DEEP HASHING WITH HASH CENTER UPDATE FOR EFFICIENT IMAGE RETRIEVAL

Abin Jose, Daniel Filbert, Christian Rohlfing, Jens-Rainer Ohm

Institut für Nachrichtentechnik, RWTH Aachen University, Aachen, Germany
jose@ient.rwth-aachen.de

ABSTRACT

In this paper, we propose an approach for learning binary hash codes for image retrieval. Canonical Correlation Analysis (CCA) is used to design two loss functions for training a neural network such that the correlation between the two views to CCA is maximum. The main motivation for using CCA for feature space learning is that dimensionality reduction is possible and short binary codes could be generated. The first loss maximizes the correlation between the hash centers and the learned hash codes. The second loss maximizes the correlation between the class labels and the classification scores. In this paper, a novel weighted mean and thresholding-based hash center update scheme for adapting the hash centers is proposed. The training loss reaches the theoretical lower bound of the proposed loss functions, showing that the correlation coefficients are maximized during training and substantiating the formation of efficient feature space for retrieval. The measured mean average precision shows that the proposed approach outperforms other state-of-the-art methods.

Index Terms— Binary hashing, Hash center update, Canonical Correlation Analysis, Image Retrieval.

1. INTRODUCTION

Nowadays, deep neural networks have become state-of-the-art in feature extraction and are the basis of most Content-Based Image Retrieval methods [1]. Besides high retrieval quality, both efficiency and speed are essential requirements for a good retrieval system. Due to the dramatic and continuous growth of image datasets, evaluation of Euclidean distances for subsequent ranking and nearest neighbor search has become computationally expensive [2]. A solution for this problem would be using binary codes.

To overcome the challenges of high computational cost and speed, Approximate Nearest Neighbour [3] (ANN) search is a common alternative approach that offers more efficiency and accuracy [4]. A widely used form of ANN search is hashing. Hashing is in general classified into data-independent approaches and data-dependent approaches. Data independent approaches [5], [6], [7] mainly rely upon random projections to generate the hash functions. Locality-sensitive hashing (LSH) is [8] a popular data-independent approach, which was used in [9] and [10] to solve the ANN problem while avoiding the curse of dimensionality. Recent methods mainly concentrate on data-dependent approaches which are mainly categorized as unsupervised and supervised methods. We refer to [11] for an extensive survey. For learning hash functions, unsupervised approaches [12], [13], [14] use various metrics to supervise the learning. In contrast, the supervised approaches utilize semantic labels. In recent years, deep supervised hashing has achieved good retrieval performance [15], [16], [17], especially pair-wise and triplet approaches [18], [16]. However, the loss function depends on the distance calculation between similar and dissimilar pairs and does not utilize the entire feature statistics.

Recently, Canonical Correlation Analysis (CCA) was used such that the correlation between the feature vectors and label vectors was maximized [19]. The learned feature space was then binarized using the ITQ [20] approach. Here, statistical properties of the feature space are considered. Yuan et al. proposed a new global similarity metric which is called central similarity in [21]. Here, hash values of similar data points are encouraged to approach a common center, whereas pairs of dissimilar hash codes converge to distinct centers. There are two systematic approaches proposed in [21] to generate hash centers fulfilling the above condition: One leverages the characteristics of the Hadamard matrix, thus obtaining hash centers with maximal mutual Hamming distance, and the other uses random sampling from a Bernoulli distribution when the bit length is not a power of 2. There are as many hash centers generated as there are semantic labels. However, since each sample can contain one or more categories for images, a majority voting is proposed to account for the transitive similarity of data points sharing multiple labels. However, a major problem in this approach is that the hash centers are not updated. Another interesting work in a similar direction was proposed by Hong et al. ins [22]. Here, Linear Discriminant Analysis characteristics are trained directly on the hash codes. The proposed method updates the hash centers during training.

Inspired by the idea of updating hash centers [22], an alternative method based on the weighted mean of the hash values is proposed, which aims to reflect the movement of the formed clusters in the Hamming space. The network was trained using a CCA-based loss formulation such that the correlation between the hash codes and hash centers is maximized along with the correlation of classification scores and class labels.

