; Luo, X.; Chen, C.; Deng, M.; Cheng, W.; Lu, G. Dhwp: Learning high-quality short hash codes via weight pruning. In Proceedings of the ICASSP 2022—2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Singapore, 23–27 May 2022; pp. 4783–4787.
43. Fu, C.; Wang, G.; Wu, X.; Zhang, Q.; He, R. Deep momentum uncertainty hashing. *Pattern Recognit.* **2022**, *122*, 108264. [CrossRef]
44. Lin, J.; Li, Z.; Tang, J. Discriminative Deep Hashing for Scalable Face Image Retrieval. In Proceedings of the IJCAI, Melbourne, Australia, 19–25 August 2017; pp. 2266–2272.
45. 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.
46. Redaoui, A.; Belloulata, K. Deep Feature Pyramid Hashing for Efficient Image Retrieval. *Information* **2023**, *14*, 6. [CrossRef]
47. Ng, W.W.; Li, J.; Tian, X.; Wang, H.; Kwong, S.; Wallace, J. Multi-level supervised hashing with deep features for efficient image retrieval. *Neurocomputing* **2020**, *399*, 171–182. [CrossRef]
48. Krizhevsky, A.; Hinton, G. Learning Multiple Layers of Features from Tiny Images. 2009. Available online: https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf (accessed on 31 July 2022).
49. Bai, J.; Li, Z.; Ni, B.; Wang, M.; Yang, X.; Hu, C.; Gao, W. Loopy residual hashing: Filling the quantization gap for image retrieval. *IEEE Trans. Multimed.* **2019**, *22*, 215–228. [CrossRef]
50. Chua, T.S.; Tang, J.; Hong, R.; Li, H.; Luo, Z.; Zheng, Y. Nus-wide: A real-world web image database from national university of singapore. In Proceedings of the ACM International Conference on Image and Video Retrieval, Santorini Island, Greece, 8–10 July 2009; pp. 1–9.
51. Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; et al. Imagenet large scale visual recognition challenge. *Int. J. Comput. Vis.* **2015**, *115*, 211–252. [CrossRef]
52. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
53. Redaoui, A.; Belloulata, K. Deep Supervised Hashing with Multiscale Feature Fusion (DSHMFF). In Proceedings of the NCNETi'23, The 1st National Conference on New Educational Technologies and Informatics, Guelma, Algeria, 3–4 October 2023.
54. 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, Québec City, QC, Canada, 27–31 July 2014.
55. 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.
56. Bai, J.; Ni, B.; Wang, M.; Li, Z.; Cheng, S.; Yang, X.; Hu, C.; Gao, W. Deep progressive hashing for image retrieval. *IEEE Trans. Multimed.* **2019**, *21*, 3178–3193. [CrossRef]