






Deep Multi-Scale Hashing for Image Retrieval (DMSH)

Adil Redaoui¹ , Kamel Belloulata¹ , and Amina Belalia² 

¹ Telecommunications Department, RCAM Laboratory, Sidi Bel Abbès, Algeria
{adil.redaoui,kamel.belloulata}@univ-sba.dz

² High School of Computer Sciences of Sidi Bel Abbès, Sidi Bel Abbès, Algeria
a.belalia@esi-sba.dz

Abstract. In recent years, with the great success of deep learning, deep networks-based hashing has become a leading approach for image retrieval. Most existing deep hashing methods extract the semantic representations only from the last layer, resulting in structure information being ignored which contains additional semantic details that are useful for hash learning. To enhance the image retrieval accuracy by exploring the semantic information and the structure information (local information), We propose a new method of deep hashing called Deep Multi-Scale Hashing (DMSH). This is achieved, firstly, by extracting multiscale features from multiple convolutional layers. Secondly, the features extracted from the convolutional layers are fused to generate more robust representations for efficient image retrieval. The experiments on the CIFAR10 and NUS-WIDE datasets show the superiority of our method.

Keywords: Deep learning · Deep supervised hashing · Image retrieval · Feature pyramid · Multi-scale feature

1 Introduction

The rapid advances in the internet and communication have resulted in images' massive overloading to the internet [22, 40, 41], making it extremely difficult to retrieve large-scale data accurately and effectively, representing a practical research problem. The hash-based image retrieval technique [36] has attracted increasing attention to guarantee retrieval quality and computation efficiency. The hashing methods perform to map images into compact binary codes to take advantage of binary codes' superior computational and storage capabilities.

Existing hashing techniques can be divided into groups that are data-dependent and Data-independent. Data-independent approaches, representing locally-sensitive hashing (LSH) [11], are used in random projections as the hash functions to learn binary codes. However, these methods do have some limitations. The data-independent methods do not use auxiliary information, which makes them suffer in terms of poor image retrieval accuracy. Furthermore, They

require long codes that cost plenty of storage [1, 11, 15]. To address this problem, data-driven methods exploit training information to create hash functions, obtaining shorter binary codes with better performance. Moreover, they can be categorized-into unsupervised hashing methods [12, 13, 25, 29] and supervised hashing methods [7, 9, 20, 23, 24, 33, 34].

On the other hand, deep hashing techniques [4, 10, 21, 42] have been proposed after the great success of deep neural networks on many computer vision tasks. Compared with traditional hashing approaches, deep hashing techniques exhibit advantages in extracting high-level semantic features and producing binary codes into an end-to-end framework. However, many recent works on deep hashing methods [14, 17, 31] used the penultimate layer features in fully connected layers as the global image descriptor. However, they are undesirable image features because the high-level features exhibit global details but lack local characteristics.

To address the above problems, in this paper, we propose a new deep hashing method called Deep Multi-Scale Hashing (DMSH), which use FPN in deep hashing—using FPN. Extracting the multi-level semantic and visual information from the input image is facilitated. To be more specific, An FPN builds a pathway of bottom-up and the top-down features with links between the features the network produces at different scales. The hash codes are then learned from these feature scales and fuse to obtain the final ones. Moreover, the network employs various hashing results depending on different scale features. Consequently, the network would improve retrieval recall while preventing precision degradation. The main contributions of the work, in brief, are as follows:

1. A Deep Multi-Scale Hashing (DMSH) learns hash codes from various feature scales and fuses them to generate the final binary codes. As a result, the network will be able to retrieve better.
2. A new deep hashing method is proposed that conducts joint optimization of feature representation learning and binary codes learning in a deep, unified framework.
3. Experimentation based on two large-scale data-sets show that DMSH provides state-of-the-art performance in real applications.

2 Proposed Method

2.1 Problem Definition

Let $X = \{x_i\}_{i=1}^N \in \mathbb{R}^{d \times N}$ denote the training dataset with N image samples, where $Y = \{y_i\}_{i=1}^N \in \mathbb{R}^{K \times N}$ represents the ground truth labels of the x_i , K is the number of classes. Let's say that the pairwise labels for training images are represented by the matrix $S = \{s_{ij}\}$, where $s_{ij} = 0$ represents no semantic similarity between samples x_i and x_j and $s_{ij} = 1$ represents semantic similarity. The objective of deep hashing methods with pairwise labels is to Learning a nonlinear hash function $f : x \mapsto B \in \{-1, 1\}^L$, which can the ability to convert each input data x_i into binary codes $b_i \in \{-1, 1\}^L$, L is the Hash code length.