The major contributions of this paper are: 1) Initial hash centers, are selected around which the hash codes are clustered [21]. A novel weighted mean and thresholding-based hash center update scheme is proposed. 2) The loss function is formulated using CCA such that the generated hash codes and hash centers have maximum correlation. This loss function is combined with the CCA-based classification loss [19] which maximizes the correlation of classification scores with the class labels. 3) The theoretical lower bound is determined for both loss functions based on the rank of the two views of CCA, and an optimum regularization factor is chosen. Experiments were conducted on the datasets MS-COCO [23] and NUS-WIDE [24]. The training and test curves and t-SNE [25] plots of the generated feature space were plotted. Mean average precision was computed and the performance is compared with other supervised approaches such as DPSH [15], DCCH [19], CSQ [21], DSDH [26], DDSH [17], DTSH [27], LDH [22], HashNet [3], DHN [28], DNNH [29], and CNNH [30].

The paper is organized as follows: Section 2 explains the net-

work architecture, and loss function. The novel hash center update is detailed in Section 3. Experimental results are discussed in Section 4 and concluding remarks are drawn in Section 5.

2. PROPOSED APPROACH

Network architecture: Deep hashing is used in the proposed Deep Central Similarity Hashing (DCSH) method. The network architecture of the proposed approach is given in Fig. 1. The residual network [31] is used as the basic feature extractor which was pre-trained on ImageNet [32]. Followed by the residual layers, a hashing layer consisting of a fully connected layer and subsequent sigmoid activation function is used to generate hashes. The output dimension of this layer is therefore equal to the required number of bits. An intermediate layer is used subsequently to the hashing layer to generate high output dimensionality from the input hash codes. This intermediate layer is required, since the loss function used in this architecture, L_{DCSH}, is a dimensionality reduction method. A higher number of input dimensions exceeding the number of distinct classes in the dataset is required as input to this layer. There are two losses: 1) L_{hash} correlating the hashing outputs with the semantic hash centers. 2) L_{class} correlating the classification scores with the label information of the dataset. The final training loss, L_{DCSH} then takes both losses into account for the optimization.

Hash code generation: In the proposed approach, hash values are generated by the output of the hashing layer. It contains a sigmoid activation function which produces hashes $\vec{x}_h \in [0,1]^B$ with B elements, and B being the number of required bits for each of the N images. After successful training of the network, the values of the hashing outputs are likely to be close to 0 or 1. An additional thresholding $\tau(\cdot)$ at 0.5 is performed on each element to obtain the desired binary codes.

Loss formulation: The proposed approach uses Canonical Correlation Analysis (CCA) at two output layers of the network to formulate the training loss (Fig. 1). CCA aims to find transformations of two input views that maximally correlate their mapped representations. DCCH [19] uses CCA such that a neural network can be trained to generate non-linear mappings of its input which maximally correlate to a given target. This loss formulation is used to optimize the outputs of a neural network from two layers. The layers from which the loss is calculated are shown in Fig. 1. For two data views $\mathbf{X}$ and $\mathbf{Y}$, CCA optimizes projections $\vec{a}$ and $\vec{b}$ which maximize the correlation ρ between the projected inputs as:

$$\rho(\vec{a}^*, \vec{b}^*) = \max_{\vec{a},\vec{b}} \frac{\vec{a}^T \mathbf{\Sigma_{XY}} \vec{b}}{\sqrt{\vec{a}^T \mathbf{\Sigma_{XX}} \vec{a} \quad \vec{b}^T \mathbf{\Sigma_{YY}} \vec{b}}} \tag{1}$$

s.t. $\vec{a}^T \mathbf{\Sigma_{XX}} \vec{a} = \vec{b}^T \mathbf{\Sigma_{YY}} \vec{b} = 1$, where $\mathbf{\Sigma_{XX}}, \mathbf{\Sigma_{XY}}$, and $\mathbf{\Sigma_{YY}}$ denotes the covariances, $\mathrm{cov}(\mathbf{X},\mathbf{X}), \mathrm{cov}(\mathbf{X},\mathbf{Y})$, and $\mathrm{cov}(\mathbf{Y},\mathbf{Y})$ respectively. This optimization problem can be solved by using a Singular Value Decomposition, shown by Mardia et al. in [33]. The CCA based loss function is formulated as:

