Similarity Preserving Deep Asymmetric Quantization for Image Retrieval

Junjie Chen, William K. Cheung
Department of Computer Science, Hong Kong Baptist University, Hong Kong, China
{csjjchen, william}@comp.hkbu.edu.hk

Abstract

Quantization has been widely adopted for large-scale multimedia retrieval due to its effectiveness of coding high-dimensional data. Deep quantization models have been demonstrated to achieve the state-of-the-art retrieval accuracy. However, training the deep models given a large-scale database is highly time-consuming as a large amount of parameters are involved. Existing deep quantization methods often sample only a subset from the database for training, which may end up with unsatisfactory retrieval performance as a large portion of label information is discarded. To alleviate this problem, we propose a novel model called Similarity Preserving Deep Asymmetric Quantization (SPDAQ) which can directly learn the compact binary codes and quantization codebooks for all the items in the database efficiently. To do that, SPDAQ makes use of an image subset as well as the label information of all the database items so the image subset items and the database items are mapped to two different but correlated distributions, where the label similarity can be well preserved. An efficient optimization algorithm is proposed for the learning. Extensive experiments conducted on four widely-used benchmark datasets demonstrate the superiority of our proposed SPDAQ model.

Introduction

The sheer volume of high dimensional multimedia data like images and videos in search engines and social networks makes precise and yet efficient retrieval a challenging research problem which has attracted a lot of attention in recent years (Wang et al. 2018; Yu et al. 2018).

To achieve the two often conflicting objectives of precision and efficiency, *hashing* and *quantization*, as two most popular Approximated Nearest Neighbor (ANN)(Andoni and Indyk 2006) approaches, have been widely used due to their low storage cost and computational efficiency. Basically, hashing and quantization both encode high-dimensional data items with compact binary codes, where the similarities between items can be preserved, *i.e.*, indexing semantically similar images with similar binary codes.

Image hashing aims to learn a mapping function that projects real-valued visual features into a binary vector space. More conventional methods (Gong and Lazebnik 2011; Liu et al. 2012; Shen et al. 2015; Liu et al. 2018b; Chen, Wang, and Cheung 2018) learn the hashing function based on some pre-defined features. With the progress of deep learning (Krizhevsky, Sutskever, and Hinton 2012), deep learning based hashing methods have been proposed (Zhu et al. 2016; Li, Wang, and Kang 2015; Li et al. 2017; Qiu et al. 2017; Lai et al. 2015), where compatible visual features and binary hash codes can be simultaneously learned from raw pixel images using one deep network.

Nevertheless, the Hamming distance is less distinct and would lose much fine-grained information as discussed in (Chen, Cheung, and Wang 2018; He et al. 2018), leading to unsatisfactory retrieval performance. In parallel, quantization is another popular coding approach that can avoid these two problems as it tries to minimize the distance between items with learned binary indexes and their corresponding real-value codewords, resulting in more accurate ranking and fine-grained information preserved. Existing quantization methods like vector quantization (VQ) (Gray 1984), production quantization (PQ) (Jegou, Douze, and Schmid 2011) and composite quantization (CQ) (Zhang, Du, and Wang 2014) are designed for unsupervised coding based on some predefined features. Recently, a number of deep supervised quantization methods have been proposed (Cao et al. 2016; 2017; Yu et al. 2018; Liu et al. 2018a) to incorporate quantization into a deep supervised learning framework so that the feature learning and quantization can be jointly optimized. Thus, the extracted visual features from a deep model (e.g. deep convolution network) can result in more compatible codebooks, leading to a higher retrieval accuracy.

Most of the deep quantization methods use deep convolution network as the backbone for image representations learning. As a large amount of parameters is needed for convolution networks, training deep quantization models given a large-scale database is time-consuming. For example, the complexity of triplet training in DVSQ (Cao et al. 2017) is $\mathcal{O}(N^3)$, which grows cubically as N increases. To make it tractable, existing deep quantization models only sample a subset from the database for training. Unfortunately, discarding a large portion of the supervision information often leads to poor retrieval performance as the inferred codebooks will not good enough for quantizing the whole large-scale database. In this paper, we propose a novel quantization model named Similarity Preserving Deep Asymmetric

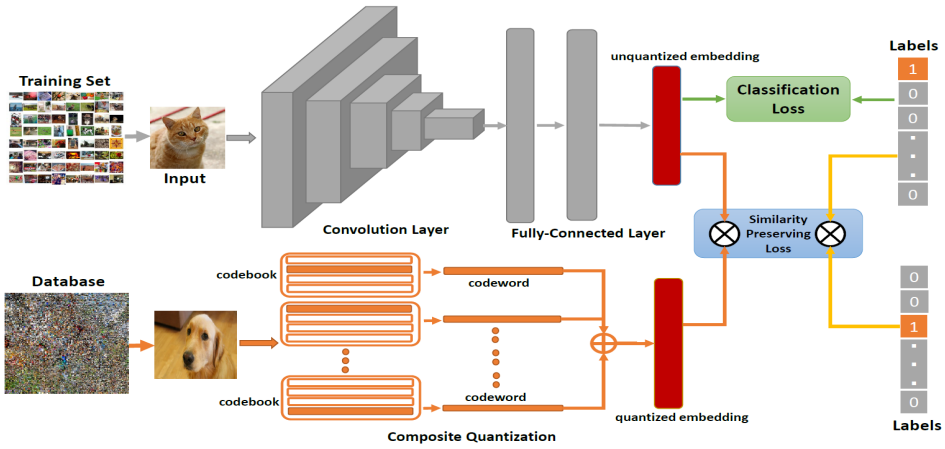


Figure 1: The framework of the proposed SPDAQ model which make use of a CNN network and composite quantization to learn unquantized and quantized embeddings for training set images and database images respectively.

