Research Article

Improved CNN-Based Hashing for Encrypted Image Retrieval

Wenyan Pan ¹, Meimin Wang ¹, Jiaohua Qin ², and Zhili Zhou ¹

¹Jiangsu Engineering Center of Network Monitoring & School of Computer and Software,
Nanjing University of Information Science and Technology, Nanjing 210044, China

²College of Computer Science and Information Technology, Central South University of Forestry & Technology,
Changsha 410004, China

Correspondence should be addressed to Jiaohua Qin; qinjiaohua@163.com and Zhili Zhou; zhou_zhili@163.com

Received 10 January 2021; Revised 5 February 2021; Accepted 17 February 2021; Published 26 February 2021

Academic Editor: Yuan Tian

Copyright © 2021 Wenyan Pan et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

As more and more image data are stored in the encrypted form in the cloud computing environment, it has become an urgent problem that how to efficiently retrieve images on the encryption domain. Recently, Convolutional Neural Network (CNN) features have achieved promising performance in the field of image retrieval, but the high dimension of CNN features will cause low retrieval efficiency. Also, it is not suitable to directly apply them for image retrieval on the encryption domain. To solve the above issues, this paper proposes an improved CNN-based hashing method for encrypted image retrieval. First, the image size is increased and inputted into the CNN to improve the representation ability. Then, a lightweight module is introduced to replace a part of modules in the CNN to reduce the parameters and computational cost. Finally, a hash layer is added to generate a compact binary hash code. In the retrieval process, the hash code is used for encrypted image retrieval, which greatly improves the retrieval efficiency. The experimental results show that the scheme allows an effective and efficient retrieval of encrypted images.

1. Introduction

With the development of cloud computing, more and more companies and individuals store image data on the cloud server. Therefore, how to efficiently retrieve images in the cloud becomes an urgent problem. Cloud computing [1] is an emerging new computing paradigm with efficient image storage, which makes it an attractive choice for image retrieval. Despite the benefits, image information privacy becomes the main concern with image retrieval in cloud computing.

In order to protect the image information, it is necessary to encrypt the image before it is submitted to the cloud. The widely used encryption methods include chaotic image encryption [2] and Arnold transform [3]. However, it is not suitable to directly apply image retrieval technology in the plaintext domain for image retrieval on the encryption domain. Therefore, how to protect image information in the cloud computing while quickly retrieving the images that users need is an urgent problem that needs to be solved in the field of encrypted image retrieval.

In the field of image retrieval, most previous approaches exploit the frequency domain feature [4, 5], SIFT [6]. However, these approaches are based on hand-crafted features which cannot represent the image content comprehensively because of the low retrieval accuracy.

With the development of deep learning, the CNNs [7–11] have shown significant improvements in the performance on various tasks. However, the most CNNs usually have hundreds of layers, thus making networks more inefficient. Most state-of-the-art lightweight architectures, such as MobileNet [12] and ShuffleNet [13], become more efficient because of their network architectures. These networks can be carried out in a timely fashion on a computationally limited platform.

Even though the CNN-based representation is an appealing solution for image retrieval in the plaintext domain, it is inefficient to directly compute the similarity between two CNN features, such as 4096-dimensional vectors of the full connection layer in AlexNet. Recently, some approaches have been using deep architectures for hash learning for image retrieval [14, 15]. However, most of them are used for

the plaintext domain, but lacks research on the encryption domain.

In order to address the above issues, this paper proposes an improved CNN-based hashing method for encrypted image retrieval (DLHEIR). In our method, we increase the size of the input image of the CNN to obtain better features and replace a part of the structure of the DenseNet network with inverted residual block to reduce the computational cost and parameters. The improved CNNs are used to generate hash codes for encrypted image retrieval.

Our main contributions are as follows:

- (1) This paper proposes an improved CNN-based hashing method for encrypted image retrieval (DLHEIR). This network can learn image representations to generate the binary hash code for rapid image retrieval.
- (2) We used images with larger sizes as input to the CNN to obtain better features. Moreover, the inverted residual block is introduced into our method, which can reduce the computational cost and parameters.