$$L_{\mathrm{DCSH}} = -\sum_{i=1}^{k} \sigma_i = -\sum_{i=1}^{k} \rho_i, \tag{2}$$

with k largest singular values σ_i of the matrix $\mathbf{K}$ which is defined as $\mathbf{K} := \mathbf{\Sigma_{XX}}^{-1/2} \mathbf{\Sigma_{XY}} \mathbf{\Sigma_{YY}}^{-1/2}$. The equivalence between the correlation coefficients and the singular values is also proven in [33]. The training loss minimizes the negative sum of correlation coefficients. Since the maximum positive correlation can reach a value of 1, the lower bound of the loss will be $-k$. Furthermore, k can only be as high as the rank of matrix $\mathbf{K}$. For input vectors $\mathbf{X}$ and target data view $\mathbf{Y}$, the upper bound of k is:

$$k_{\max} = \mathrm{rank}(\mathbf{K}) = \min(\mathrm{rank}(\mathbf{X}), \mathrm{rank}(\mathbf{Y})) - 1 \, . \tag{3}$$

The subtraction of 1 from either $\mathrm{rank}(\mathbf{X})$ or $\mathrm{rank}(\mathbf{Y})$ is due to the inherent subtraction of the mean for the covariance matrices. The loss function of DCSH consists of two CCA evaluations: the hashing and the classification losses which are detailed in the following.

Hashing loss: First, the DCSH loss is evaluated by using the outputs of the hashing layer $\mathbf{X}_{\mathrm{h}}$ with their corresponding hash centers $\mathbf{Y}_{\mathrm{h}}$. These hash centers act as targets in Hamming space towards which the respective output hashes of the network should converge. The size of the two data views $\mathbf{X}_{\mathrm{h}} \in [0,1]^{M \times B}$ and $\mathbf{Y}_{\mathrm{h}} \in \{0,1\}^{M \times B}$ is determined by the batch size M and the number of bits B in the binary codes. Therefore, the hashing loss can be formulated as:

$$L_{\mathrm{hash}} = L_{\mathrm{DCSH}}(\mathbf{X}_{\mathrm{h}}, \mathbf{Y}_{\mathrm{h}}) \, . \tag{4}$$

Since each hash center, used in the target view $\mathbf{Y}_{\mathrm{h}}$, represents one of the C categories in the underlying dataset, it consists of only at most C distinct rows. Assuming that $M > C$, $\mathrm{rank}(\mathbf{Y}_{\mathrm{h}}) = \min(B, C)$. Therefore, the maximal number of correlation coefficients according to Eq. (3) is:

$$\begin{aligned} k_{\max} &= \min(\min(B, M), \min(B, C)) - 1 \\ &= \min(B, C) - 1 \text{ , for } M > B \text{ and } M > C. \end{aligned} \tag{5}$$

According to (2), this effectively means that during training the negative sum of $\min(B, C) - 1$ correlation coefficients is minimized. As the correlation maximally can reach a value of 1, the lower bound of the hashing loss goes to $-(\min(B, C) - 1)$.

Classification loss: The second component of the proposed DCSH loss function is a classification loss, which performs a CCA of the classification scores $\mathbf{X}_{\mathrm{c}}$ from the output of the final network layer and a target group indicator matrix $\mathbf{Y}_{\mathrm{c}}$ (see Fig. 1). In this context, the first data view $\mathbf{X}_{\mathrm{c}} \in \mathbb{R}^{M \times L}$ is defined as a matrix consisting of M rows each denoting an L-dimensional feature representation $\vec{x}^T \in \mathbb{R}^L$ of a sample of the current batch with M images. As a second data view, a so-called group indicator matrix $\mathbf{Y}_{\mathrm{c}} \in \{0,1\}^{M \times C}$ is used with M rows of label vectors $\vec{y} = [y_1, ..., y_C]$ with C entries (with a value 1 indicating the classes associated to the respective images and a value of 0 otherwise). Here, the size of the two used data views $\mathbf{X}_{\mathrm{c}} \in [0,1]^{M \times C}$ and $\mathbf{Y}_{\mathrm{c}} \in \{0,1\}^{M \times C}$ is determined by the batch size M and the number of distinct classes C in the underlying dataset. Given these views, the classification loss is:

$$L_{\mathrm{class}} = L_{\mathrm{DCSH}}(\mathbf{X}_{\mathrm{c}}, \mathbf{Y}_{\mathrm{c}}) \, . \tag{6}$$

Assuming a higher batch size than distinct classes $M > C$, the maximal number of correlation coefficients according to (3) results in:

$$k_{\max} = \min(M, C) - 1 = C - 1, \text{for } M > C. \tag{7}$$

The lower bound of the classification loss then comes to a value of $-(C-1)$.