Quantization (SPDAQ) to address the aforementioned challenge. The framework of proposed SPDAQ is shown as in Figure 1. Our contributions are summarized as follows:

- Unlike existing deep quantization methods which perform metric learning on unquantized embeddings, our model takes the first attempt to adopt Asymmetric Quantizer Distance (AQD) (Jegou, Douze, and Schmid 2011) to approximate the predefined metric for optimization. Furthermore, SPDAQ can directly learn quantization codebooks and binary codes for each database items by formulating it as Maximum Inner Product Search (MIPS) (Shrivastava and Li 2014) problem. To the best of our knowledge, it is the first time to learn directly the quantization codebooks and the explicit binary codes for each database item discretely in a deep learning framework.
- By sampling a subset of M images as input into the deep convolution network, the complexity of the pairwise training can be reduced to $\mathcal{O}(MN)$ ($M \ll N$), which allows us to utilize all the available label information efficiently. Specifically, the subset images and database items are mapped to two different but correlated distributions, where the pairwise label similarity is largely preserved by the AQD between the inferred image embedding and the composite quantized representation.
- Furthermore, a well-designed optimization algorithm is derived which is highly scalable to large-scale datasets.
- Extensive experiments conducted on four widely-used benchmark datasets show that SPDAQ outperforms the existing deep quantization methods and achieves the state-of-the-art performance for the image retrieval task.

Related Work

Deep Pairwise Supervised Hashing (DPSH) (Li, Wang, and Kang 2015) and Deep Hashing Network (DHN) (Zhu et al. 2016) are two deep models which jointly learn visual features and hash codes by approximating pairwise similarity. Deep Supervised Discrete Hashing (DSDH)(Li et al. 2017) tries to preserve discrete constraints, and DSH-GANs (Qiu et al. 2017) learns to hash with a generative adversarial network (Goodfellow et al. 2014). In parallel, DQN (Cao et al. 2016) and PQN (Yu et al. 2018) were proposed to jointly learn product quantization codebooks and visual features in a supervised deep learning framework. DTQ (Liu et al. 2018a) adopts composite quantization and learns visual features in a triplet manner. DVSQ (Cao et al. 2017) learns quantization codebooks with labels and auxiliary text embeddings.

NAMVH (Da et al. 2018) is a model which is similar to our proposed SPDAQ model where the quantized embeddings for all the database items are also explicitly learned. However, they differ in a number of aspects. NAMVH adopts binary embeddings with limited model capacity as a consequence while SPDAQ uses composite quantization which gives real-valued embeddings so that the label similarity can be better preserved thus resulting in more accurate ranking. Also, NAMVH takes a set of pre-defined features as input and feed them to a fully connected multi-layer feed-forward network for non-linear dimension reduction. CNN is adopted in SPDAQ instead to learn features directly from raw images. To make the CNN training with large-scale database possible, SPDAQ's pairwise training complexity is $\mathcal{O}(MN)$ ($M << N$), while that of NAMVH is quadratic $\mathcal{O}(N^2)$.

Methodology

In this paper, we propose a novel model named Similarity Preserving Deep Asymmetric Quantization (SPDAQ) that tries to learn unquantized embeddings for an image subset (for the efficiency reason) and at the same time another set of composite quantized embeddings for the whole database so that the Asymmetric Quantizer Distance (AQD) between the unquantized and quantized representations should preserve the image label similarity. Details of the model formulation and the optimization algorithm are presented in the following.

Notation

We use lowercase letters like $\mathbf{b}$ to denote a vector while uppercase letters like $\mathbf{B}$ to denote a matrix. $\mathbf{B}_{ij}$ denotes the $(i,j)th$ element of matrix $\mathbf{B}$. $\mathbf{B}_{*j}$ denotes the jth column and $\mathbf{B}_{i*}$ denotes the ith row of matrix $\mathbf{B}$ respectively. Furthermore, we will use $\|\cdot\|_F$, $\|\cdot\|_2$ and $\|\cdot\|_0$ to denote Frobenius norm, L_2 norm and L_0 norm.

Model Formulation

Given a large image database with N items $\mathbf{X}^{data} = \{\mathbf{x}_i\}_{i=1}^N$ and their corresponding semantic labels $\mathbf{Y}^{data} = \{\mathbf{y}_i\}_{i=1}^N$, the problem is to perform the quantization by learning D codebooks $\{\mathbf{C}^i\}_{i=1}^D$ for the construction of look-up tables where each entry (i.e. codeword) is indexed by its corresponding D binary vectors $\{\mathbf{b}^i\}_{i=1}^D$. High efficient image retrieval can thus be achieved via the look-up tables. As most of the deep learning methods proposed for the learning task (Cao et al. 2017; 2016; Li et al. 2017; Li, Wang, and Kang 2015), we also first sample a small subset of M items for training $\mathbf{X}^{train} = \{\mathbf{x}_i\}_{i=1}^M$ and $\mathbf{Y}^{train} = \{y_i\}_{i=1}^M$. Only the subset of images are fed into the deep model for parameter tuning to avoid linearly scanning the whole database, which is very time-consuming.

In this paper, we adopt composite quantization following (Cao et al. 2017; Liu et al. 2018a) as the basis due to its superiority over other quantization methods and the matrix implementation simplicity. The overall framework of our proposed SPDAQ model is depicted in Figure 1. It contains three parts: (i) a deep convolution network to learn the unquantized visual embeddings for the image subset, (ii) a composite quantization part to directly learn the quantization codebooks and the indexes for the database items, and (iii) an objective functions including a classification loss and a similarity preserving loss to guide the model training.

