Supplement Material for One Loss for Quantization: Deep Hashing with Discrete Wasserstein Distributional Matching

Khoa D. Doan, Peng Yang, Ping Li
Cognitive Computing Lab
Baidu Research
10900 NE 8th St. Bellevue, WA 98004, USA
{khoadoan106, pengyang01, pingli98}@gmail.com

This document provides additional details and experimental results to support the main submission. We begin by providing a more detailed discussion on the existing quantization approaches in Section A. Next, we discuss the detailed experimental setup and implementation of the methods in Section B and provide additional experiments on the related works in Section C. Then, we provide the formal proofs for the Theorems in the main paper in Section D. Finally, we discuss the limitations of this work in Section E.

A. Related Quantization Methods

Earlier works [5, 8, 9] avoid the discrete output and employ the tanh activation. To ensure the low-quantization error between the discrete constraint and continuous relaxation, they minimize a linear combination of heuristic objectives. It has been shown that these methods have suboptimal performance [4, 7]. CSQ [10] uses bi-modal Laplacian prior, which is defined as $L_q = E_x(|||h(x)| - \mathbf{1}||_1)$ where $\mathbf{1}$ is an all-one vector in R^m, to learn hash codes with low-quantization error. Since it is difficult to calculate the derivative of this non-smooth objective function, CSQ adopt the smooth function log cosh [10], as follows: $L_q = E_x(logcosh(|2h(x)| - \mathbf{1}))$. CSQ, however, lacks the code-balance objective. HashNet [2] approximate the discrete output with a tanh activation function, but uses a continuation technique to gradually squash the activation of the tanh activation toward -1 and 1 during training. Coding balance is also not considered in HashNet. GreedyHash [7] directly use the sign function for the discrete constraint, and during backpropagation, it uses the straight through operator, where the gradients are transmitted intactly to the front layer to avoid the vanishing gradients. DSDH [4] constrain the outputs of the last layer to be binary codes directly by using an alternating method that alternates between discrete and continuous optimizations to reduce the quantization. The primary limitations of these methods are that (i) learning on the non-smooth quantization function (i.e., sign) is relaxed for easier optimization, defeating the purpose of directly using the discrete function, and (i) coding balance is not rigorously considered.

B. Detailed Experimental Setup

To ensure a fair evaluation between the proposed quantization approach and the previous quantization methods, we utilize two well-known deep hash function architectures, VGG11 [6] and AlexNet [3]. The experiments are then conducted on several well-studied datasets for image retrieval and learning to hash domains.

- **DSDH** [4]: DSDH learns the binary codes based on Fisher's discriminant analysis by maximizing the separability between labeled data from different classes while the unlabeled data are used for regularization.
- **HashNet** [2]: HashNet balances the positive and negative pairs in the training data to trade-off between precision and recall. HashNet utilizes the pairwise labels to preserve the similarity and a continuation technique to lower the quantization error.
- **GreedyHash** [7]: GreedyHash proposes to minimize the quantization loss on the code bottleneck by ignoring the entropy of the codes. GreedyHash utilizes the point-wise classification task on the binary codes to preserve the similarity.
- **DCH** [1]: DCH designs a pairwise cross-entropy loss based on the Cauchy distribution that penalizes assignment of similar image pairs to binary codes with larger Hamming distance than a radius threshold.
- **CSQ** [10]: CSQ uses Hadamard matrix as "hash centers" then learns the binary code with binary cross entropy loss.

- **DBDH** [11]: DBDH directly outputs the binary code to further reduce the quantization error. For optimization, DBDH and uses the straight-through estimator for discrete gradient propagation.

B.1. Dataset Details

In this section, we provide the detailed description of the datasets used to evaluate the methods in our paper.

NUS-WIDE dataset contains 269,648 images, each of which belongs to at least one of the 81 concepts. We randomly select 5,000 images for the query set, with the remaining images used as the retrieval set. 10,000 images randomly selected in the retrieval set are used for training.

COCO dataset contains 123,287 images, labeled with at least 1 out of 80 semantic concepts. Similarly, the query set is randomly constructed with 5,000 images, with the remaining images used as the retrieval set (10,000 randomly selected images in this set are used for training).

CIFAR-10 contains 60,000 images organized into 10 semantic classes, out of which 1,000 images are randomly selected as the query set, and the remaining images used as the retrieval set. Similarly, 5,000 images randomly selected from the retrieval set are used for training.

