Deep Quantization Network for Efficient Image Retrieval

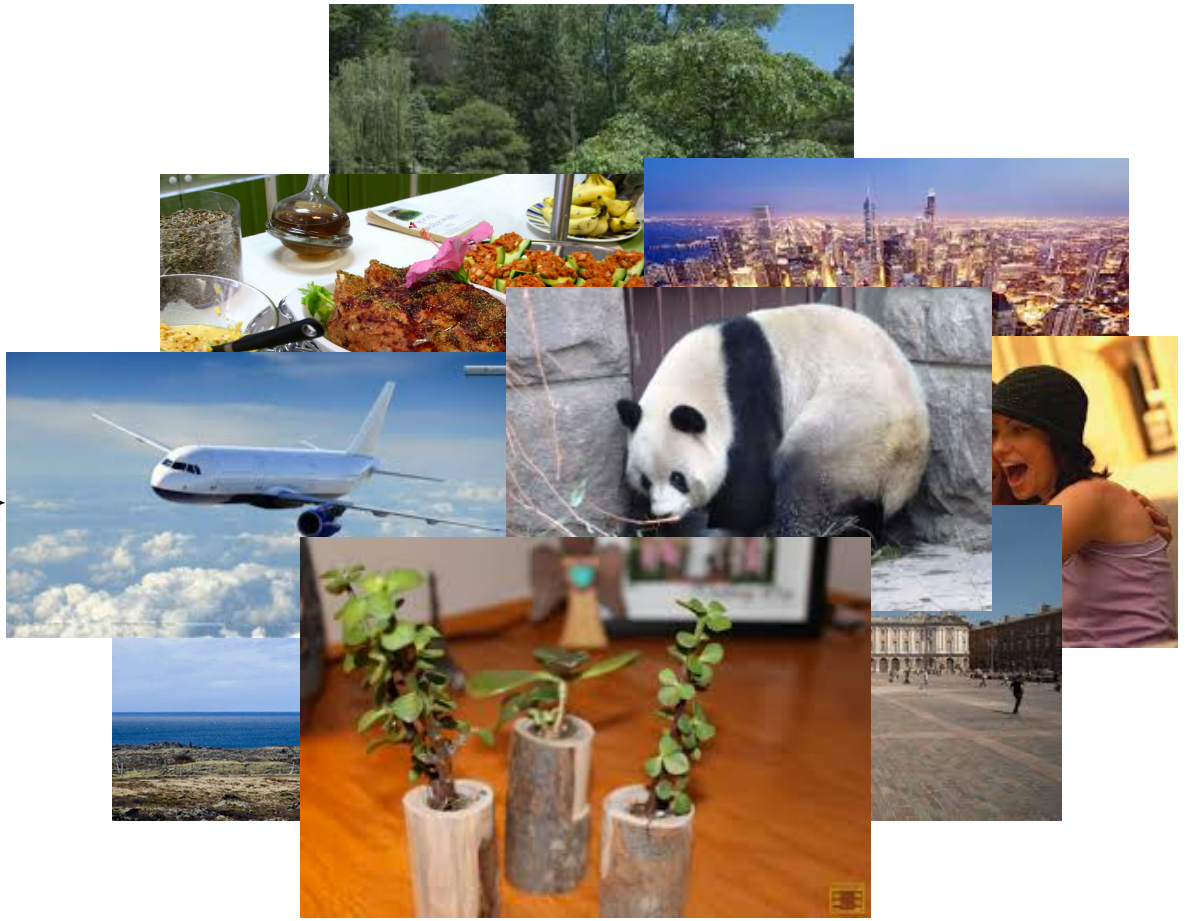
Yue Cao, Mingsheng Long, Jianmin Wang, Han Zhu, and Qingfu Wen