Similarity Matrix Construction Inspired by the Maximum Inner Product Search (MIPS), we adopt pairwise training and compute the pairwise similarity matrix $\mathbf{S} \in [0,1]^{M\times N}$ from labels. For two items $\mathbf{x}_i$ and $\mathbf{x}_j$ associated with label vectors $\mathbf{y}_i$ and $\mathbf{y}_j$, we compute their similarity as

$$sim(\mathbf{x}_i, \mathbf{x}_j) = \frac{\mathbf{y}_i^T \mathbf{y}_j}{\|\mathbf{y}_i\|_2 \|\mathbf{y}_j\|_2} \in [0,1] \tag{1}$$

Note that the complexity of similarity matrix $\mathbf{S}$ is $\mathcal{O}(MN)$, which is unacceptable for large-scale datasets given a large value of N even though $M \ll N$. To address this high complexity issue, we rewrite matrix $\mathbf{S}$ as the multiplication of two matrices

$$\mathbf{S} = \mathbf{L}^{train}(\mathbf{L}^{data})^T \tag{2}$$

where $\mathbf{L}_{i*}^{train} = \frac{\mathbf{y}_i^T}{\|\mathbf{y}_i\|_2}$ is a L_2-norm normalized label vector. A similar normalization trick can be applied to Y^{data} to get the normalized label matrix $\mathbf{L}^{data}$.

Unquantized Embedding Learning Following (Li, Wang, and Kang 2015; Li et al. 2017), we adopt CNN-F (Chatfield et al. 2014) as the backbone to learn unquantized visual embeddings for the image subset. CNN-F is a deep convolution network consisting of five convolution layers and three fully-connected layers as AlexNet (Krizhevsky, Sutskever, and Hinton 2012). We replace the last fully-connected layer with a linear projection which maps the seventh high-dimensional visual features to low-dimensional embeddings.

Similarity Preserving Loss Given a query sample $\mathbf{q}$ and a database item $\mathbf{x}$, the AQD as in (Cao et al. 2017) is formulated as

$$AQD(\mathbf{q}, \mathbf{x}) = (\mathbf{z}_q)^T \sum_{i=1}^{D} \mathbf{C}^i \mathbf{b}^i \tag{3}$$

where $\mathbf{z}_q$ is an unquantized embedding of the query sample $\mathbf{q}$. We directly adopt Eq. (3) to approximate the similarity matrix $\mathbf{S}$ computed with Eq. (2). Thus, the similarity preserving loss is formulated as a constrained optimization problem, given as

$$\begin{aligned} \min \quad & \sum_{i=1}^{M}\sum_{j=1}^{N} \|\mathcal{F}(\mathbf{x}_i)^T \mathbf{z}_j - \gamma \cdot \mathbf{S}_{ij}\|_2^2 \\ \text{s.t.} \quad & \mathbf{z}_j = \sum_{p=1}^{D} \mathbf{C}^p \mathbf{b}_j^p,\ \|\mathcal{F}(\mathbf{x}_i)\|_2 = 1 \\ & \mathbf{b}_j^p \in \{0,1\}^K, \|\mathbf{b}_j^p\|_0 = 1, \forall p \in \{1 \dots D\} \end{aligned} \tag{4}$$

where $\mathcal{F}(\mathbf{x}_i) \in R^L$ is the low-dimensional unquantized embedding obtained through a deep convolution network for raw pixel image $\mathbf{x}_i$ from the subset. γ is a suitably selected scaling parameter which scales the similarity as the range of distance. We simply set γ as the number of codebooks $\gamma = D$. $\mathbf{z}_j$ is the embedding for jth item in the database obtained by composite quantization $\sum_{p=1}^{D} \mathbf{C}^p \mathbf{b}_j^p$, where $\mathbf{C}^p \in R^{L\times K}$ denotes the pth codebook and each codebook contains K codewords. $\mathbf{b}_j^p$ is a one-hot vector of jth item associated with $\mathbf{C}^p$. Since the value of inner product is not only affected by the angle between vectors but also their norm scale, we add the unit norm constraint to the unquantized embedding but relax this constraint on database items for model training flexibility. The unit-norm constraint is easily satisfied using the internal operator in Tensorflow.

Eq. (4) can be equivalently written as

$$\min \sum_{i=1}^{M}\sum_{j=1}^{N} \|\mathcal{F}(\mathbf{x}_i)^T \mathbf{z}_j - \gamma \cdot \mathbf{S}_{ij}\|_2^2 + \eta \sum_{j=1}^{N} \|\mathbf{z}_j - \sum_{p=1}^{D} \mathbf{C}^p \mathbf{b}_j^p\|_2^2 \tag{5}$$

The newly introduced parameter η is important to balance the similarity approximation and the quantization error, which, however, is time-consuming to tune. Equivalently, we take one more step by directly learning the quantized embedding for database items in a discrete way, formulated as

$$\begin{aligned} \min \quad & \sum_{i=1}^{M}\sum_{j=1}^{N} \|\mathcal{F}(\mathbf{x}_i)^T \sum_{p=1}^{D} \mathbf{C}^p \mathbf{b}_j^p - \gamma \cdot \mathbf{S}_{ij}\|_2^2 \\ \text{s.t.} \quad & \mathbf{b}_j^p \in \{0,1\}^K, \|\mathbf{b}_j^p\|_0 = 1, \forall p \in \{1 \dots D\} \\ & \|\mathcal{F}(\mathbf{x}_i)\|_2 = 1. \end{aligned} \tag{6}$$

Note that the unquantized embeddings for subset samples are obtained through the deep network, while the quantized embeddings for database items are directly learned by preserving the correlations (i.e. the predefined similarity). By doing so, SPDAQ actually maps subset samples and database items to two different but correlated distributions, which makes the model more flexible and better preserve the semantic similarity. Also, utilizing more supervision information ensures that the learned quantization codebooks can encode more semantic information, thus leading to better performance in retrieval stage. We will verify this in experimental part.

Eq. (6) can be rewritten in a matrix form as

$$\begin{aligned} \min \quad & Q = \|\mathcal{F}(\mathbf{X})^T\mathbf{C}\mathbf{B} - \gamma \cdot \mathbf{S}\|_F^2 \\ \text{s.t.} \quad & \mathbf{B} \in \{0,1\}^{DK\times N} \\ & \mathbf{b}_j \in \{0,1\}^{DK}, \|\mathbf{b}_j^p\|_0 = 1, \forall p \in \{1\dots D\} \\ & \|\mathcal{F}(\mathbf{x}_i)\|_2 = 1, \forall i \in \{1\dots M\} \end{aligned} \tag{7}$$

where $\mathcal{F}(\mathbf{X}) \in R^{L\times M}$ is the embedding matrix of subset images obtained through CNN. $\mathbf{C}$ is the concatenation of D codebooks $\mathbf{C} = [\mathbf{C}^1 \cdots \mathbf{C}^D] \in R^{L\times KD}$, and $\mathbf{B}$ is the concatenation of N binary vectors $\mathbf{B} = [\mathbf{b}_1 \cdots \mathbf{b}_N] \in \{0,1\}^{DK\times N}$ of N database items.