The organization of the remaining part is given as follows. Section 2 discusses the related works. Section 3 introduces the proposed method. Section 4 shows our experimental results, and we conclude this paper in Section 5.

2. Related Work

Content-based image retrieval (CBIR) refers to the retrieving of the needed information in large-scale multimedia data according to the content of the image. Recently, image retrieval has been applied in many fields, such as image search [16, 17] and image steganography [18]. However, it cannot be applied in cloud computing due to the privacy of images.

The searchable encryption (SE) method enables the users to store encrypted data in the cloud computing and supports data search in the encrypted domain. Xia et al. [19] proposed an encrypted image retrieval scheme (PSSE) in the cloud environment, which uses MPEG-7 visual descriptors as image features. The KNN is used to protect features, and the local sensitive hashing is used to improve retrieval efficiency. Qin et al. [20] proposed an encrypted image retrieval approach in the cloud computing environment, which employs the improved Harris algorithm and Local Sensitive Hash (LSH) to retrieve encrypted images. Shen et al. [21] proposed a secure content-based image retrieval method, which uses a secure multiparty computation technique to encrypt image features. Cheng et al. [4] proposed an encrypted JPEG image retrieval scheme based on the Markov process, which uses encryption to encrypt DCT coefficients to protect the confidentiality of the JPEG image content. Xia et al. [22] proposed an outsourcing CBIR scheme based on the BOEW model. Ferreira et al. [23] proposed a secure framework for outsourcing privacy-protected storage and retrieving in a large shared image repository. Lu et al. [24] proposed a privacy protection image retrieval method based on an encrypted image collection which uses a set of visual words to represent images, and the Jaccard distance is used to

measure the similarity between images. Xia et al. [25] proposed a privacy-preserving image retrieval method based on Scale Invariant Feature Transform (SIFT) features and Earth Mover's Distance (EMD). Weng et al. [26] proposed a privacy preserving framework for an application called outsourcing media search. The framework relies on multi-media hashing and symmetric encryption to protect image information. However, these approaches are based on hand-crafted features, which do not consider the global information of the image, resulting in low accuracy for encrypted image retrieval.

CNNs have recently provided an attractive solution for many version tasks. The previous approaches are attributed to the ability of CNN to learn the rich image representations, which can be applied to the field of image retrieval [27, 28]. However, due to the high-computational cost of computing the similarity between two CNN features, some approaches use CNNs to automatically learn binary hashing codes [29–31]. However, these approaches are applicable only in the plaintext domain, and there are few approaches that focus on CNN-based encryption image retrieval.

In this paper, CNNs are applied to the field of encrypted image retrieval. With the powerful representation ability of CNNs' features, the accuracy of encrypted image retrieval is improved. At the same time, the retrieval efficiency is greatly improved by using the hash code.

3. Proposed Method

3.1. System Model. The system model is shown in Figure 1, and the system model mainly consists of three parts: data owner, cloud server, and query user.

Data owner has the image dataset $M = \{m_1, m_2, \dots, m_n\}$. To preserve the image content, the dataset needs to be encrypted, generating the encrypted dataset $E = \{e_1, e_2, \dots, e_n\}$, where n is the number of images in the dataset. To achieve rapid image retrieval, the data owner needs to generate the hash code corresponding to the image dataset. Both the encrypted image and hash code are outsourced to the cloud server. The data owner also needs to send the key to the query user when receiving the retrieval request.

Cloud server stores the encrypted dataset and hash code from the data owner. When receiving the retrieval request from the query user, the cloud server needs to calculate the similarity between the hash code from the data owner and the trapdoor of the query image and returns the top k retrieval results to the query user.

Query user generates the trapdoors for the query images and uploads it to the cloud server. We define the trapdoor as the hash code for query images, which utilize the same method as the data owner does. After receiving the resulting images, the query user sends a request to the data owner and obtains the key, and the user can decrypt the encrypted image with the key.

3.2. Overview of the Proposed Method. The proposed method mainly includes six functions, which are executed by the data owner, cloud server, and query user.

FIGURE 1: System model.