School of Software
Tsinghua University

The Thirtieth AAAI Conference on Artificial Intelligence, 2016

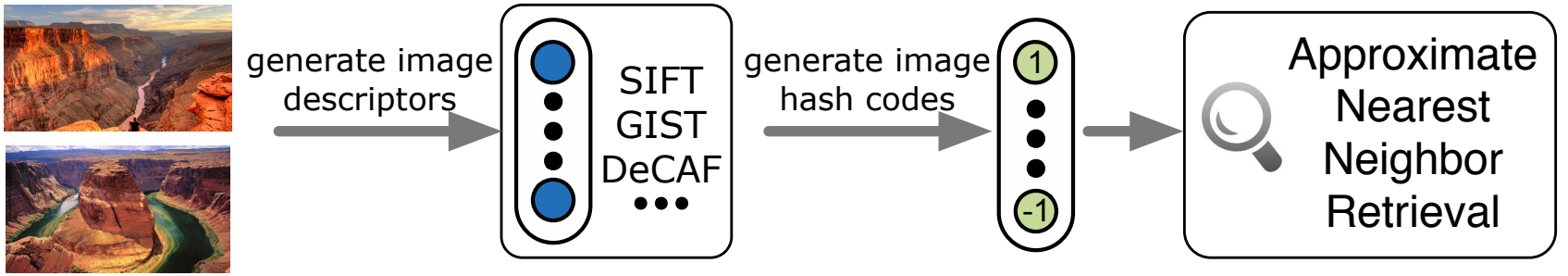
Large Scale Image Retrieval

Find Visually Similar Images



From these Images

Hashing Methods



Superiorities

Memory

- 128-d float : 512 bytes → 16 bytes
- 1 billion items : 512 GB → 16 GB

Time

- Computation: x10 - x100 faster
- Transmission (disk / web): x30 faster

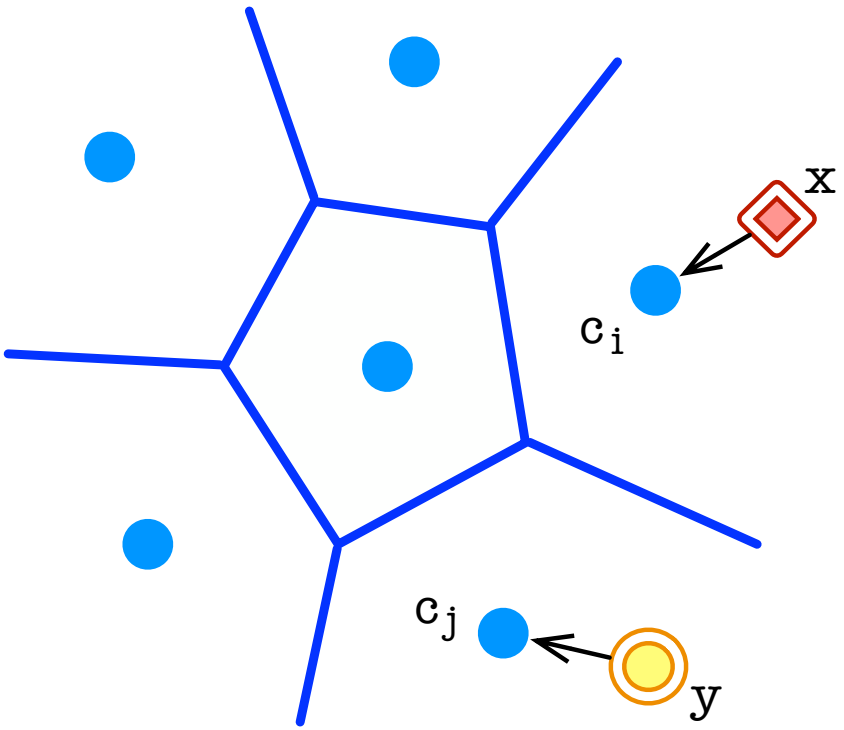
Categories

- Hamming Embedding Methods
- Quantization Methods

Applications

- Approximate nearest neighbor search
- Compact representation, Feature Compression for large datasets