Loss combination and normalization: The final training objective of the proposed deep hashing method is designed by a linear combination of the two losses as, $L_{\mathrm{DCSH_{combined}}} = L_{\mathrm{hash}} + \alpha L_{\mathrm{class}}$. The regularization factor α balances the contribution of both components for the optimization objective. By using the theoretical lower bounds of hashing and classification loss as shown in Eq. (5) and Eq. (7), an equal weighting of their contributions can be computed as, $\alpha = \frac{\min(B,C)-1}{C-1}$. Setting α to this value results

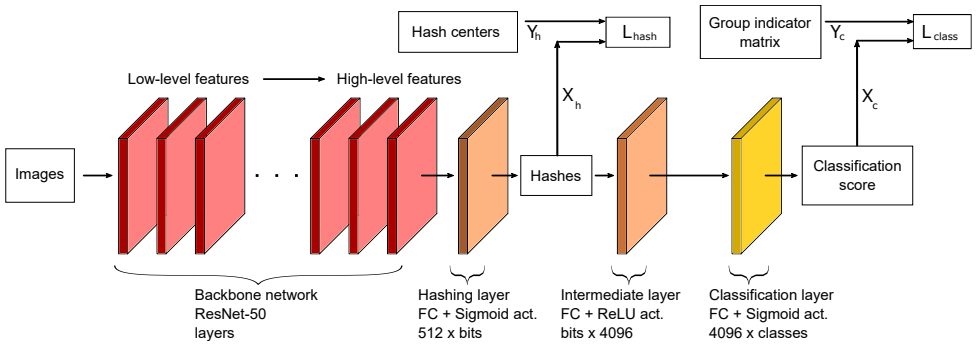


Fig. 1: Overview of the proposed Deep Central Similarity Hashing network architecture. ResNet layers according to [31] is used as the backbone network for basic feature extraction. Both hashing and classification layer, consist of a fully connected (FC) layer with subsequent sigmoid activation. The intermediate layer comprises a fully connected layer with subsequent ReLU activation. Bits indicate the bit length of the hash code. Classes indicate the number of categories in the dataset.

in the theoretical lower bounds of the hashing and classification loss becoming equal. However, it has been beneficial in subsequent experiments to choose α in each case of B and C as, $\alpha = \frac{B-1}{C-1}$. Therefore, in case of datasets for which $C < B$, the classification loss is emphasized. Experiments on multi-labeled datasets all satisfy $C > B$ and thus, α normalizes the contribution of classification and hashing loss in the final training objective. The theoretical lower bound of the combined DCSH loss can then be determined as:

$$\begin{aligned} \min L_{\mathrm{DCSH_{combined}}} &= \min L_{\mathrm{hash}} + \alpha \min L_{\mathrm{class}} \\ &= -(\min(B, C) - 1) - (B - 1)\,. \end{aligned} \tag{8}$$