2.2 Model Architecture

Lin [26] suggested that a feature pyramid network was applied to improve object detection in RetinaNet [27]. FPN aids the network in learning more effectively and detecting objects at various scales present in the image. Some previous methods worked by providing input to the network, such as an image pyramid. While doing this does enhance the feature extraction procedure, it also lengthens processing time and is less effective.

FPN solved this issue by creating the bottom-up and top-down connection of entities and merging them with network entities created at different levels via a lateral connection.

Figure 1 displays the architecture of our proposed method (DMSH). We used VGG-19 as the backbone network. The FPN we used is similar to the original FPN [26], with the difference that, in light of the authors’ expertise, we employed concatenation layers rather than adding layers within the feature pyramid. FPN extracts four final features, each presenting the input image’s features at various scales. The features obtained are fused using a convolutional (Conv1 × 1) layer to get a merged feature and then apply dropout layers, followed by the first hash layers.

At the end of the design, we assembled the five hash layers. We then connected the latter to the final hash layer, and finally, we connected the final hash layer to the classification layer (the number of neurons in the classification layer is equal to the number of categories of the dataset). With this procedure, the network uses various hash results based on different features scale. The network will be able to retrieve images more effectively.

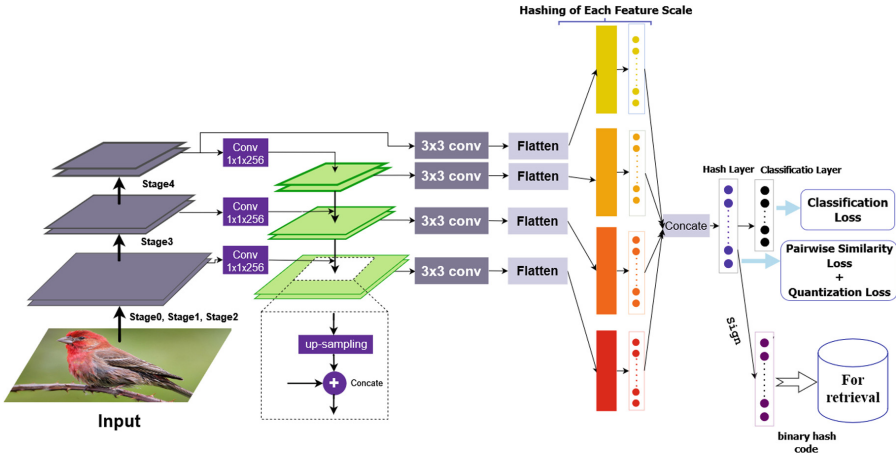


Fig. 1. The architecture of our proposed method: Deep Multi-Scale Hashing (DMSH).

Table 1. Details of the feature extraction network. Note that we use the features of the layers marked by “#”. For simplicity, we omit ReLU and batch normalization layers.

Conv block	Layers	Kernel size	Feature size
1	Conv2D	$64 \times 3 \times 3$	224×224
	Conv2D#	$64 \times 3 \times 3$	
	MaxPooling	2×2	
2	Conv2D	$128 \times 3 \times 3$	112×112
	Conv2D#	$128 \times 3 \times 3$	
	MaxPooling	2×2	
3	Conv2D	$256 \times 3 \times 3$	56×56
	Conv2D	$256 \times 3 \times 3$	
	Conv2D	$256 \times 3 \times 3$	
	Conv2D#	$256 \times 3 \times 3$	
	MaxPooling	2×2	
4	Conv2D	$256 \times 3 \times 3$	28×28
	Conv2D	$256 \times 3 \times 3$	
	Conv2D	$256 \times 3 \times 3$	
	Conv2D#	$256 \times 3 \times 3$	
	MaxPooling	2×2	
5	Conv2D	$256 \times 3 \times 3$	14×14
	Conv2D	$256 \times 3 \times 3$	
	Conv2D	$256 \times 3 \times 3$	
	Conv2D#	$256 \times 3 \times 3$	
	MaxPooling	2×2	