The following functions are executed in the data owner:

- (1) *Key Generation*. $\text{KeyGen}(1^k) \rightarrow K$. The input of the function is parameter k , and it returns the key K . After the user authorization, the data owner sends the key K to the user for decrypting the encrypted image.
- (2) *Image Encryption*. $\text{EncImg}(K, M) \rightarrow E$. The inputs of the function are the key K and the image dataset M , and it returns the encrypted image dataset E .
- (3) *Hash Code Generation*. $\text{hashgen}(M, \text{bits}) \rightarrow \text{HC}$. By adopting our method, the input of the function is the image dataset M , and this function returns the hash code HC .

The following functions are executed in the query user:

- (1) *Trapdoor Generation*. $\text{TrapGen}(Q) \rightarrow \text{HC}_q$. The input of this function is the query image Q . Construct trapdoor and generate hash code HC_q of query image.
- (2) *Image Decryption*. $\text{Dec}(K, R) \rightarrow \text{Img}_R$. The inputs of this function are the key K and the similar encrypted image R returned by the cloud server, and it decrypts the similar encrypted image to return a similar image Img_R .

The following function is executed in the cloud server:

- (1) *Search*. $\text{Search}(\text{HC}, \text{HC}_q) \rightarrow R$. The function calculates the similarity between HC_q corresponding to the query image and the HC corresponding to the encrypted image dataset and it returns similar encrypted image set R .

3.3. Improved Convolutional Neural Network Hashing. In this section, we will introduce our method, which consists of two main components, image preprocessing and network architecture.

3.3.1. Image Preprocessing. Before training or testing the network, the input images should be resized to the same size. For example, when training and testing DenseNet, all images should resize to 224×224 before feeding into the network.

The large image is resized to 224×224 or 299×299 by cropping or warping. The cropping may lose important information of the image, while the warping may change the aspect ratio of the image, and this will affect the features extracted by the CNN.

Consequently, in this paper, we increase the input image size of CNNs. Specifically, for the Core10K dataset, we calculate the maximum image height and width, and then, the largest value height and width are taken as the image size. For example, for the Core10K dataset, the maximum image height and width in the Core10K dataset is 384 and 256, so the size of the input image is resized to 384×384 .

3.3.2. Network Architecture. Inverted Residual Block. The network architecture of our method is shown in Figure 2. Specifically, the image is resized to 384×384 as the input of the DenseNet201. Then, the inverted residual block is introduced to replace a part of the architecture in the DenseNet, which can greatly reduce computational cost and parameters.

The inverted residual block consists of depthwise separable convolution. The computational cost c_d of depthwise separable convolution is shown in the following equation:

$$c_d = k \times k \times C_{in} \times W_{in} \times H_{in} + C_{in} \times W_{out} \times H_{out} \times C_{out}. \quad (1)$$

The parameter p_d of depthwise separable convolution is computed in the following equation:

$$p_d = k \times k \times C_{in} + 1 \times 1 \times C_{in} \times C_{out}. \quad (2)$$

For standard convolutions, the computational cost c_s and parameter p_s are computed by the following equation:

FIGURE 2: Network architecture.

$$c_s = k \times k \times C_{in} \times C_{out} \times H_{out} \times W_{out}, \quad (3)$$

$$p_s = k \times k \times C_{in} \times C_{out}. \quad (4)$$

Suppose the input feature map of depthwise separable convolution has the size $W_{in} \times H_{in} \times C_{in}$ and the output feature map has the size $W_{out} \times H_{out} \times C_{out}$, where C_{in} and C_{out} are the channel of the feature map, W_{in} and W_{out} are the width of the feature map, H_{in} and H_{out} are the height of the feature map, respectively, and $k \times k$ denotes the kernel size. The computational cost ratio r_c of the depthwise separable convolution to standard convolution is shown in the following equation:

$$r_c = \frac{C_{dsc}}{C_{std}} = \frac{1}{C_{out}} + \frac{1}{K^2}. \quad (5)$$

The parameters' cost ratio r_p of the depthwise separable convolution to standard convolution is shown in the following equation:

$$r_p = \frac{P_{dsc}}{P_{std}} = \frac{1}{C_{out}} + \frac{1}{K^2}. \quad (6)$$

Equations (4) and (5) show that the depth separable convolution uses less computational cost and parameters than standard convolution.