B.2. Implementation Details

For each method, we uses either AlexNet or Vgg11 as the backbone (i.e., the hash function). We modify the hashing methods in our experiments by replacing their original quantization losses with the proposed quantization approaches. The code of the experiments is provided along with this document as parts of the Supplementary Materials.

C. Additional Experiments

C.1. Additional retrieval comparisons

We report additional mAP results when learning the 128-bit hash functions with VGG11 backbone [6] in Table 4. Additional results on AlexNet backbone [3] are also reported in Tables 5 and 6. We can observe that the proposed quantization achieves better performance, in terms of mAP, compared to the original quantization approaches, as shown in Table 5. We also observe a similar result for Precision@1000 which is another common metric used to evaluate hashing methods, as in Table 6,

D. Proofs

Theorem 1. *The proposed distance Sliced Wasserstein calculation in Theorem 1, denoted as HSWD, is a valid distance function of probability measures in this space.*

Table 4. Mean Average Precision (mAP) for learning the 128-bit hash functions on the three image datasets. The blue value (in bold) along the mAP value of each of the proposed approaches shows the relative improvement over the original algorithm, while the italicized value indicates no improvement.

Method	CIFAR-10	NUS-WIDE	COCO
DSDH [4]	0.8356	0.8704	0.8240
DSDH-S	0.8544/**2.3%**	0.8710/**0.1%**	0.8480/**2.9%**
DSDH-C	0.8685/**3.9%**	0.8864/**1.8%**	0.8480/**2.9%**
HashNet [2]	0.8590	0.8758	0.8292
HashNet-S	0.8730/**1.6%**	0.8780/**0.3%**	0.8388/**1.1%**
HashNet-C	0.8769/**2.1%**	0.8880/**1.4%**	0.8342/**0.6%**
GreedyHash [7]	0.8625	0.8409	0.7832
GreedyHash-S	0.8685/**0.7%**	0.8439/**0.4%**	0.7901/**0.9%**
GreedyHash-C	0.8726/**1.2%**	0.8510/**1.2%**	0.7957/**1.6%**
DCH [1]	0.8480	0.7937	0.7667
DCH-S	0.8521/**0.5%**	0.8016/**1.0%**	0.7691/**0.3%**
DCH-C	0.8599/**1.4%**	0.8126/**2.4%**	0.7790/**1.6%**
CSQ [10]	0.6783	0.7550	0.8146
CSQ-S	0.8401/**4.1%**	0.8555/**3.2%**	0.8554/**2.3%**
CSQ-C	0.8457/**4.8%**	0.8558/**3.2%**	0.8652/**3.4%**
DBDH [11]	0.8553	0.8641	0.8122
DBDH-S	0.8743/**2.2%**	0.8800/**1.8%**	0.8470/**4.3%**
DBDH-C	0.8702/**1.7%**	0.8738/**1.1%**	0.8435/**3.6%**

Proof. We first prove that HSWD satisfies the triangle inequality. Let I_i be column i of the Identity Matrix $I \in R^{m\times m}$. We have:

$$\mathcal{D}(h(X), B) \approx \left(\frac{1}{m}\sum_{l=1}^{m}[\mathcal{W}(h(X)_{l,:}, B_{l,:})]^2\right)^{1/2} \tag{1}$$

$$= \left(\frac{1}{m}\sum_{l=1}^{m}[\mathcal{W}(I_i^T h(X), I_i^T B_{l,:})]^2\right)^{1/2} \tag{2}$$

Since this is equivalent to SWD where there the projections are I_i for $i = 1, ..., m$, HSWD is a valid distance. □

E. Limitations

This paper presented a quantization approach to improve the retrieval performance of existing deep supervised hash-learning methods. Our study mainly targets the deep supervised image hashing domain. Our work can be applied along with existing methods in this domain to enhance the overall performance. However, we believe there is a large room for the applications of the proposed approach in a more general settings, e.g., unsupervised image hashing or discrete-optimization problems.