2.3 Objective Function

Pairwise Similarity Loss. We carry out our deep hashing method by maintaining the greatest similarity between each pair of images in the Hamming space. The inner product is used to calculate the pairwise similarity. For two binary codes, b_i and b_j , the inner product is written as follows: $dist_H(b_i, b_j) = \frac{1}{2}b_i^T b_j$

Given that all binary codes of points are $B = \{b_i\}_{i=1}^N$, we can write the likelihood of pairwise labels $S = \{s_{ij}\}$ as follows:

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

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

According to the equation above, we may deduce that the greater the inner product $\langle b_i, b_j \rangle$ corresponds to a smaller equivalent $dist_H(b_i, b_j)$ and a larger $p(1|b_i, b_j)$.

This indicates that when $s_{ij} = 1$, the hash codes b_i and b_j are regarded as similar, and vice versa.

We may get the following optimization problem by considering the negative log-likelihood of the pairwise labels in S :

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

The above optimization problem makes the distance between two similar points as small as possible, precisely what pairwise similarity-based hashing approaches aim to achieve.

Pairwise Quantization Loss. In real-world applications, discrete hash codes (binary codes) measure similarity. Optimizing discrete hash coding (binary) in CNN is difficult, however. Therefore, gradient disappearance through the back-propagation stage is avoided using the continuous hash coding version.

Discrete hash codes are utilized in real-world applications to determine similarity. However, discrete hash coding in CNN is challenging to optimize. Therefore, continuous hash coding prevents gradient disappearance during the back-propagation phase. Where the hash layer output is u_i and $b_i = \text{sgn}(u_i)$. Hence, quantization loss has been introduced to reduce the gap between discrete and continuous hash codes. An objective function is defined as

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

Q is the mini-batches.

Classification Loss. We apply the classification loss (the cross-entropy loss) to identify the classes to obtain robust multiscale features. The classification loss function can be expressed as follows:

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

where $y_{i,k}$ is the label, $p_{i,k}$ is the output of the i -th training sample, which corresponds to the k -th class.

In conclusion, quantization loss, pairwise similarity loss, and classification loss can be combined to produce the overall loss function:

$$J = J_1 + \beta J_2 + \gamma J_3 \quad (5)$$

3 Experiments

3.1 Datasets

The **CIFAR-10** [19] contains 60,000 images of 10 different categories, and each image is 32×32 in size. Following [2], we sampled 100 images per category as a

query set (a total of 1000) and the remaining images as a database. Additionally, 500 images per category are selected (a total of 5000) from the retrieval database as a training set.

NUS-WIDE. [8] is a multi-label data set containing approximately 270,000 images collected from Flickr consisting of 81 ground-truth concepts. We randomly sampled 2100 images from the 21 most frequent classes as the query set, while the rest as a database, and selected 10,000 images from the retrieval database as a training set.

3.2 Experimental Settings

Our DMSH implementation utilizes PyTorch. We use a convolutional network called VGG-19, pre-trained on ImageNet [30]. In all training, we use the Adam [18] algorithm. For the hyperparameters of the objective function, the beta to 0.1 and the alpha is set to 0.01.

3.3 Evaluation Metrics

We use four evaluation metrics to measure the retrieval performance of different hashing methods: Mean Average Precision (MAP), Precision-Recall curves, precision curves w.r.t, and Precision curve within Hamming radius 2. Five unsupervised approaches are included in our comparison of the proposed DMSH, i.e., SGH [16], LSH [11], SH [38], ITQ [13], PCAH [37], and two supervised approaches, i.e., KSH [28], SDH [32], and eight deep hashing approaches, i.e., DPH [3], CNNH [39], DNNH [21], DCH [5], DHN [42], HashNet [6], LRH [2], DHDW [35].

3.4 Results

Tables 2 and 3 displays the MAP results for all methods on the CIFAR-10 and NUS-WIDE data sets with different code lengths. The results in the table demonstrate how well the proposed DMSH method performs compared to all other techniques. In particular, on the datasets CIFAR-10 and NUS-WIDE, DMSH delivers absolute gains in average mAP of 49.4% and 24.44%, respectively, above SDH, the top shallow hashing technique. However, we can observe that deep hashing performs better than classical hashing techniques, mainly because it can produce more reliable representations of features. For deep hashing methods, On CIFAR-10 and NUS-WIDE, respectively, the average mAP of our suggested DMSH approach increased by 11.5% and 8.3% per cent compared to the second-best hashing method LRH.