Densenet201 consists of four dense blocks, which consists of 6, 12, 48, and 32 BN-ReLU-Conv (1×1)-BN-ReLU-Conv (3×3) structures, respectively, where BN indicates batch normalization, ReLU indicates linear rectifier function, and Conv (1×1) indicates a Conv2D layer with N filters of kernel size 1-by-1. In order to reduce computational cost and parameters of the network, the last 14 BN-ReLU-Conv (1×1)-BN-ReLU-Conv (3×3) were replaced by an inverted residual block. Then, a hash layer is added, which consists of a convolution layer, batch normalization, sigmoid activation, and pooling layer. Finally, SoftMax is added to form our network.

Hash Layer. In this section, we will systematically describe the hash layer. It consists of three main layers, which are a convolutional layer, a batch normalization layers, activation function, and a global average pooling layer. The convolutional layer is a Conv2D layer with N filters of kernel size 1-by-1. For the activation function, we choose sigmoid so that parameters are approximated to (0, 1).

Suppose the input feature map has size of $l \times r \times q$, where l , r , and q are height, width, and channel of the feature map, respectively. The output of the feature map hash layer has size of $1 \times 1 \times N$, and the feature $AP = \{ap_1, ap_2, \dots, ap_q\}$ is obtained.

In feature extraction, firstly, all images are resized to 384×384 before being fed into the network, and the feature AP of the global average pooling layer is extracted, and the binary codes are obtained by using the hash function to binarize AP by a threshold. The hash function is shown in the following equation:

$$HC = \begin{cases} 1, & \text{if } ap_i \geq th, \\ 0, & \text{if } ap_i < th, \end{cases}, \quad i = 1, 2, \dots, q, \quad (7)$$

where ap_i is the parameter in AP and th is the threshold of the hash function.

4. Experimental Results and Analysis

The experiments were performed on the Corel10K dataset [32]. Corel10K is a benchmark dataset for image retrieval. It includes 100 categories, and each category contains 100 similar images.

The experiment code was written in Python and Matlab R2016a under the Windows 10 system, using Intel(R) Core (TM) i7-9700KF CPU @ 3.60GHz, 16.00 GB RAM, and a Nvidia GeForce GTX 2080Ti GPU.

In the experiment, 80 images were randomly selected from each category of the Corel10K dataset as the training set, and the remaining images were used as the test set. DenseNet201 was selected as the backbone network. In the fine-tuning, we use the pretrain model which is trained on the ImageNet dataset. The stochastic gradient descent (SGD) is used as the network optimizer, the learning rate is set to 0.01, the momentum is set to 0.9, the batch size is set to 64, and epochs are set to 200.

4.1. Retrieval Precision. In our experiments, "precision" was used as the evaluation metric, which is defined as $P_k = k'/k$, where k' is the number of real similar images in the k retrieved images. In the experiment, we use the test set as the query image and the training set as the query image

collection to test the retrieval precision. We compare our method with the other methods [6, 17]. The experimental results are reported in Figure 3.

As shown in Figure 3, it is clear that our method outperforms conventional methods [6, 17]. This is because these methods all utilize the hand-craft feature, which limits their performance. In particular, the performance gap is not obvious as k increases, except $k = 100$. Also, note that our method with 48 bits has better performance than the model with others.

We also evaluate the role of image size for retrieval precision. The experimental results are shown in Table 1.

It is clear from Table 1 that the increase in the image scale consistently improves retrieval precision in different hash bits. This is because using larger images is beneficial for performance improvement. A scale larger than our method would instead increase the memory consumption of GPU and computational cost and parameters.

4.2. Comparison of Model Parameters and MFLOPs. In this section, we compare parameters and MFLOPs of our method with the original CNN combined with the hash layer. The experimental results are reported in Table 2.

Floating point operations per second (FLOPs) is a measure of computer performance, which is widely used to measure the computation cost in CNN models, such as ShuffleNet [13]. As can be seen from Table 2, we can find that our method has less parameters and MFLOPs.