Vector Quantization



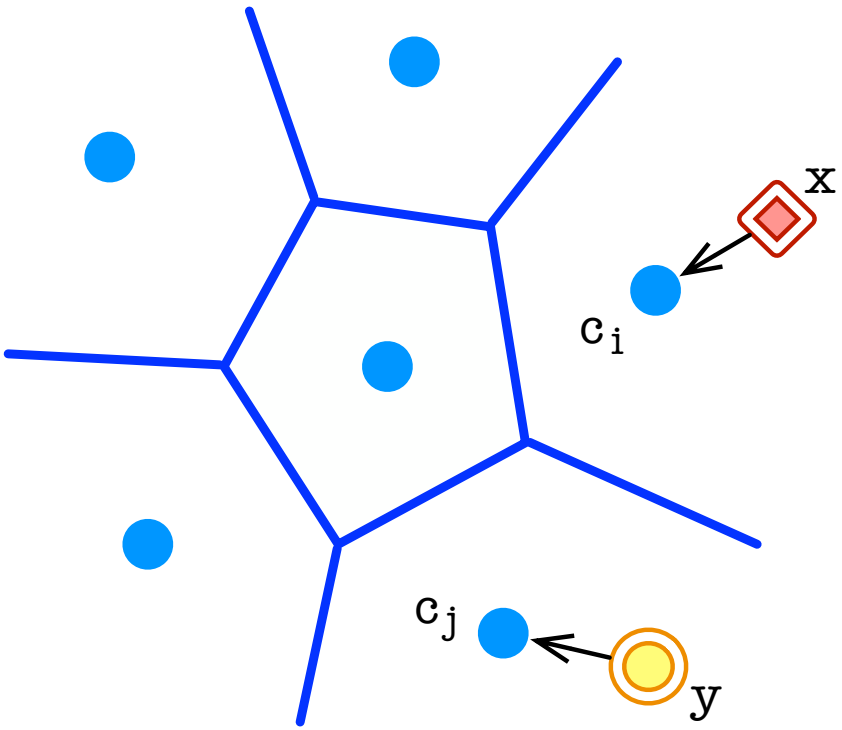
Vector Quantization

code words: K

code length: $B = \log_2 K$

$$\mathbf{x} \xrightarrow[\text{codeword}]{\text{nearest}} \mathbf{c}_i \xrightarrow[\text{stored}]{\text{code}} \mathbf{i}(\mathbf{x})$$

Vector Quantization



Vector Quantization

code words: K

code length: $B = \log_2 K$

$$\mathbf{x} \xrightarrow[\text{codeword}]{\text{nearest}} \mathbf{c}_i \xrightarrow[\text{stored}]{\text{code}} \mathbf{i}(\mathbf{x})$$

VQ for ANN Search

$$d(x, y) \approx d(c_i, c_j) \triangleq lookup(i, j)$$

construct a K-by-K (also 2^B-by-2^B) look-up table

Product Quantization (PQ) [pami 11']

Loss

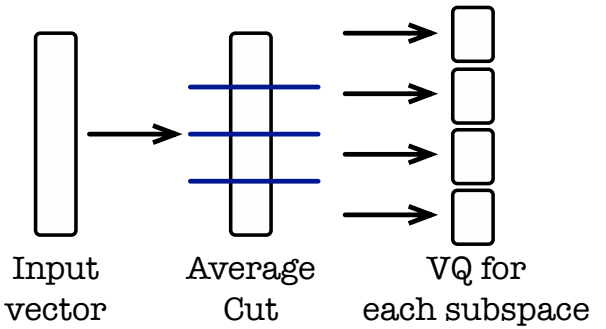
$$\min_{c_1,\ldots,c_M} \sum_{i=1}^{N} \left\| x_i - c_{i(x)} \right\|^2$$
$$\text{s.t. } c \in c_1 \times c_2 \times \ldots \times c_M$$