3. HASH CENTER UPDATE DURING TRAINING

An important part of the training procedure in DCSH is to generate semantic hash centers, which are updated in each epoch. According to [21], either the Hadamard matrix, in case of required bit length of a power of 2, or a Bernoulli distribution $\mathrm{Bern}(0.5)$, for other bit lengths is used to generate the initial hash centers. Both approaches provide hash centers reflecting distinct classes which are sufficiently far apart with respect to the Hamming distance. Each hash center $\vec{h}_c^{(0)}, c \in \{1, ..., C\}$ represents one of the C classes in the dataset. The index 0 represents the initial state of the hash centers. These hash centers $\vec{h}_c^{(0)}$, are elements of $\mathbf{Y_h}$, which is one view of CCA loss L_{hash}. Opposed to [21] which keeps the hash centers constant, our approach additionally performs an update after each training epoch with the goal to better reflect semantic information. Thus, the updated versions are able to better represent the class centers more dynamically. Using this approach adapts the hash centers to the semantic information. Since a sigmoid activation is used in the hashing layer, the resulting hash values $\vec{x_h}$ will be in the range between $[0, 1]$. These $\vec{x_h}$ are elements of $\mathbf{X_h}$ which is the second view of the CCA loss L_{hash}. The following proposed methods for updating the hash centers require them to be in the range of $[-1, 1]$. Therefore, we apply a simple mapping, $f(\vec{x_h}) = 2\vec{x_h} - 1$. However, for valid further training the updated hash centers have to be again binary values $\{0, 1\}$. Therefore, a final remapping to the required binary space is performed as well.

In our approach, the hash values can be associated with multiple classes at the same time. Therefore, the hash values are weighted lighter when containing more categories in their labels. This is based on the assumption that with increasing category associations, hash values are less able to represent a single category. Therefore, a weighting factor $w_{\mathrm{n}} = \frac{1}{|l_{\mathrm{n}}|}$ for each of the N hash values is introduced based on the number of classes $|l_{\mathrm{n}}|$ in its corresponding label l_{n}. First, the hashes of the training images are evaluated by a forward pass after weight update of the network in the current epoch. Then, the obtained hash values are grouped in sets $\mathcal{G}_c^{(i)}$, each consisting of those hashes which at least contain the respective class c in their corresponding label. Note, that for multi-labeled data the resulting hash value groups $\mathcal{G}_c^{(i)}$ are not distinct sets. This is because hash values labeled with multiple categories c at the same time are also contained in each corresponding group $\mathcal{G}_c^{(i)}$. Now, the weights w_{n} are used to compute a weighted mean of the hash values contained in each group $\mathcal{G}_c^{(i)}$ at epoch i:

$$\tilde{\vec{h}}_c^{(i+1)} = \frac{1}{|\mathcal{G}_c^{(i)}|} \sum_{\vec{x_h}^{(i)} \in \mathcal{G}_c^{(i)}} w_{\mathrm{n}}\, \vec{x_h}^{(i)}\,. \tag{9}$$

By applying a weighted mean, the contribution of each hash value is adapted to its reflecting semantic information about the class of a group. The steps to perform a hash center update during DCSH training are summarized in Algorithm below. A subsequent sign based thresholding of the obtained mean vector $\tilde{\vec{h}}_c^{(i+1)}$ results in the updated binary hash center $\vec{h}_c^{(i+1)}$.

Algorithm DCSH hash center update

Require: Hash centers $\mathcal{H}^{(i)} = \{\vec{h}_c^{(i)}\}, c = 1, ..., C$ with C classes, and output hashes $\{\vec{x_h}^{(i)}\} \in [0; 1]^n$ with $n = 1, ..., N$ from N training images at epoch i.

1: For each class c, group all hash outputs associated with c in their label l_{n}:
$\mathcal{G}_c^{(i)} = \{\vec{x_h}^{(i)} : c \text{ in } l_{\mathrm{n}}\}$, for $c = 1, ..., C$.

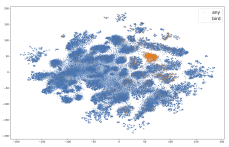
2: Calculate weights for each output hash based on number of classes in its label $|l_{\mathrm{n}}|$:
$w_{\mathrm{n}} = \frac{1}{|l_{\mathrm{n}}|}$.

3: Calculate weighted mean of grouped hashing values:
$\tilde{\vec{h}}_c^{(i+1)} = \frac{1}{|\mathcal{G}_c^{(i)}|} \sum_{\vec{x_h}^{(i)} \in \mathcal{G}_c^{(i)}} w_{\mathrm{n}}\, \vec{x_h}^{(i)}$, for $c = 1, ..., C$.

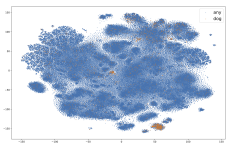
4: Create updated binary hash centers by thresholding:
$\vec{h}_c^{(i+1)} = \mathrm{sign}(\tilde{\vec{h}}_c^{(i+1)})$.