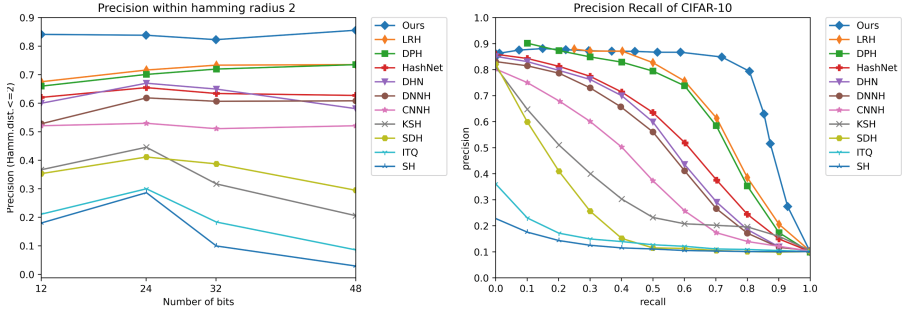
Table 2. Mean Average Precision (MAP) of hamming ranking for different number of bits on CIFAR-10.

Method	CIFAR-10 (MAP)			
	12 bits	24 bits	32 bits	48 bits
SH [38]	0.127	0.128	0.126	0.129
ITQ [13]	0.162	0.169	0.172	0.175
KSH [28]	0.303	0.337	0.346	0.356
SDH [32]	0.285	0.329	0.341	0.356
CNNH [39]	0.439	0.511	0.509	0.522
DNNH [21]	0.552	0.566	0.558	0.581
DHN [42]	0.555	0.594	0.603	0.621
HashNet [6]	0.609	0.644	0.632	0.646
DPH [3]	0.698	0.729	0.749	0.755
LRH [2]	0.684	0.700	0.727	0.730
DMSH	0.800	0.823	0.838	0.840

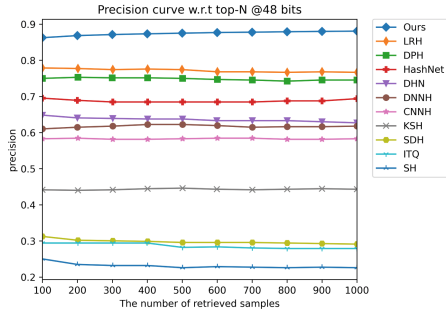
Table 3. Mean Average Precision (MAP) of hamming ranking for different number of bits on NUS-WIDE, The MAP values are calculated on the top 5,000 retrieval images.

Method	NUS-WIDE (MAP)			
	12 bits	24 bits	32 bits	48 bits
SH [38]	0.454	0.406	0.405	0.400
ITQ [13]	0.452	0.468	0.472	0.477
KSH [28]	0.556	0.572	0.581	0.588
SDH [32]	0.568	0.600	0.608	0.637
CNNH [39]	0.611	0.618	0.625	0.608
DNNH [21]	0.674	0.697	0.713	0.715
DHN [42]	0.708	0.735	0.748	0.758
HashNet [6]	0.643	0.694	0.737	0.750
DPH [3]	0.770	0.784	0.790	0.786
LRH [2]	0.726	0.775	0.774	0.780
DMSH	0.826	0.850	0.853	0.859

Precision curves represent the retrieval performance in Figs. 2a and 3a ($P@H = 2$). The proposed DMSH performs noticeably better than alternative approaches. According to the precision curves, the proposed DMSH approach still has the highest precision rate when the code length rises.