Table 5. Mean Average Precision (mAP) for different numbers of bits on the three image datasets. The blue value (in bold) along the mAP value of each of the proposed approaches shows the relative improvement over the original algorithm, while the italicized value indicates no improvement

Method	CIFAR-10			NUS-WIDE		
	16 bits	32 bits	64 bits	16 bits	32 bits	64 bits
DSDH	0.7366	0.7655	0.7728	0.8080	0.8248	0.8370
DSDH-S	0.7886/**7.1%**	0.8031/**4.9%**	0.8128/**5.2%**	0.8111/**0.4%**	0.8247/**0.0%**	0.8371/**0.0%**
DSDH-C	0.7995/**8.6%**	0.8133/**6.2%**	0.8190/**6.0%**	0.8156/**0.9%**	0.8346/**1.2%**	0.8436/**0.8%**
HashNet	0.6175	0.8016	0.8184	0.7578	0.8134	0.8469
HashNet-S	0.7762/**25.7%**	0.8175/**2.0%**	0.8304/**1.5%**	0.7722/**1.9%**	0.8176/**0.5%**	0.8437/*-0.4%*
HashNet-C	0.7140/**15.6%**	0.8092/**1.0%**	0.8300/**1.4%**	0.7464/*-1.5%*	0.8068/*-0.8%*	0.8412/*-0.7%*
GreedyHash	0.7888	0.8002	0.8258	0.7535	0.7930	0.8097
GreedyHash-S	0.7841/*-0.6%*	0.8175/**2.2%**	0.8256/**0.0%**	0.7575/**0.5%**	0.7893/*-0.5%*	0.8083/*-0.2%*
GreedyHash-C	0.7902/**0.2%**	0.8174/**2.1%**	0.8217/*-0.5%*	0.7583/**0.6%**	0.7894/*-0.4%*	0.8158/**0.7%**
DCH	0.7830	0.8063	0.7990	0.7819	0.7866	0.7918
DCH-S	0.7958/**1.6%**	0.8055/*-0.1%*	0.7939/*-0.6%*	0.7807/*-0.1%*	0.7908/**0.5%**	0.7860/*-0.7%*
DCH-C	0.8010/**2.3%**	0.8055/*-0.1%*	0.7986/**0.0%**	0.7878/**0.8%**	0.7886/**0.3%**	0.7843/*-0.9%*
CSQ	0.7840	0.7976	0.7992	0.7813	0.8202	0.8366
CSQ-S	0.8035/**2.5%**	0.8105/**1.6%**	0.8099/**1.3%**	0.7958/**1.9%**	0.8227/**0.3%**	0.8363/**0.0%**
CSQ-C	0.8017/**2.3%**	0.8109/**1.7%**	0.8034/**0.5%**	0.7937/**1.6%**	0.8274/**0.9%**	0.8377/**0.1%**
DBDH	0.7617	0.7731	0.7864	0.8101	0.8278	0.8377
DBDH-S	0.8024/**5.3%**	0.8073/**4.4%**	0.8119/**3.2%**	0.8153/**0.6%**	0.8342/**0.8%**	0.8424/**0.6%**
DBDH-C	0.8030/**5.4%**	0.8122/**5.0%**	0.8064/**2.5%**	0.8109/**0.1%**	0.8338/**0.7%**	0.8435/**0.7%**

Table 6. Precision@1000 for different numbers of bits on the three image datasets. The blue value (in bold) along the mAP value of each of the proposed approaches shows the relative improvement over the original algorithm, while the italicized value indicates no improvement