4.3. Efficiency. The time consumptions of the retrieval, feature extraction, index construction, and trapdoor generation are compared in this section.

Time Consumption of Retrieval. In order to utilize the powerful computing power of the cloud server, the retrieval is applied in the cloud server, and the most similar k images are returned by calculating the Euclidean distance between two hash codes. Table 3 presents the time consumption of retrieval when retrieving images $k = 20$.

It can be seen from Table 3, the retrieval time consumption increases as the retrieval collection increases. It is clear that our method achieves better efficiency [6, 17]. This is because our method utilized the low-dimension binary hash code, which achieved efficiency in image retrieval.

Time Consumption of Feature Extraction. We also compared the time consumption of feature extraction with the CSD and SCD descriptors in the MPEG-7 feature extraction method of [17], and the time consumption of SIFT feature extraction in [6]. The experimental results are shown in Figure 4.

Figure 4 shows the feature extraction times for different numbers of image collection. Compared with [6, 17], the time consumption of our method is shorter on different numbers of image collections in most cases. This is because the time consumption of feature extraction in our method mainly consists of two parts: time consumption of the load model and hashing. Compared with complex conventional methods, our method is more efficient.

FIGURE 3: Top- k retrieval precision.

TABLE 1: Retrieval precision of different input image sizes (%).

Method		k				
Image size	Bits	20	40	60	80	100
384×384	12 bits	73.92	73.56	73.04	72.55	60.38
	24 bits	82.33	82.31	82.36	82.29	67.21
	32 bits	84.82	84.80	84.75	84.69	69.06
	48 bits	86.23	86.21	86.15	86.08	69.91
224×224	12 bits	72.55	72.29	72.13	71.85	59.63
	24 bits	80.10	80.02	80.01	79.90	65.23
	32 bits	81.50	81.42	81.38	81.34	66.37
	48 bits	82.26	82.24	82.19	82.11	66.01

TABLE 2: Compare the number of parameters and MFLOPs for different methods.

Method	Parameters	MFLOPs
DLHEIR (48 bits)	176.64	158.25
DLHEIR (32 bits)	176.54	158.25
DLHEIR (24 bits)	176.49	158.21
DLHEIR (12 bits)	176.41	158.13
Original CNN + hash layer (48 bits)	184.19	164.97
Original CNN + hash layer (32 bits)	183.87	164.68
Original CNN + hash layer (24 bits)	183.71	164.54
Original CNN + hash layer (12 bits)	183.46	164.32

Time Consumption of Index Construction. In our method, the similarity is directly computed by two hash codes without index construction, so there is no time consumption of index construction in our method. The time consumption of index construction comparison with Xia and Qin is shown in Figure 5.

TABLE 3: Time consumption of retrieval.

Methods	Number of images in data collection				
	2000	4000	6000	8000	10000
DLHEIR (48 bits)	0.58	0.742	0.908	1.073	1.249
DLHEIR (32 bits)	0.537	0.657	0.776	0.894	1.016
DLHEIR (24 bits)	0.505	0.608	0.637	0.799	0.908
DLHEIR (12 bits)	0.483	0.554	0.637	0.703	0.785
Xia (CSD)	2.51	4.60	6.26	8.34	10.49
Xia (SCD)	1.65	3.79	5.36	6.53	8.78
Qin	0.95	1.46	2.25	2.88	3.63

FIGURE 4: Time consumption of feature extraction (s).

FIGURE 5: Time consumption of index construction (ms).

Time Consumption of Trapdoor Generation Time. Similar to the feature extraction, the trapdoor generation incurs the hash code generated by the data owner, so the time

FIGURE 6: Time consumption of trapdoor generation (ms).

consumption of the trapdoor construction is the hash code generation time of the query image. The experimental results are shown in Figure 6.

We test the time consumption of trapdoor generation compared with the [6, 17] in Figure 6. Our method has more time consumption to these methods. This is because, in feature extraction, we need to extract features from the deep layer of DenseNet, so the time consumption is more than [6, 17].