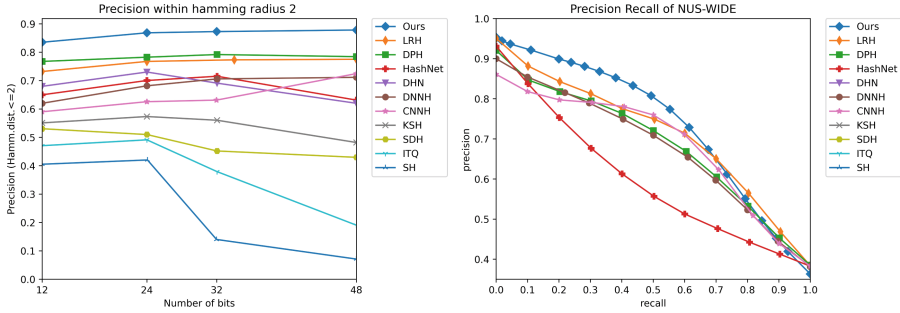
(a) Precision within hamming radius 2 (b) Precision Recall Curve on 48 bits



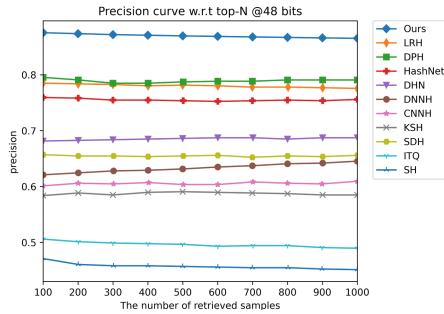
(c) Precision curve w.r.t top-N @48 bits

Fig. 2. The comparison results on the CIFAR-10 dataset under three evaluation metrics.

We further highlight the effectiveness of our method in Figs. 2b, 3b, 2c and 3c, where we compare our method’s precision concerning top returned samples and Precision-Recall with other techniques. Figures 2c and 3c show that, for return sample counts between 100 and 1000, the suggested DMSH approach provides the highest precision with 48 bits. Based on Figs. 2a and 3b, it can be shown that our DMSH delivers significantly high precision at a low recall level, which is necessary for precision-first retrieval and is frequently utilized in real-world systems. In summary, our technique, DMSH, outperforms the compared techniques.



(a) Precision within hamming radius 2 (b) Precision Recall Curve on 48 bits



(c) Precision curve w.r.t top-N @48 bits

Fig. 3. The comparison results on the NUS-WIDE dataset under three evaluation metrics.



Fig. 4. Top 20 retrieved results from CIFAR-10 dataset by DMSH with 48-bit hash codes. The first column shows the query images, the retrieval results of DMSH are shown at other columns.

4 Conclusions

In this paper, we developed an end-to-end Deep Multi-Scale Hashing (DMSH) for large-scale image retrieval, Which generates robust hash codes by optimizing the semantic loss, similarity loss, and quantization loss. In addition, the network employs different hashing results depending on various scale features. As a result, the network would improve retrieval recall while preventing precision degradation. The experimental results on two image retrieval data sets demonstrated that our method outperforms other state-of-the-art hashing methods. The suggested model can provide robust representative features due to its scalable structure, which allows for use in many additional computer vision tasks.

References

1. Andoni, A., Indyk, P.: Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. *Commun. ACM* **51**(1), 117–122 (2008)
2. Bai, J., et al.: Loopy residual hashing: filling the quantization gap for image retrieval. *IEEE Trans. Multimedia* **22**(1), 215–228 (2019)
3. Bai, J., et al.: Deep progressive hashing for image retrieval. *IEEE Trans. Multimedia* **21**(12), 3178–3193 (2019)
4. Cakir, F., He, K., Bargal, S.A., Sclaroff, S.: Hashing with mutual information. *IEEE Trans. Pattern Anal. Mach. Intell.* **41**(10), 2424–2437 (2019)
5. 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*, pp. 1229–1237 (2018)
6. 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*, pp. 5608–5617 (2017)
7. Chen, Z., Zhou, J.: Collaborative multiview hashing. *Pattern Recogn.* **75**, 149–160 (2018)
8. 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*, pp. 1–9 (2009)
9. Cui, Y., Jiang, J., Lai, Z., Hu, Z., Wong, W.: Supervised discrete discriminant hashing for image retrieval. *Pattern Recogn.* **78**, 79–90 (2018)
10. Erin Liang, 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*, pp. 2475–2483 (2015)
11. Gionis, A., Indyk, P., Motwani, R., et al.: Similarity search in high dimensions via hashing. In: *Vldb*, vol. 99, pp. 518–529 (1999)
12. 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*, pp. 484–491 (2013)
13. 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.* **35**(12), 2916–2929 (2012)
14. Gordo, A., Almazán, J., Revaud, J., Larlus, D.: Deep image retrieval: learning global representations for image search. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) *ECCV 2016. LNCS*, vol. 9910, pp. 241–257. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46466-4_15