5: **return** Updated hash centers $\mathcal{H}^{(i+1)}$ for epoch $i + 1$.

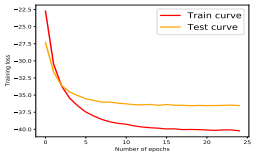
Fig. 2: Algorithm: DCSH hash center update.



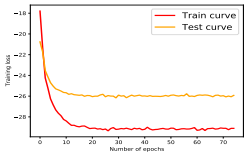
(a) MS-COCO bird class in orange



(b) NUS-WIDE dog class in orange



(c) MS-COCO loss curve



(d) NUS-WIDE loss curve

Fig. 3: T-SNE of 64 dimensional features for a) MS-COCO highlighting bird category, and b) NUS-WIDE highlighting dog category. The training and test loss for 32 bit case is shown. The lower bound of -62 could not be reached as the correlation coefficients will never reach the maximum value.

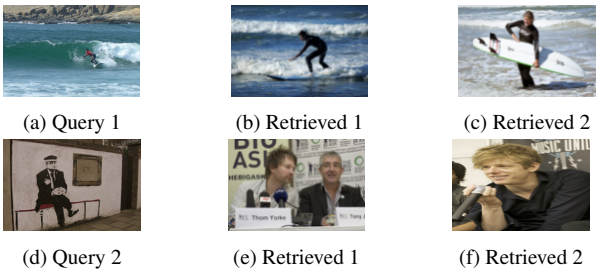
MS-COCO					NUS-WIDE				
Method	16 bits	32 bits	48 bits	64 bits	Method	12 bits	24 bits	32 bits	48 bits
DCSH (ours)	**0.805**	**0.847**	**0.859**	**0.861**	**DCSH (Ours)**	**0.823**	**0.833**	**0.841**	**0.857**
CSQ [21]	0.796	0.838	-	**0.861**	DPSH [15]	0.794	0.822	0.838	0.851
DCCH [19]	0.659	0.729	0.731	0.739	DCCH [19]	0.782	0.814	0.825	0.834
HashNet [3]	0.687	0.718	0.730	0.736	CSQ [21]	-	-	0.825	-
DHN [28]	0.677	0.701	0.695	0.694	DSDH [26]	0.776	0.808	0.820	0.829
DNNH [29]	0.593	0.603	0.604	0.610	DDSH [17]	0.791	0.815	0.821	0.827
CNNH [30]	0.564	0.574	0.571	0.567	DTSH [27]	0.773	0.808	0.812	0.814

Table 1: MAP on MS-COCO and NUS-WIDE for different approaches.

4. EXPERIMENTAL RESULTS

Two multi-labeled datasets, were used for evaluating the final retrieval performance of DCSH. The training and test curves were plotted for each dataset. Furthermore, the retrieval performance was measured by calculating mean average precision (MAP) [17]. The neural network architecture used is ResNet-50 [31]. We used a batch size of 200 in our experiments, with an initial learning rate and learning rate decay of 0.0008 and 0.1 every 10^{th} epoch which was determined by a vast grid search. Stochastic gradient descent is used as the optimizer.

MS-COCO [23] is associated with several of 80 distinct object classes. The data split for image retrieval evaluation is based on the setup described in HashNet [3]. The model was trained for 25 epochs. The training and test loss values during training for 32 bits is given in Fig. 3c. Theoretically, the lower bound of the loss function is $-2(B-1)$ with B number of bits, as discussed in (8). However, it can be seen in Fig. 3c that this lower bound is not reached. This is because a multi-labeled dataset cannot be fully class-wise clustered as each image may belong to several classes. Therefore, the correlation coefficients of the underlying loss are not able to reach the maximum. The training reaches a lower bound indicating formation of optimum feature space. Since MS-COCO is a multi-labeled dataset, it is not feasible to depict all class associations for the data points at once by using different colors. Instead, single categories have been selected to be highlighted against the remaining categories in Fig. 3a. There are cases where these clusters represent a unique category, as clearly shown in the example of 'bird' depicted in Fig. 3a. The retrieval performance of DCSH for MS-COCO in terms of MAP is given in Table 1. Results are compared to state-of-the-art approaches. For all the bit lengths, the MAP value outperforms previous approaches. The top 2 retrieval results for a query image with labels 'surfboard' and 'person' are shown in Figs. 4a,4b, and 4c.