4.4. Security Analysis

- (i) *The Privacy of Image Content.* In our method, the images stored on the cloud server are encrypted with an encryption method. The key is generated by the data owner. Thus, the privacy of the image content in our scheme is well protected.
- (ii) *The Privacy of Hash Code.* The hash code may reveal the information about the image content. In our method, the hash code mapped from the feature vectors are protected by a one-way hash function. Thus, the hash code is well protected.

5. Conclusion

This paper proposes an improved CNN-based hashing method for encrypted image retrieval. In our method, we increase the size of the input image of the CNN to obtain better features and replace part of the structure of the DenseNet network with inverted residual block to reduce the computational cost and parameters, and a hash layer is added for hash code generation. These hash codes are used for encrypted image retrieval. The experimental results show that the method achieves better performance and greatly improves the retrieval efficiency. In the future, we plan to design more efficient methods to reduce the burden on users.

Data Availability

The Corel10K data used to support the findings of this study have been deposited in the “<http://www-db.stanford.edu/~wangz/image.vary.jpg.tar>.”

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant nos. 61972205, U1836208, and U1836110, National Key R&D Program of China under Grant 2018YFB1003205, Priority Academic Program Development (PAPD) of Jiangsu Higher Education Institutions Fund, Collaborative Innovation Center of Atmospheric Environment and Equipment Technology (CICAET) Fund, China, and Ministry of Science and Technology (MOST), Taiwan, under Grant nos. 108-2221-E-259-009-MY2 and 109-2221-E-259-010.

References

- [1] L. Wang, G. Von Laszewski, A. Younge et al., “Cloud computing: a perspective study,” *New Generation Computing*, vol. 28, no. 2, pp. 137–146, 2010.
- [2] X.-Y. Wang, L. Yang, R. Liu, and A. Kadir, “A chaotic image encryption algorithm based on perceptron model,” *Nonlinear Dynamics*, vol. 62, no. 3, pp. 615–621, 2010.
- [3] W. Chen, C. Quan, and C. J. Tay, “Optical color image encryption based on Arnold transform and interference method,” *Optics Communications*, vol. 282, no. 18, pp. 3680–3685, 2009.
- [4] H. Cheng, X. Zhang, J. Yu, and F. Li, “Markov process-based retrieval for encrypted JPEG images,” *EURASIP Journal on Information Security*, vol. 2016, no. 1, 1 page, 2016.
- [5] R. Bellafqira, G. Coatrieux, D. Bouslimi, G. Quellec, and M. Cozic, “Secured outsourced content based image retrieval based on encrypted signatures extracted from homomorphically encrypted images,” arXiv preprint arXiv:1704.00457, 2017.
- [6] J. Qin, Y. Cao, X. Xiang, Y. Tan, L. Xiang, and J. Zhang, “An encrypted image retrieval method based on SimHash in cloud computing,” *Computers, Materials & Continua*, vol. 62, no. 3, pp. 389–399, 2020.
- [7] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.
- [8] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” arXiv preprint arXiv:1409.1556, 2014.
- [9] C. Szegedy, W. Liu, Y. Jia et al., “Going deeper with convolutions,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–9, Boston, MA, USA, June 2015.
- [10] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. B. Wojna, “Rethinking the inception architecture for computer vision,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2818–2826, Las Vegas, NV, USA, June 2016.
- [11] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, Las Vegas, NV, USA, June 2016.
- [12] A. G. Howard, M. Zhu, B. Chen et al., “Mobilenets: efficient convolutional neural networks for mobile vision applications,” arXiv preprint arXiv:1704.04861, 2017.
- [13] X. Zhang, X. Zhou, M. Lin, and J. Sun, “Shufflenet: an extremely efficient convolutional neural network for mobile devices,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6848–6856, Salt Lake City, Utah, June 2018.
- [14] K. Lin, H. F. Yang, J. H. Hsiao, and C.-S. Chen, “Deep learning of binary hash codes for fast image retrieval,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 27–35, Boston, MA, USA, June 2015.
- [15] H. Liu, R. Wang, S. Shan, and X. Chen, “Deep supervised hashing for fast image retrieval,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2064–2072, Las Vegas, NV, USA, June 2016.
- [16] Z. Zhou, Q. M. J. Wu, Y. Yang, and X. Sun, “Region-level visual consistency verification for large-scale partial-duplicate image search,” *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 16, no. 2, pp. 1–25, 2020.
- [17] A. Gordo, J. Almazán, J. Revaud, and D. Larlus, “Deep image retrieval: learning global representations for image search,” in *Proceedings of the European Conference on Computer Vision*, pp. 241–257, Amsterdam, The Netherlands, October 2016.
- [18] Z. Zhou, Y. Mu, and Q. M. J. Wu, “Coverless image steganography using partial-duplicate image retrieval,” *Soft Computing*, vol. 23, no. 13, pp. 4927–4938, 2019.
- [19] Z. Xia, X. Wang, L. Zhang, Z. Qin, X. Sun, and K. Ren, “A privacy-preserving and copy-deterrence content-based image retrieval scheme in cloud computing,” *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 11, pp. 2594–2608, 2016.
- [20] J. Qin, H. Li, X. Xiang et al., “An encrypted image retrieval method based on Harris corner optimization and LSH in cloud computing,” *IEEE Access*, vol. 7, pp. 24626–24633, 2019.
- [21] M. Shen, G. Cheng, L. Zhu, X. Du, and J. Hu, “Content-based multi-source encrypted image retrieval in clouds with privacy preservation,” *Future Generation Computer Systems*, vol. 109, pp. 621–632, 2020.
- [22] Z. Xia, L. Jiang, D. Liu, L. Lu, and B. Jeon, “BOEW: a content-based image retrieval scheme using bag-of-encrypted-words in cloud computing,” *IEEE Transactions on Services Computing*, no. 1, p. 1, 2019.
- [23] B. Ferreira, J. Rodrigues, J. Leitao, and H. Domingos, “Practical privacy-preserving content-based retrieval in cloud image repositories,” *IEEE Transactions on Cloud Computing*, vol. 7, no. 3, pp. 784–798, 2019.
- [24] W. Lu, A. Swaminathan, A. L. Varna, and M. Wu, “Enabling search over encrypted multimedia databases. Media forensics and security,” *International Society for Optics and Photonics*, vol. 7254, p. 725418, 2009.
- [25] Z. Xia, Y. Zhu, X. Sun, Q. Zhan, and K. Ren, “Towards privacy-preserving content-based image retrieval in cloud computing,” *IEEE Transactions on Cloud Computing*, vol. 6, no. 1, pp. 276–286, 2018.
- [26] L. Weng, L. Amsaleg, and T. Furon, “Privacy-preserving outsourced media search,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 10, pp. 2738–2751, 2016.