15. Indyk, P., Motwani, R.: Approximate nearest neighbors: towards removing the curse of dimensionality. In: Proceedings of the 30th Annual ACM Symposium on Theory of Computing, pp. 604–613 (1998)
16. Jiang, Q.Y., Li, W.J.: Scalable graph hashing with feature transformation. In: 24th International Joint Conference on Artificial Intelligence (2015)
17. Jiang, Q.Y., Li, W.J.: Asymmetric deep supervised hashing. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32 (2018)
18. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint [arXiv:1412.6980](https://arxiv.org/abs/1412.6980) (2014)
19. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)
20. Kulis, B., Darrell, T.: Learning to hash with binary reconstructive embeddings. In: Advances in Neural Information Processing Systems, vol. 22 (2009)
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, pp. 3270–3278 (2015)
22. Li, S., Chen, Z., Li, X., Lu, J., Zhou, J.: Unsupervised variational video hashing with 1d-cnn-lstm networks. *IEEE Trans. Multimedia* **22**(6), 1542–1554 (2019)
23. 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, pp. 1963–1970 (2014)
24. 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, pp. 2552–2559 (2013)
25. Lin, G., Shen, C., Wu, J.: Optimizing ranking measures for compact binary code learning. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8691, pp. 613–627. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10578-9_40
26. Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2117–2125 (2017)
27. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2980–2988 (2017)
28. Liu, W., Wang, J., Ji, R., Jiang, Y.G., Chang, S.F.: Supervised hashing with kernels. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition, pp. 2074–2081. IEEE (2012)
29. Liu, W., Wang, J., Mu, Y., Kumar, S., Chang, S.F.: Compact hyperplane hashing with bilinear functions. arXiv preprint [arXiv:1206.4618](https://arxiv.org/abs/1206.4618) (2012)
30. Russakovsky, O., et al.: Imagenet large scale visual recognition challenge. *Int. J. Comput. Vis.* **115**(3), 211–252 (2015)
31. 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, pp. 1522–1530 (2017)
32. Shen, F., Shen, C., Liu, W., Tao Shen, H.: Supervised discrete hashing. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 37–45 (2015)
33. Song, J., Gao, L., Liu, L., Zhu, X., Sebe, N.: Quantization-based hashing: a general framework for scalable image and video retrieval. *Pattern Recogn.* **75**, 175–187 (2018)

34. Strecha, C., Bronstein, A., Bronstein, M., Fua, P.: Ldhash: Improved matching with smaller descriptors. *IEEE Trans. Pattern Anal. Mach. Intell.* **34**(1), 66–78 (2011)
35. Sun, Y., Yu, S.: Deep supervised hashing with dynamic weighting scheme. In: 2020 5th IEEE International Conference on Big Data Analytics (ICBDA), pp. 57–62. IEEE (2020)
36. Wang, J., Zhang, T., Sebe, N., Shen, H.T., et al.: A survey on learning to hash. *IEEE Trans. Pattern Anal. Mach. Intell.* **40**(4), 769–790 (2017)
37. Wang, J., Kumar, S., Chang, S.F.: Semi-supervised hashing for large-scale search. *IEEE Trans. Pattern Anal. Mach. Intell.* **34**(12), 2393–2406 (2012)
38. Weiss, Y., Torralba, A., Fergus, R.: Spectral hashing. In: *Advances in Neural Information Processing Systems*, vol. 21 (2008)
39. Xia, R., Pan, Y., Lai, H., Liu, C., Yan, S.: Supervised hashing for image retrieval via image representation learning. In: 28th AAAI Conference on Artificial Intelligence (2014)
40. Yan, C., Li, Z., Zhang, Y., Liu, Y., Ji, X., Zhang, Y.: Depth image denoising using nuclear norm and learning graph model. *ACM Trans. Multimedia Comput. Commun. Appl. (TOMM)* **16**(4), 1–17 (2020)
41. Yan, C., Shao, B., Zhao, H., Ning, R., Zhang, Y., Xu, F.: 3d room layout estimation from a single RGB image. *IEEE Trans. Multimedia* **22**(11), 3014–3024 (2020)
42. Zhu, H., Long, M., Wang, J., Cao, Y.: Deep hashing network for efficient similarity retrieval. In: *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 30 (2016)