(a) Query 1 (b) Retrieved 1 (c) Retrieved 2

(d) Query 2 (e) Retrieved 1 (f) Retrieved 2

Fig. 4: Query image and top two retrieval results for MS-COCO ('surfboard', 'person' category - first row) and NUS-WIDE ('person' category - second row).

NUS-WIDE was introduced by Chua et al. in [24] in which each image is associated with one or more of 81 concepts. The experimental setup for NUS-WIDE in DDSH [17] is used here. The network was trained for 75 epochs. Fig. 3d shows the training and test loss during training for 32 bits. The theoretical lower-bound in this case is $-2(B-1)$ as well, and cannot be reached. In Fig. 3b, orange color indicates the 'dog' category. It can be seen that there is clearly a dominant area in the feature space and, a nearest neighbor search would result in retrieved images very likely being associated with the similar semantics. MAP has been measured for the proposed approach and the numbers were compared to state-of-the-art approaches. The results are outlined in Table 1 and here as well the MAP value outperforms previous methods. An example query image and the top 2 retrieval results for category 'person' are shown in Figs. 4d,4e,4f.

The main reason for the increase in MAP value is the use of the ResNet-50 model which is a good feature extraction model. Another reason is the introduction of a novel hash center update scheme as the hash centers are dynamically adapted. Also, from the inspection of training curves, it is clear that the loss reaches close to the lower bound, which in turn indicates that the correlation values are close to 1. Further, for addressing the reason for better performance, we have conducted an ablation study that indicates the importance of the hash loss. The results of the ablation study, more retrieval results, t-SNE plots, train and test loss for all bit lengths are given in the supplementary file. [1].

5. CONCLUSIONS AND OUTLOOK

In this paper, we have proposed an approach for learning hash codes for image retrieval. The neural network is trained using a loss function in such a way that the correlation between the hash codes and hash centers is maximized. Canonical Correlation Analysis (CCA) is utilized in the loss formulation, and hash codes and hash centers are chosen as the two views of CCA. The network is also trained using the classification loss, which maximizes the correlation between category labels and classification scores. The hash centers are dynamically updated. The experimental results substantiate the generation of an optimized feature space with minimum intra-class scatter and maximum inter-class scatter. This is in fact due to the equivalence of CCA with Linear Discriminant Analysis [34]. As a future research, more effective representations of individual classes could be explored.

[1] http://www.ient.rwth-aachen.de/cms/cca-22/

6. REFERENCES

[1] Li H. Zhou, W. and Q. Tian, "Recent advance in content-based image retrieval: A literature survey," *arXiv preprint arXiv:1706.06064*, 2017.

[2] J. Wang, T. Zhang, N. Sebe, H. T. Shen, et al., "A survey on learning to hash," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 4, pp. 769–790, 2017.

[3] Long M. Wang J. Cao, Z. and P. S. Yu, "Hashnet: Deep learning to hash by continuation," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 5608–5617.

[4] Liu W. Kumar S. Wang, J. and S. F. Chang, "Learning to hash for indexing big data-a survey," *Proceedings of the IEEE*, vol. 104, no. 1, pp. 34–57, 2015.

[5] B. Kulis and T. Darrell, "Learning to hash with binary reconstructive embeddings.," in *NIPS*. Citeseer, 2009, vol. 22, pp. 1042–1050.

[6] Li J. Yan S. Zhang B. Ji, J. and Q. Tian, "Super-bit locality-sensitive hashing," in *Advances in neural information processing systems*. Citeseer, 2012, pp. 108–116.

[7] Y. Mu and S. Yan, "Non-metric locality-sensitive hashing," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2010, vol. 24.

[8] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni, "Locality-sensitive hashing scheme based on p-stable distributions," in *Proceedings of the twentieth annual symposium on Computational geometry*, 2004, pp. 253–262.

[9] Motwani R. Indyk, P., "Approximate nearest neighbors: towards removing the curse of dimensionality," in *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, 1998, pp. 604–613.