Classification Loss To learn discriminative embeddings of subset images, a classification loss is added

$$\min \quad \sum_{i=1}^{M} H(\mathcal{F}(\mathbf{x}_i), \mathbf{y}_i) \tag{8}$$

where label vector $\mathbf{y}_i$ could be one-hot (i.e. single-label dataset) or k-hot (i.e. multi-label dataset). H is a predefined classification loss function. For classifying single-label dataset, H will be a softmax loss function. For multi-label datasets, binary entropy is applied to each label as the classification loss.

Overall Objective We formulate the overall objective as:

$$\begin{aligned} \min \ L(\theta, \mathbf{C},\mathbf{B}) &= Q + \lambda \sum_{i=1}^{M} H(\mathcal{F}(\mathbf{x}_i), \mathbf{y}_i) \\ &= \|\mathcal{F}(\mathbf{X})^T\mathbf{C}\mathbf{B} - \gamma \cdot \mathbf{S}\|_F^2 + \lambda \sum_{i=1}^{M} H(\mathcal{F}(\mathbf{x}_i), \mathbf{y}_i) \\ \text{s.t.} \ & \mathbf{B} \in \{0,1\}^{DK\times N} \\ & \mathbf{b}_j \in \{0,1\}^{DK}, \|\mathbf{b}_j^p\|_0 = 1, \forall p \in \{1\dots D\} \\ & \|\mathcal{F}(\mathbf{x}_i)\|_2 = 1, \forall i \in \{1\dots M\} \end{aligned} \tag{9}$$

where λ is a hyper-parameter to balance the classification loss and the similarity preserving loss, which is adaptively updated. θ denotes the deep convolution network parameters.

Optimization

We propose an alternative algorithm to optimize the objective function Eq. (9). We learn the visual embeddings $\mathcal{F}(\mathbf{X})$, the codebooks $\mathbf{C}$ and the binary index $\mathbf{B}$ alternatively. Each step decreases the objective Eq. (9) until convergence.

Initialization We randomly initialize each codeword in a codebook $\mathbf{C}$ and then normalize each codeword to unit length using L_2 norm. The binary index $\mathbf{b}_j^p$ is randomly initialized to be a one-hot vector.

Learning $\mathcal{F}(\mathbf{X})$ with C and B fixed The standard back propagation (BP) is used to update the model parameters θ and $\mathcal{F}(\mathbf{X})$. We take the gradient of the objective $L(\theta, \mathbf{C}, \mathbf{B})$ with respect to θ as follows:

$$\begin{aligned} \frac{\partial L(\theta, \mathbf{C}, \mathbf{B})}{\partial \theta} =& 2 \cdot ((\mathbf{C}\mathbf{B})^T(\mathbf{C}\mathbf{B})\mathcal{F}(\mathbf{x}_i) - \gamma \mathbf{C}\mathbf{B}\mathbf{S}_{i*}^T)\frac{\mathcal{F}(\mathbf{x}_i)}{\partial \theta} \\ &+ \lambda \frac{H(\mathcal{F}(\mathbf{x}_i), \mathbf{y}_i)}{\partial \theta} \end{aligned} \tag{10}$$

To optimize the deep model, we back-propagate the gradient $\frac{\partial L(\theta, \mathbf{C}, \mathbf{B})}{\partial \theta}$ through the chain rule and update the parameters using the gradient descent method.

Learning C with $\mathcal{F}(\mathbf{X})$ and B fixed To optimize $\mathbf{C}$, we first filter out non-related terms in Eq. (9) and obtain

$$\min \quad \|\mathcal{F}(\mathbf{X})^T\mathbf{C}\mathbf{B} - \gamma \cdot \mathbf{S}\|_F^2. \tag{11}$$

Many optimization methods can be used to optimize Eq. (11), such as gradient descent. Here, we introduce an auxiliary parameter $\mathbf{Z}$ so that analytical solutions can be obtained.

Let $\mathbf{Z} = \mathcal{F}(\mathbf{X})^T\mathbf{C}$. We can rewrite Eq. (11) as a constrained optimization problem, given as

$$\begin{aligned} \min \quad & \|\mathbf{Z}\mathbf{B} - \gamma \cdot \mathbf{S}\|_F^2 \\ \text{s.t.} \quad & \mathbf{Z} = \mathcal{F}(\mathbf{X})^T\mathbf{C} \end{aligned} \tag{12}$$

which can be further rewritten as a Lagrange function

$$\min \quad \|\mathbf{Z}\mathbf{B} - \gamma \cdot \mathbf{S}\|_F^2 + \mu\|\mathbf{Z} - \mathcal{F}(\mathbf{X})^T\mathbf{C}\|_F^2 \tag{13}$$

where μ is a hyper-parameter which is simply set as $\mu = 1.0$. Then, we can alternatively update $\mathbf{Z}$ and $\mathbf{C}$ until convergence. For updating $\mathbf{C}$, we take the derivate of Eq. (13) w.r.t $\mathbf{C}$ as

$$\mathcal{F}(\mathbf{X})\mathcal{F}(\mathbf{X})^T\mathbf{C} - \mathcal{F}(\mathbf{X})\mathbf{Z} = 0 \tag{14}$$

The updating formulation for $\mathbf{C}$ becomes

$$\mathbf{C} = (\mathcal{F}(\mathbf{X})\mathcal{F}(\mathbf{X})^T)^{-1}\mathcal{F}(\mathbf{X})\mathbf{Z}. \tag{15}$$

Similarly, the updating formulation for $\mathbf{Z}$ will be

$$\mathbf{Z} = (\gamma \cdot \mathbf{S}\mathbf{B}^T + \mathcal{F}(\mathbf{X})^T\mathbf{C})(\mathbf{B}\mathbf{B}^T + \mathbf{I})^{-1}. \tag{16}$$

Computing $\mathbf{S}$ explicitly incurs high complexity. With the factorization in Eq. (2), we can first compute $\mathbf{G} = (\mathbf{L}^{data})^T\mathbf{B}^T$ and then compute $\mathbf{S}\mathbf{B}^T = \mathbf{L}^{train}\mathbf{G}$ to avoid the high complexity problem.