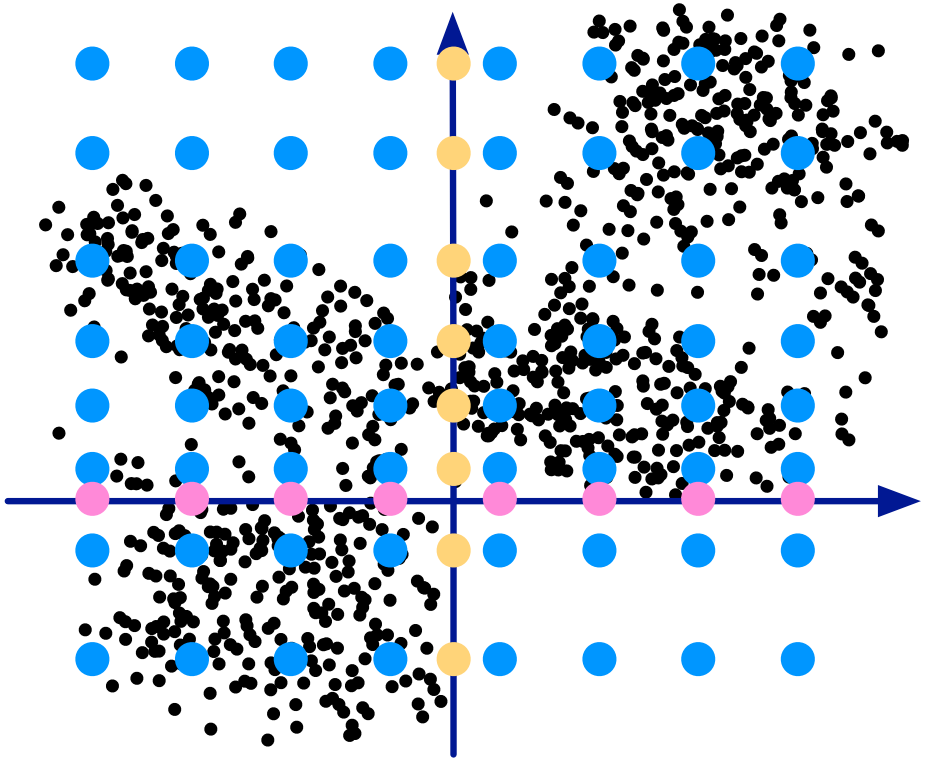
Pros

- Huge codebook: $K = k^M$
- Tractable: M k-by-k tables

Cons

- Sensitive to Projection





Optimized Product Quantization (OPQ) [cvpr 13']

Loss

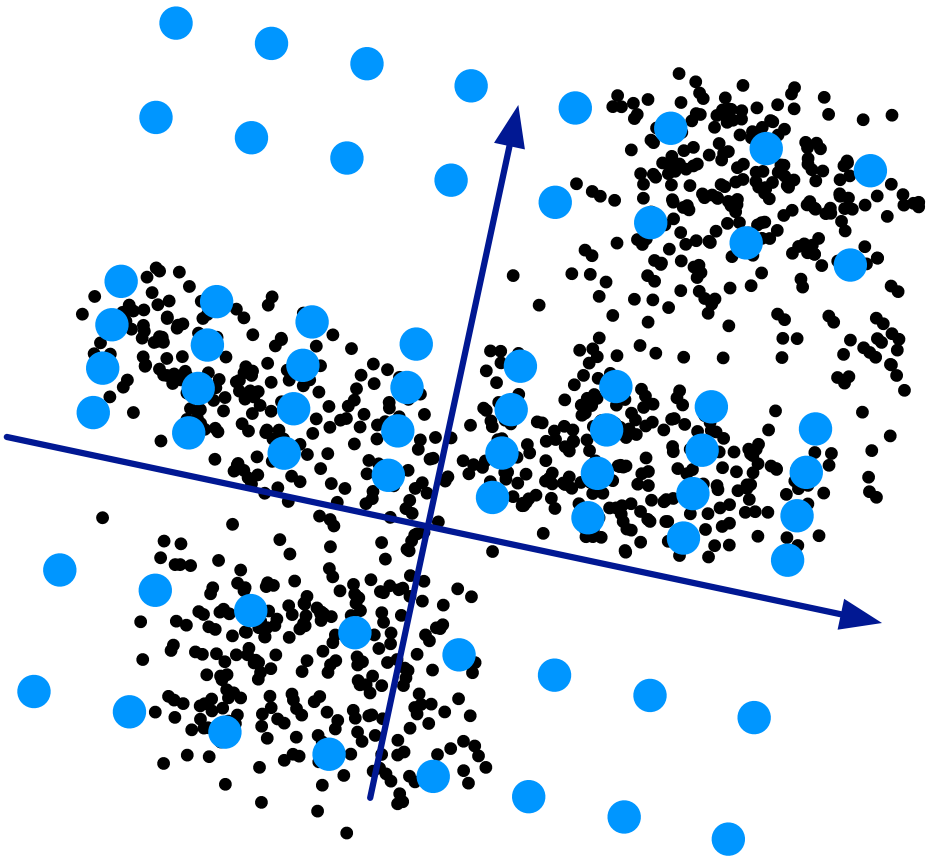
$$\min_{R,c_1,\ldots,c_M} \sum_{i=1}^{N} \left\| x_i - c_{i(x)} \right\|^2$$
$$\text{s.t. } Rc \in c_1 \times c_2 \times \ldots \times c_M,$$
$$R^T R = I$$