[10] Indyk P. Motwani R. Gionis, A. et al., "Similarity search in high dimensions via hashing," in *Vldb*, 1999, vol. 99, pp. 518–529.

[11] J. Wang, H. T. Shen, J. Song, and J. Ji, "Hashing for similarity search: A survey," *arXiv preprint arXiv:1408.2927*, 2014.

[12] A. Krizhevsky, G. Hinton, et al., "Learning multiple layers of features from tiny images," 2009.

[13] K. Lin, J. Lu, C-S Chen, and J. Zhou, "Learning compact binary descriptors with unsupervised deep neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 1183–1192.

[14] S. Huang, Y. Xiong, Y. Zhang, and J. Wang, "Unsupervised triplet hashing for fast image retrieval," in *Proceedings of the on Thematic Workshops of ACM Multimedia 2017*, 2017, pp. 84–92.

[15] W. J. Li, S. Wang, and W. C. Kang, "Feature learning based deep supervised hashing with pairwise labels," *arXiv preprint arXiv:1511.03855*, 2015.

[16] Shi Y. Wang, X. and K. M. Kitani, "Deep supervised hashing with triplet labels," in *Asian conference on computer vision*. Springer, 2016, pp. 70–84.

[17] Q. Y. Jiang, X. Cui, and W. J. Li, "Deep discrete supervised hashing," *IEEE Transactions on Image Processing*, vol. 27, no. 12, pp. 5996–6009, 2018.

[18] Ren L. Lu J. Zhou J. Yuan, X., "Relaxation-free deep hashing via policy gradient," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 134–150.

[19] A. Jose, E. S. Ottlik, C. Rohlfing, and J. R. Ohm, "Optimized feature space learning for generating efficient binary codes for image retrieval," *Signal Processing: Image Communication, under final revision*, 2021.

[20] Y. Gong, S. Lazebnik, A. Gordo, and F. Perronnin, "Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 12, pp. 2916–2929, 2012.

[21] Wang T. Zhang X. Tay F. EH Jie Z. Liu W. Yuan, L. and J. Feng, "Central similarity quantization for efficient image and video retrieval," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 3083–3092.

[22] Chang Y-T Qin H. Hung W-C Tsai Y-H. Hong, W. and M-H Yang, "Image hashing via linear discriminant learning," in *The IEEE Winter Conference on Applications of Computer Vision*, 2020, pp. 2531–2539.

[23] Maire-M. Belongie S. Hays J.-Perona P. Ramanan-D. Dollár P. Lin, T-Y and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*. Springer, 2014, pp. 740–755.

[24] Tang-J. Hong R. Li H.-Luo Z. Chua, T-S. and Y. Zheng, "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*, 2009, pp. 1–9.

[25] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne.," *Journal of machine learning research*, vol. 9, no. 11, 2008.

[26] Sun Z.-He R. Li, Q. and T. Tan, "Deep supervised discrete hashing," *arXiv preprint arXiv:1705.10999*, 2017.

[27] X. Wang, Y. Shi, and K. M. Kitani, "Deep supervised hashing with triplet labels," in *Asian conference on computer vision*. Springer, 2016, pp. 70–84.

[28] Long M.-Wang J. Zhu, H. and Y. Cao, "Deep hashing network for efficient similarity retrieval," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2016, vol. 30.

[29] Pan Y.-Liu Y. Lai, H. and S. Yan, "Simultaneous feature learning and hash coding with deep neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3270–3278.

[30] Pan Y.-Lai H. Liu C. Xia, R. and S. Yan, "Supervised hashing for image retrieval via image representation learning," in *Proceedings of the AAAI conference on artificial intelligence*, 2014, vol. 28.

[31] Zhang X.-Ren S. He, K. and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[32] Deng J.-Su H. Krause J. Satheesh-S. Ma S.-Huang Z. Russakovsky, O., A. Karpathy, A. Khosla, M Bernstein, et al., "Imagenet large scale visual recognition challenge," *International journal of computer vision*, vol. 115, no. 3, pp. 211–252, 2015.

[33] K. V. Mardia, "Multivariate analysis," Tech. Rep., 1979.

[34] A. J. Izenman, "Linear discriminant analysis," in *Modern multivariate statistical techniques*, pp. 237–280. Springer, 2013.