Method	CIFAR-10			NUS-WIDE		
	16 bits	32 bits	64 bits	16 bits	32 bits	64 bits
DSDH	0.8252	0.8406	0.8396	0.8117	0.8294	0.8425
DSDH-S	0.8526/**3.3%**	0.8543/**1.6%**	0.8644/**2.9%**	0.8162/**0.6%**	0.8312/**0.2%**	0.8446/**0.2%**
DSDH-C	0.8645/**4.8%**	0.8739/**4.0%**	0.8811/**4.9%**	0.8195/**1.0%**	0.8391/**1.2%**	0.8487/**0.7%**
HashNet	0.6193	0.8613	0.8711	0.7581	0.8158	0.8524
HashNet-S	0.8470/**36.8%**	0.8755/**1.7%**	0.8804/**1.1%**	0.7743/**2.1%**	0.8199/**0.5%**	0.8491/*-0.4%*
HashNet-C	0.7698/**24.3%**	0.8715/**1.2%**	0.8719/**0.1%**	0.7456/*-1.7%*	0.8078/*-1.0%*	0.8456/*-0.8%*
GreedyHash	0.8561	0.8616	0.8701	0.7601	0.8009	0.8198
GreedyHash-S	0.8583/**0.3%**	0.8656/**0.5%**	0.8695/*-0.1%*	0.7657/**0.7%**	0.7973/*-0.5%*	0.8180/*-0.2%*
GreedyHash-C	0.8517/*-0.5%*	0.8700/**1.0%**	0.8652/*-0.6%*	0.7630/**0.4%**	0.7931/*-1.0%*	0.8200/**0.0%**
DCH	0.8621	0.8568	0.8639	0.7843	0.7898	0.7925
DCH-S	0.8622/**0.0%**	0.8761/**2.3%**	0.8730/**1.1%**	0.7846/**0.0%**	0.7923/**0.3%**	0.7887/*-0.5%*
DCH-C	0.8654/**0.4%**	0.8635/**0.8%**	0.8606/*-0.4%*	0.7893/**0.6%**	0.7914/**0.2%**	0.7868/*-0.7%*
CSQ	0.8510	0.8571	0.8619	0.7903	0.8285	0.8446
CSQ-S	0.8661/**1.8%**	0.8732/**1.9%**	0.8667/**0.6%**	0.8034/**1.7%**	0.8318/**0.4%**	0.8442/*-0.1%*
CSQ-C	0.8670/**1.9%**	0.8688/**1.4%**	0.8619/**0.0%**	0.8007/**1.3%**	0.8353/**0.8%**	0.8455/**0.1%**
DBDH	0.8440	0.8421	0.8488	0.8122	0.8323	0.8435
DBDH-S	0.8626/**2.2%**	0.8675/**3.0%**	0.8732/**2.9%**	0.8177/**0.7%**	0.8388/**0.8%**	0.8486/**0.6%**
DBDH-C	0.8658/**2.6%**	0.8731/**3.7%**	0.8655/**2.0%**	0.8135/**0.1%**	0.8380/**0.7%**	0.8490/**0.7%**

References

[1] Yue Cao, Mingsheng Long, Bin Liu, and Jianmin Wang. Deep cauchy hashing for hamming space retrieval. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1229–1237, Salt Lake City, UT, 2018. 1, 2

[2] Zhangjie Cao, Mingsheng Long, Jianmin Wang, and Philip S Yu. Hashnet: Deep learning to hash by continuation. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 5608–5617, Venice, Italy, 2017. 1, 2

[3] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1106–1114, Lake Tahoe, NV, 2012. 1, 2

[4] Qi Li, Zhenan Sun, Ran He, and Tieniu Tan. Deep supervised discrete hashing. In *Neural Information Processing Systems (NIPS)*, pages 2479–2488, Long Beach, CA, 2017. 1, 2

[5] Fumin Shen, Chunhua Shen, Wei Liu, and Heng Tao Shen. Supervised discrete hashing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 37–45, Boston, MA, 2015. 1

[6] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *Proceedings of the 3rd International Conference on Learning*

Representations (ICLR), San Diego, CA, 2015. 1, 2

[7] Shupeng Su, Chao Zhang, Kai Han, and Yonghong Tian. Greedy hash: Towards fast optimization for accurate hash coding in CNN. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 806–815, Montréal, Canada, 2018. 1, 2

[8] Rongkai Xia, Yan Pan, Hanjiang Lai, Cong Liu, and Shuicheng Yan. Supervised hashing for image retrieval via image representation learning. In Carla E. Brodley and Peter Stone, editors, *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence (AAAI)*, pages 2156–2162, Québec City, Canada, 2014. 1

[9] Huei-Fang Yang, Kevin Lin, and Chu-Song Chen. Supervised learning of semantics-preserving hash via deep convolutional neural networks. *IEEE Trans. Pattern Anal. Mach. Intell.*, 40(2):437–451, 2018. 1

[10] Li Yuan, Tao Wang, Xiaopeng Zhang, Francis EH Tay, Zequn Jie, Wei Liu, and Jiashi Feng. Central similarity quantization for efficient image and video retrieval. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3083–3092, Seattle, WA, 2020. 1, 2

[11] Xiangtao Zheng, Yichao Zhang, and Xiaoqiang Lu. Deep balanced discrete hashing for image retrieval. *Neurocomputing*, 403:224–236, 2020. 2