Pros

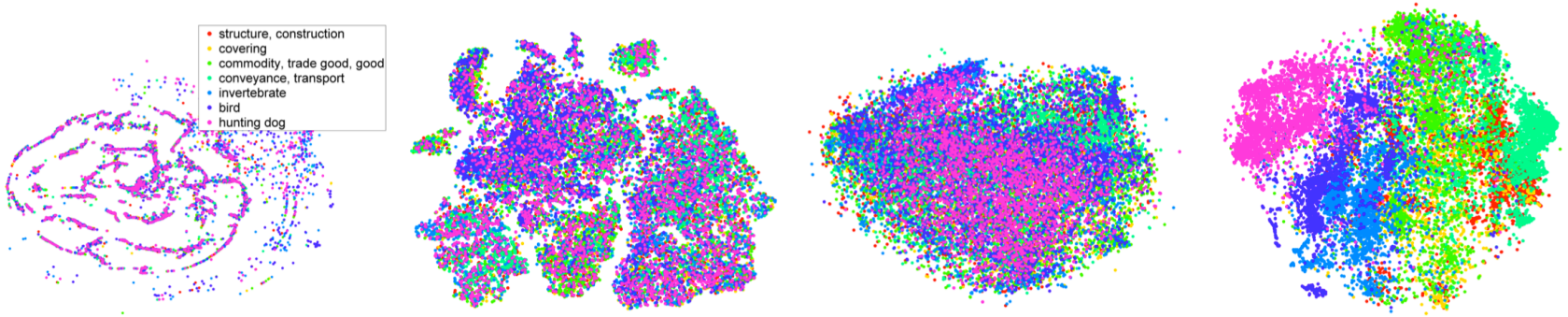
- Huge codebook: $K = k^M$
- Tractable: M k-by-k tables
- Insensitive for rotation

Cons

- high correlated between subspaces



OPQ with Deep Features



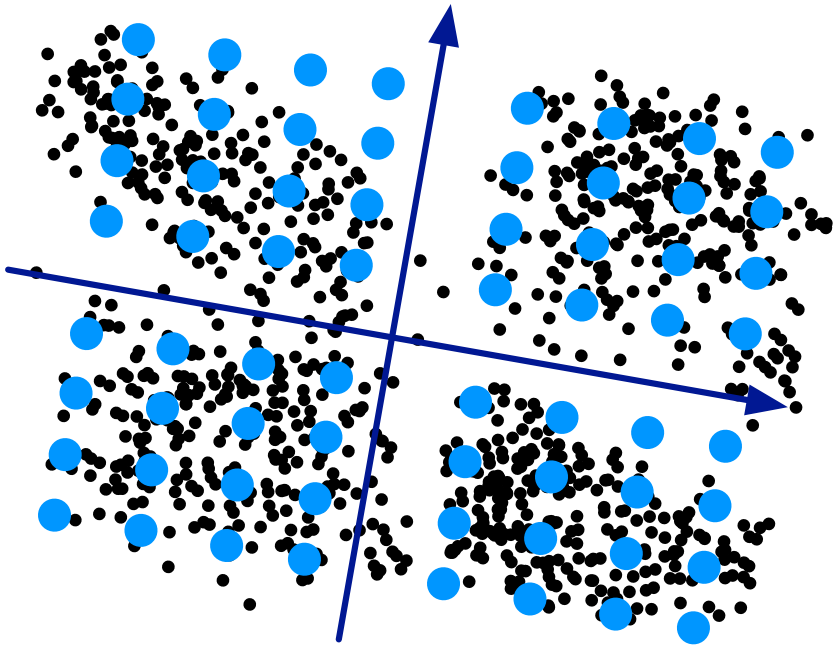
(a) LLC (b) GIST (c) DeCAF_1 (d) DeCAF_6