Learning B with $\mathcal{F}(\mathbf{X})$ and C fixed We first rewrite the objective by keeping only the parts related to $\mathbf{B}$, given as

$$\begin{aligned} \min \ & \|\mathcal{F}(\mathbf{X})^T\mathbf{C}\mathbf{B} - \mathbf{S}\|_F^2 \\ \text{s.t.} \ & \mathbf{b}_j \in \{0,1\}^{DK}, \mathbf{B} \in \{0,1\}^{DK\times N} \\ & \|\mathbf{b}_j^p\|_0 = 1, \forall p \in \{1\dots D\}. \end{aligned} \tag{17}$$

For convenience, we eliminate the scale term γ. With the binary constraint on $\mathbf{b}$, Eq. (17) becomes a mixed-integer

programming problem (MIP). To solve an MIP, greedy local search is usually applied to find an approximated solution.

For the ith item, the binary vector is a concatenation of D binary sub-vector $\mathbf{b}_i = [\mathbf{b}_i^1 \cdots \mathbf{b}_i^D]$. We define its neighbor as $\mathbf{b}_i^{'}$ with only one binary sub-vector different (e.g. $\mathbf{b}_i^j \neq \mathbf{b}_i^{j^{'}}$), and then perform local search. That means we iteratively update $\{\mathbf{b}_i^j\}_{j=1}^D$.

Note that each item $\mathbf{b}_i$ in $\mathbf{B}$ is independent of each other. Since the size of database N is typically large, we can optimize B batch by batch. We first sample a batch index $\Omega \in \{1 \cdots N\}$. Then we obtain $\mathbf{B}_\Omega \in \{0,1\}^{DK \times \Omega}$ and $\mathbf{S}_\Omega \in \{0,1\}^{M \times \Omega}$. Note that $\mathbf{S}_\Omega$ can be computed explicitly since $|\Omega|$ can be small. In the following, we reuse notations $\mathbf{B}$ and $\mathbf{S}$ to denote the sampled sub-matrix for convenience.

Let $\mathbf{Z} = \mathcal{F}(\mathbf{X})^T\mathbf{C} \in R^{M \times DK}$, which is decomposed as D components $\mathbf{Z} = [\mathbf{Z}^1, \cdots, \mathbf{Z}^D]$. Similarly, we have $\mathbf{B} = [\mathbf{B}^1, \cdots, \mathbf{B}^D]$. Eq. (17) can be rewritten as

$$\min \quad \|\sum_{i=1}^{D} \mathbf{Z}^i\mathbf{B}^i - \mathbf{S}\|_F^2 \tag{18}$$

For the optimization of the ith component, we can rewrite Eq. (18) as

$$\min \quad \|\mathbf{Z}^i\mathbf{B}^i - \widehat{\mathbf{S}}\|_F^2 \tag{19}$$

where $\widehat{\mathbf{S}} = \mathbf{S} - \sum_{k=1,k\neq i}^{D} \mathbf{Z}^k\mathbf{B}^k$. Minimizing Eq. (19) is equivalent to

$$\min \quad Tr((\mathbf{B}^i)^T(\mathbf{Z}^i)^T\mathbf{Z}^i\mathbf{B}^i) - Tr(\widehat{\mathbf{S}}^T\mathbf{Z}^i\mathbf{B}^i) \tag{20}$$

where $Tr(\cdot)$ is the trace norm. Let $\mathbf{G} = diag((\mathbf{Z}^i)^T\mathbf{Z}^i) - \widehat{\mathbf{S}}^T\mathbf{Z}^i$, where $diag(\cdot)$ means selecting the diagonal vector. The solution of Eq. (20) is to set the element in each column of $\mathbf{B}^i$ as 1, whose index equals to the index of the minimum value in each row of $\mathbf{G}$.

Out-of-Sample Extension

For testing, given a new query point $\mathbf{x}_q$, we can compute the AQD as formulated in Eq. (3) between $\mathbf{x}_q$ and the data item $\mathbf{x}_i$ in the image database based on the inferred binary codes $\mathbf{B}$ and codebooks $\mathbf{C}$ for the retrieval.

To compute the AQD efficiently, we first pre-compute the inner product look-up table of size $1 \times K$ between the embedding $\mathbf{z}_q$ and codewords. So, there will be D $1 \times K$ look-up tables altogether. With the D pre-computed look-up tables, the distance $AQD(\mathbf{q}, \mathbf{x})$ can be efficiently computed by summing up the codeword distance directly obtained from look-up tables. It is only slightly more costly than computing the Hamming distance (Jegou, Douze, and Schmid 2011).

Experimental Results

We apply the proposed SPDAQ model to a number of publicly available image datasets and compare its performance with a number of state-of-the-art methods.

Datasets and Experimental Settings

Four datasets are adopted: CIFAR-10 (Krizhevsky and Hinton 2009), NUS-WIDE-21, NUS-WIDE-81 (Chua et al. 2009) and MS-COCO (Lin et al. 2014).

CIFAR-10 contains $60,000$ labeled color images in 10 classes. Each class contains $6,000$ images of size 32×32.

NUS-WIDE-81 contains $269,648$ labeled images collected from Flickr in 81 classes. It is a multi-label dataset and each image is annotated with some of the 81 labels. Note that it is challenging because it is highly class-imbalanced, where some labels are associated with tens of thousands of images while some labels with only tens of images.

