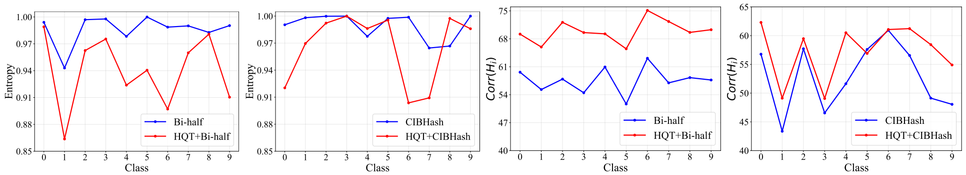
Unsupervised Hashing Network with Hyper Quantization Tree: Supplementary Material

Sungeun Kim[1]
kimsungeun@ajou.ac.kr
Jongbin Ryu*[1,2]
jongbinryu@ajou.ac.kr

[1] Department of Artificial Intelligence,
Ajou University, Republic of Korea
[2] Department of Computer Engineering,
Ajou University, Republic of Korea

1 Entropy and Correlation Analysis

We further evaluate the global and intra-class performance of entropy and correlation analysis. As shown in Fig. 1, the proposed HQT has lower class-wise entropy and higher class-wise correlation than the baseline network in most classes. On the other hand, as shown in Table 1, our HQT increases entropy and decreases correlation on the global score, which is the opposite of class-wise performance. These results indicate that the proposed HQT optimizes both correlation and entropy regarding the Fisher criteria.



(a) Entropy(Bi-half) (b) Entropy(CIBHash) (c) Correlation(Bi-half) (d) Correlation(CIBHash)

Figure 1: Class-wise performance of the intra-class entropy and correlation.

	Bi-half	HQT+Bi-half	CIBHash	HQT+CIBHash
$S(H)$	0.999	0.998	0.999	0.993
Mean$(S(H_i))$	0.986	0.940	0.989	0.966
$\Omega_S(\uparrow)$	1.013	**1.062**	1.009	**1.027**
$corr(H)$	82.34	93.92	45.63	49.14
Mean$(corr(H_i))$	57.59	69.88	52.82	57.29
$\Omega_{corr}(\downarrow)$	1.429	**1.343**	0.864	**0.857**

Table 1: Results of entropy and correlation

* Corresponding author.

2 Results on Additional Evaluation Protocol

Unsupervised hashing has been actively researched, so evaluation protocols differ across methods. Therefore, we perform additional experiments on protocols to verify that our proposed HQT is effective compared to additional methods. For a fair comparison, we used the same protocols used in previous studies[3, 7] as summarized in Table 2.

Dataset	Train	Query	Retrieval	Class	Metric
CIFAR-10(III)[3]	5,000	10,000	50,000	10	mAP@5000
CIFAR-10(IV)[7]	10,000	1,000	59,000	10	mAP@5000
NUS-WIDE(II)[3]	10,500	5,000	181,577	10	mAP@5000

Table 2: Details of the additional evaluation protocols.

We compare SOTA hashing networks such as BGAN[5], GreedyHash[6], BinGAN[8], and TBH[4] in Table 3. Also, we perform more experimental comparisons on the same evaluation protocol of UHSCM[7]. In Table 3 and 4 show that our HQT method consistently improves hashing performance on additional evaluation protocols.

Method	CIFAR-10(III)			NUS-WIDE(II)		
	16bits	32bits	64bits	16bits	32bits	64bits
BGAN	0.525	0.531	0.562	0.684	0.714	0.730
BinGAN	0.476	0.512	0.520	0.654	0.709	0.713
GreedyHash	0.448	0.473	0.501	0.633	0.691	0.731
TBH	0.532	0.573	0.578	0.717	0.725	0.735
Bi-half*	0.466	0.522	0.570	0.769	0.777	0.792
CIBHash	0.590	0.622	0.641	0.790	0.807	0.815
CIMON*	0.571	0.614	0.636	0.785	0.796	0.808
HQT+Bi-half	0.510	0.622	0.641	0.763	0.789	0.796
HQT+CIBHash	0.616	0.653	0.664	**0.796**	**0.808**	**0.817**
HQT+CIMON	**0.658**	**0.671**	**0.710**	0.785	0.807	0.815

Table 3: Experimental comparison between SOTA and our HQT on CIFAR-10(III) and NUS-WIDE(II) protocols.

Method	16bits	32bits	64bits
UHSCM*	0.851	0.853	0.833
HQT+UHSCM	**0.854**	**0.859**	**0.845**

Table 4: Comparison UHSCM CIFAR-10(IV) results

3 Computational Cost for HQT

We measure the computational cost required by HQT for training a hashing network. Table 5 demonstrates that the computational cost needed to build HQT is significantly less than the overall training time. In addition, HQT is only utilized for calculating $\mathcal{L}_{HQT}$ during the training phase, hence there is no computational cost during the inference phase. Therefore, we can conclude that the proposed HQT incurs a minimal additional computational cost during only the training phase.