- [27] A. S. Razavian, J. Sullivan, S. Carlsson, and A. Maki, “[Paper] visual instance retrieval with deep convolutional networks,” *ITE Transactions on Media Technology and Applications*, vol. 4, no. 3, pp. 251–258, 2016.
- [28] M. Tzelepi and A. Tefas, “Relevance feedback in deep convolutional neural networks for content based image retrieval,” in *Proceedings of the 9th Hellenic Conference on Artificial Intelligence*, pp. 1–7, Thessaloniki Greece, May 2016.
- [29] V. A. Nguyen and M. N. Do, “Deep learning based supervised hashing for efficient image retrieval,” in *Proceedings of the 2016 IEEE International Conference on Multimedia and Expo (ICME)*, pp. 1–6, Seattle, WA, USA, July 2016.
- [30] R. Zhang, L. Lin, R. Zhang, W. Zuo, and L. Zhang, “Bit-scalable deep hashing with regularized similarity learning for image retrieval and person re-identification,” *IEEE Transactions on Image Processing*, vol. 24, no. 12, pp. 4766–4779, 2015.
- [31] X. Li, Q. Xue, and M. C. Chuah, “CASHEIRS: cloud assisted scalable hierarchical encrypted based image retrieval system,” in *Proceedings of the IEEE INFOCOM 2017-IEEE Conference on Computer Communications*, pp. 1–9, Atlanta, GA, USA, May 2017.
- [32] J. Z. Wang, *Semantics-sensitive Integrated Matching for Picture Libraries and Biomedical Image Databases*, Stanford University, Stanford, CA, USA, 2000.