NUS-WIDE-21 is a subset of NUS-WIDE-81, containing $195,834$ images. Images in NUS-WIDE-81 associated with the most-frequent 21 labels are selected. In this way, each label will be associated with at least $5,000$ images.

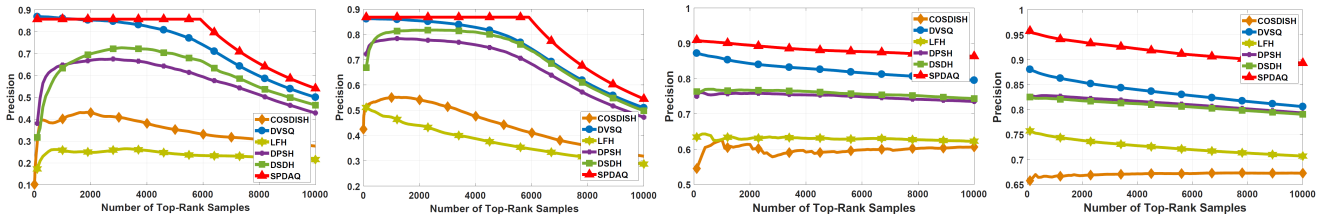
MS-COCO contains $82,783$ training images and $40,504$ validation images, where each image is labeled by some of the 80 labels. After pruning the images without labels, we obtain $122,218$ images in total. This dataset is also highly class-imbalanced.

We compare the performance of the proposed SPDAQ model with several state-of-the-art models including hashing and quantization : **COSDISH** (Kang, Li, and Zhou 2016), **LFH** (Zhang et al. 2014), **DSH-GAN** (Qiu et al. 2017), **DAPH**(Shen et al. 2017), **DSDH** (Li et al. 2017), **DPSH** (Li, Wang, and Kang 2015), **PQN** (Yu et al. 2018), **DQN** (Cao et al. 2016), **DTQ** (Liu et al. 2018a), **DHN** (Zhu et al. 2016) and **DVSQ** (Cao et al. 2017). All of them are deep models, except COSDISH and LFH.

For CIFAR-10, we randomly select $1,000$ images (100 images per class) to form the testing query set and take the rest $59,000$ images as the database as in (Liu et al. 2018a; Li, Wang, and Kang 2015). Since training with the whole image database is time-consuming for the existing deep quantization methods, we follow the original settings as in their papers and sample a subset of $5,000$ images (500 images per class) from the database for training. For NUS-WIDE-21, we adopt the widely-used protocol and randomly sample $2,100$ images (100 images per class) as the testing query set while the remaining images as the retrieval database. A subset of $10,500$ images (500 images per class) will be further sampled for training. Note that our proposed SPDAQ model also only has the sampled subset of images fed into the CNN for learning the parameters as the existing methods. For the conventional methods COSDISH and LFH, we use the identical training set as that of the deep methods. For fair comparison, we use the deep features extracted from the 7$^{\text{th}}$ layer of CNN-F as the inputs for both COSDISH and LFH. For the more challenging datasets MS-COCO and NUS-WIDE-81, we randomly sample $10,000$ images as the subset for training and $5,000$ images as the testing query set, as in (Liu et al. 2018a). The remaining images forms the retrieval database. For the dimension of the inferred embedding, we set $L = 300$ for CIFAR-10, NUS-WIDE-81 and MS-COCO as in (Cao et al. 2017; Liu et al. 2018a). For NUS-WIDE-21, we set $L = 128$ which works well for our implementation of SPDAQ. The learning rate is fine-tuned in the range of $[10^{-3}, 10^{-7}]$ for each dataset. For the composite quantization, we set the number of codewords in each codebook as $K = 256$. Therefore, each one-hot vector $\mathbf{b}$ can be encoded using a binary

Method	CIFAR-10 (mAP@All)				NUS-WIDE-21 (mAP@5000)			
	8 bits	24 bits	32 bits	48 bits	8 bits	24 bits	32 bits	48 bits
COSDISH	33.7	42.3	46.9	47.6	60.4	66.4	69.8	72.4
LFH	23.2	33.8	36.1	42.9	63.5	73.9	73.2	76.2
DSH-GAN	–	78.1	78.7	80.2	–	85.6	86.1	86.3
DAPH	75.7	82.1	83.1	84.5	71.7	77.1	78.7	81.6
DPSH	62.2	72.7	74.4	75.7	76.8	82.2	83.8	85.1
DSDH	65.0	78.6	80.1	82.0	76.5	80.8	82.0	82.9
PQN	–	–	–	–	–	81.9	82.3	83.0
DQN	52.7	55.8	56.4	58.0	–	77.6	78.3	79.2
DTQ	78.5	79.0	79.2	–	–	–	–	–
DVSQ*	80.3	80.3	80.8	81.1	84.8	85.4	85.6	85.4
SPDAQ	**88.4**	**88.4**	**89.1**	**89.3**	**90.0**	**93.1**	**93.1**	**93.4**

Table 1: Mean Average Precision (%) for different numbers of bits on CIFAR-10 and NUS-WIDE-21 datasets. The best mAPs are shown in bold. DVSQ* denotes we run the codes provided by the authors to get the results.



(a) CIFAR-10 @8bit (b) CIFAR-10 @24bit (c) NUS-WIDE-21 @8bit (d) NUS-WIDE-21 @24bit

Figure 2: Retrieval performance evaluated with Top-K precision curve (@8 bits and @24 bits respectively) on CIFAR-10 and NUS-WIDE-21 datasets. Best view in color.

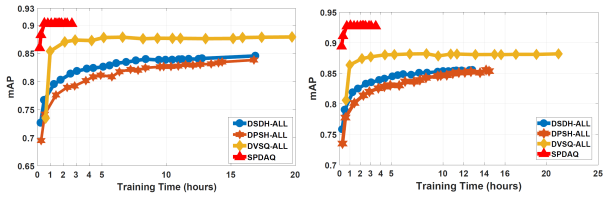
vector with $log_2 K$ bits. Suppose there are D codebooks, the length of the binary code will be $Dlog_2 K = 8D$. We set the number of epochs as 50 for all the datasets.