Dataset		#Samples	Hashing Network			HQT
			Bihalf	CIBHash	CIMON	
CIFAR10(I)	16bits	5,000	1.00	1.60	2.01	0.02
	32bits		1.01	1.72	2.03	0.03
	64bits		1.00	1.69	2.04	0.05
CIFAR10(II)	16bits	50,000	7.62	14.80	16.40	0.18
	32bits		7.67	14.98	16.50	0.34
	64bits		7.69	14.81	16.50	0.68
Flickr25k	16bits	5,000	0.96	1.73	2.01	0.02
	32bits		0.98	1.76	2.03	0.03
	64bits		0.98	1.76	2.03	0.05
NUS-WIDE(I)	16bits	10,500	1.76	3.31	4.38	0.03
	32bits		1.77	3.36	4.39	0.05
	64bits		1.77	3.36	4.39	0.11

Table 5: Experimental results on the time complexity. The reported number for time consumption is the hour. The time consumption to generate HQT is much less compared to the training time of all hashing networks.

4 Empirical Study on Weighting Parameter

We conduct an empirical study on the value of α in Eq.3 of the manuscript. In Fig. 2, the proposed HQT works favorably on $\alpha = 0.5$ for both HQT+Bi-half and HQT+CIBHash.

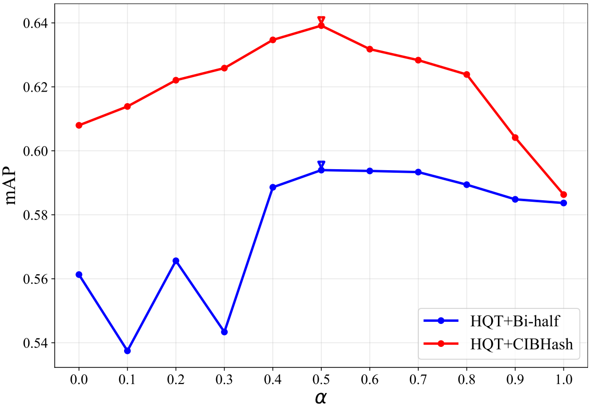


Figure 2: Experimental results on the weighting parameter α.

5 Hyper-parameter Details

We trained HQT with the same hyper-parameters for each hashing network to ensure a fair comparison. Table 6 provides implementation details for Bi-halfNet[1], CIBHash[3], CIMON[2] and UHSCM[7]. All our experiments were conducted with PyTorch v2.0 and torchvision v0.15.0, training on a NVIDIA RTX A5000.

hyperparameter	Bi-half	CIBHash	CIMON	UHSCM
backbone	VGG-16	VGG-16	VGG-16	VGG-19
data augmentation	False	True	True	True
feature length	4096	4096	4096	4096
batch size	32	64	24	128
optimizer	SGD	Adam	SGD	SGD
weight decay	5e-4	0	1e-5	1e-5
momentum	0.9	0.9	0.9	0
lr	0.0001	0.001	0.001	0.006

Table 6: Hyper-parameter settings of our implementation.

References

[1] Yunqiang Li and Jan van Gemert. Deep unsupervised image hashing by maximizing bit entropy. In *Association for the Advancement of Artificial Intelligence*, 2021.

[2] Xiao Luo, Daqing Wu, Zeyu Ma, Chong Chen, Minghua Deng, Jinwen Ma, Zhongming Jin, Jianqiang Huang, and Xian-Sheng Hua. Cimon: Towards high-quality hash codes. In *International Joint Conferenceon Artificial Intelligence*, 2021.

[3] Zexuan Qiu, Qinliang Su, Zijing Ou, Jianxing Yu, and Changyou Chen. Unsupervised hashing with contrastive information bottleneck. *International Joint Conferenceon Artificial Intelligence*, 2021.

[4] Yuming Shen, Jie Qin, Jiaxin Chen, Mengyang Yu, Li Liu, Fan Zhu, Fumin Shen, and Ling Shao. Auto-encoding twin-bottleneck hashing. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2020.

[5] Jingkuan Song, Tao He, Lianli Gao, Xing Xu, Alan Hanjalic, and Heng Tao Shen. Binary generative adversarial networks for image retrieval. In *Association for the Advancement of Artificial Intelligence*, 2018.

[6] Shupeng Su, Chao Zhang, Kai Han, and Yonghong Tian. Greedy hash: Towards fast optimization for accurate hash coding in cnn. *Neural Information Processing Systems*, 2018.

[7] Rong-Cheng Tu, Xian-Ling Mao, Kevin Qinghong Lin, Chengfei Cai, Weize Qin, Wei Wei, Hongfa Wang, and Heyan Huang. Unsupervised hashing with semantic concept mining. *Proceedings of the ACM on Management of Data*, 2023.

[8] Maciej Zieba, Piotr Semberecki, Tarek El-Gaaly, and Tomasz Trzcinski. Bingan: Learning compact binary descriptors with a regularized gan. *Advances in Neural Information Processing Systems*, 31, 2018.