Retrieval Accuracy

For performance evaluation, we adopt two widely-used evaluation metrics: mean average precision (mAP) and Top-K precision. We compute mAP for CIFAR-10 with respect to the whole database. For NUS-WIDE-21, NUS-WIDE-81 and MS-COCO datasets, we compute mAP@5000 with respect to the top $5,000$ returned images as in (Li et al. 2017; Liu et al. 2018a). All the results of the existing works reported in this paper are obtained either from their corresponding papers or by running the codes provided by the authors. For the proposed SPDAQ model, we report the average results of five runs.

The performance comparison results on CIFAR-10 and NUS-WIDE-21 in terms of mAP are shown in Table 1. We see that the proposed SPDAQ model outperforms the state-of-the-art methods by a large margin, including the deep hashing models and the deep quantization models. In terms of Top-K precision (we set $K = 10,000$), the proposed SPDAQ also significantly outperforms the existing models as shown in Figure 2. Regarding the two more challenging datasets NUS-WIDE-81 and MS-COCO, we find that the proposed SPDAQ also achieves the best results (Table 2).

We believe the significant improvement is due to two reasons: (1) By effectively utilizing all the supervision information of the database, SPDAQ directly learns quantization codebooks and binary codes for the database items, so that more semantic information is encoded. (2) SPDAQ tried to map the subset of the data and the database items to two different but correlated distributions via learning, which can better preserve the similarity.



(a) NUS-WIDE-21 @8bit (b) NUS-WIDE-21 @16bit

Figure 3: Training time evaluation of SPDAQ and other state-of-the-art deep models on large-scale dataset NUS-WIDE-21. Best view in color.

Efficiency for Model Training

We further compare our proposed SPDAQ model with the existing deep models in terms of training efficiency under 8 bits and 16 bits settings. The large-scale NUS-WIDE-21

Method	NUS-WIDE-81 (mAP@5000)				MS-COCO (mAP@5000)			
	8 bits	16 bits	24 bits	32 bits	8 bits	16 bits	24 bits	32 bits
DHN	66.8	70.2	71.3	71.6	60.7	67.7	69.7	70.1
DQN	72.1	73.5	74.7	75.2	64.9	65.3	66.6	68.5
DTQ	79.5	79.8	79.9	80.1	75.8	76.0	76.4	76.7
DVSQ	78.0	79.0	79.2	79.7	70.4	71.2	71.7	72.0
SPDAQ	**80.5**	**84.2**	**85.1**	**85.1**	**80.1**	**84.4**	**84.5**	**84.7**

Table 2: Mean Average Precision (%) on NUS-WIDE-81 and MS-COCO datasets for different deep models. The best results are in bold.

Method	CIFAR-10 (mAP@All)				NUS-WIDE-21 (mAP@5000)			
	8 bits	24 bits	32 bits	48 bits	8 bits	24 bits	32 bits	48 bits
SPDAQ-LD	<u>88.8</u>	<u>88.7</u>	89.0	<u>89.3</u>	89.5	93.0	<u>93.3</u>	<u>93.7</u>
SPDAQ-LL	13.8	17.1	17.5	17.4	38.4	49.6	60.1	65.8
SPDAQ-nQ	**89.4**	**90.1**	**90.1**	**89.9**	**93.5**	**93.6**	**93.8**	**94.1**
SPDAQ-nC	78.6	85.2	85.2	86.5	88.8	92.2	92.9	93.4
SPDAQ	88.4	88.4	<u>89.1</u>	<u>89.3</u>	<u>90.0</u>	<u>93.1</u>	93.1	93.4

Table 3: Mean Average Precision (%) for variants of the proposed SPDAQ model. The best results are in bold, while the second best ones are underlined.

dataset is used for this evaluation. To show the superiority of our model, we use the whole database as input for all the models we tested so that all the supervised labels can be used, denoted as DPSH-ALL, DSDH-ALL and DVSQ-ALL. The learning curves of the different models in terms of mAP@5000 are shown in Figure 3. All the models are evaluated with Nvidia Tesla K80 Dual GPU Module.

For fair comparison, we run all the models for 50 epochs. As shown in Figure 3, our model converges in around 3.5 hours and gives a much better performance in mAP. For DPSH-ALL and DSDH-ALL, both converge slowly taking more than 10 hours, but with mAP achieving only $\approx 84\%$ even with the whole database used. For the deep quantization method DVSQ-ALL, it also converges much slower than the proposed SPDAQ model and achieves a much lower precision than ours.

Variants of SPDAQ

We also evaluated the performance of some variants of SPDAQ using CIFAR-10 and NUS-WIDE-21. In particular, SPDAQ-nQ learns the embeddings without quantization and can be considered as the precision upper bound for SPDAQ. Also, we implement SPDAQ-LD as defined in Eq. (5). SPDAQ-LL is another version that learns quantization with only the input image subset, defined as

$$\min \sum_{i=1}^{M}\sum_{j=1}^{N} \|\mathcal{F}(\mathbf{x}_i)^T\mathbf{z}_j - \gamma \cdot \mathbf{S}_{ij}\|_2^2 + \beta \sum_{i=1}^{M} \|\mathcal{F}(\mathbf{x}_i) - \sum_{p=1}^{D} \mathbf{C}^p \mathbf{b}_i^p\|_2^2.$$

SPDAQ-nC denotes SPDAQ without the classification term. The empirical results are tabulated in Table 3.

We observe that: (1) SPDAQ and SPDAQ-LD achieve similar performance as these two models are equivalent in principle. However, SPDAQ-LD has a hyper-parameter η (Eq. (5)) which critically influences the performance but is time-consuming to tune (e.g., using grid search). This time-consuming parameter tuning process makes SPDAQ-LD undesirable. (2) We find that the performance of SPDAQ-LL severely drops compared to SPDAQ. Basically, the codebooks learned based on only the image subset fail in quantizing well the whole database, which may verify the claim that SPDAQ is able to map the subset items and database items two different distributions. (3) SPDAQ-nQ shows the upper bound of our method. We find that the performance of SPDAQ is approaching that of SPDAQ-nQ as the length of binary codes increases. (4) The comparison between SPDAQ and SPDAQ-nC shows that the classification term can help to learn more discriminative embeddings and thus further improve the retrieval accuracy.

Conclusions

We propose the Similarity Preserving Deep Asymmetric Quantization (SPDAQ) model for fast image retrieval to address the high complexity problem for training the deep model with large-scale datasets. In contrast with the existing deep quantization methods, SPDAQ learns the quantization codebooks and the binary codes directly for the database items. By effectively mapping the subset and the database items into two different but correlated distributions, similarity can be better preserved. A comprehensive empirical study is conducted for performance evaluation. Experimental results based on four benchmark datasets show that our method outperforms the existing state-of-the-art models for image retrieval in terms of both accuracy and efficiency.

References

Andoni, A., and Indyk, P. 2006. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions.

In *Proceedings of Foundations of Computer Science, 2006. FOCS'06. 47th Annual IEEE Symposium on*. IEEE.

Cao, Y.; Long, M.; Wang, J.; Zhu, H.; and Wen, Q. 2016. Deep quantization network for efficient image retrieval. In *Proceedings of AAAI*, 3457–3463.

Cao, Y.; Long, M.; Wang, J.; and Liu, S. 2017. Deep visual-semantic quantization for efficient image retrieval. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*.

Chatfield, K.; Simonyan, K.; Vedaldi, A.; and Zisserman, A. 2014. Return of the devil in the details: Delving deep into convolutional nets. *arXiv preprint arXiv:1405.3531*.

Chen, J.; Cheung, W. K.; and Wang, A. 2018. Learning deep unsupervised binary codes for image retrieval. In *Proceedings of International Joint Conference on Artificial Intelligence*.

Chen, J.; Wang, A.; and Cheung, W. K. 2018. Aphash: Anchor-based probability hashing for image retrieval. In *Proceedings of IEEE ICASSP 2018*. IEEE.

Chua, T.-S.; Tang, J.; Hong, R.; Li, H.; Luo, Z.; and Zheng, Y. 2009. 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*. ACM.

Da, C.; Meng, G.; Xiang, S.; Ding, K.; Xu, S.; Yang, Q.; and Pan, C. 2018. Nonlinear asymmetric multi-valued hashing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Gong, Y., and Lazebnik, S. 2011. Iterative quantization: A procrustean approach to learning binary codes. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 817–824. IEEE.

Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; and Bengio, Y. 2014. Generative adversarial nets. In *Proceedings of Advances in Neural Information Processing Systems*.

Gray, R. 1984. Vector quantization. *IEEE ASSP Magazine*.

He, K.; Cakir, F.; Bargal, S. A.; and Sclaroff, S. 2018. Hashing as tie-aware learning to rank. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*.

Jegou, H.; Douze, M.; and Schmid, C. 2011. Product quantization for nearest neighbor search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Kang, W.-C.; Li, W.-J.; and Zhou, Z.-H. 2016. Column sampling based discrete supervised hashing. In *Proceedings of AAAI*.

Krizhevsky, A., and Hinton, G. 2009. Learning multiple layers of features from tiny images. Technical report, Citeseer.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. In *Proceedings of Advances in Neural Information Processing Systems*, 1097–1105.

Lai, H.; Pan, Y.; Liu, Y.; and Yan, S. 2015. Simultaneous feature learning and hash coding with deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 3270–3278.

Li, Q.; Sun, Z.; He, R.; and Tan, T. 2017. Deep supervised discrete hashing. In *Proceedings of Advances in Neural Information Processing Systems*, 2482–2491.

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

Lin, T.-Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; and Zitnick, C. L. 2014. Microsoft coco: Common objects in context. In *Proceedings of European Conference on Computer Vision*, 740–755. Springer.

Liu, W.; Wang, J.; Ji, R.; Jiang, Y.-G.; and Chang, S.-F. 2012. Supervised hashing with kernels. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2074–2081. IEEE.

Liu, B.; Cao, Y.; Long, M.; and Wang, J. 2018a. Deep triplet quantization. In *Proceedings of ACM Multimedia Conference*.

Liu, H.; Ji, R.; Wang, J.; and Shen, C. 2018b. Ordinal constraint binary coding for approximate nearest neighbor search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Qiu, Z.; Pan, Y.; Yao, T.; and Mei, T. 2017. Deep semantic hashing with generative adversarial networks. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM.

Shen, F.; Shen, C.; Liu, W.; and Tao Shen, H. 2015. Supervised discrete hashing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 37–45.

Shen, F.; Gao, X.; Liu, L.; Yang, Y.; and Shen, H. T. 2017. Deep asymmetric pairwise hashing. In *Proceedings of the 2017 ACM on Multimedia Conference*, 1522–1530. ACM.

Shrivastava, A., and Li, P. 2014. Asymmetric lsh for sublinear time maximum inner product search. In *Proceedings of Advances in Neural Information Processing Systems*.

Wang, J.; Zhang, T.; Sebe, N.; Shen, H. T.; et al. 2018. A survey on learning to hash. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40(4):769–790.

Yu, T.; Yuan, J.; Fang, C.; and Jin, H. 2018. Product quantization network for fast image for fast image retrieval. In *Proceedings of European Conference on Computer Vision*.

Zhang, P.; Zhang, W.; Li, W.-J.; and Guo, M. 2014. Supervised hashing with latent factor models. In *Proceedings of the 37th international ACM SIGIR Conference on Research & Development in Information Retrieval*, 173–182. ACM.

Zhang, T.; Du, C.; and Wang, J. 2014. Composite quantization for approximate nearest neighbor search. In *Proceedings of ICML*, 838–846.

Zhu, H.; Long, M.; Wang, J.; and Cao, Y. 2016. Deep hashing network for efficient similarity retrieval. In *Proceedings of AAAI*, 2415–2421.