Pros

- Insensitive for rotation
- low correlated between subspaces

Cons

- poor quantizability: Input vector cannot be easily clustered into clusters.

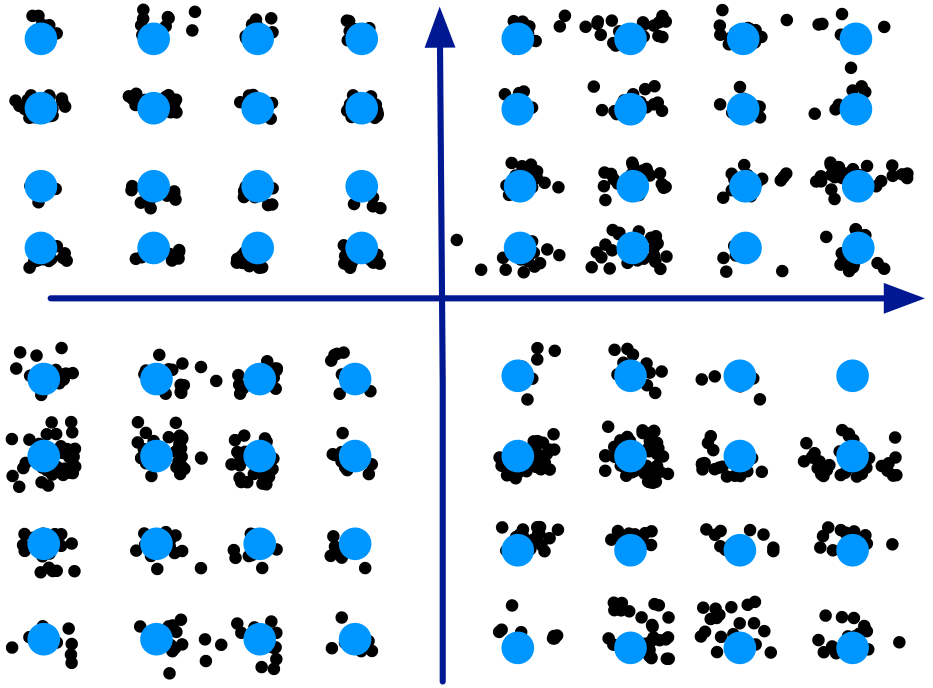


Deep Quantization

Product Quantization Loss

$$Q = \sum_{i=1}^{N} \left\| z_i^l - C h_i \right\|_2^2, \qquad (1)$$

$$C = \mathrm{diag}\left(C_1, C_2, \ldots, C_M\right) = \begin{bmatrix} C_1 & 0 & \cdots & 0 \\ 0 & C_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & C_M \end{bmatrix}.$$



Pros

- low correlated between subspaces
- easily clustered for each subspace (high quantizability)
- Look-up table is the same as PQ

Similarity Preserving

Previous works [cvpr12', aaai14']

$$L = \sum_{s_{ij} \in \mathcal{S}} \left(s_{ij} - \frac{1}{B} \langle z_i, z_j \rangle \right)^2$$

$\langle z_i, z_j \rangle \in [-R, R]$ but $s_{ij} \in \{-1, 1\}$.

Our approach

$$L = \sum_{s_{ij} \in \mathcal{S}} \left(s_{ij} - \frac{\langle z_i, z_j \rangle}{\|z_i\| \|z_j\|} \right)^2$$

$\cos(z_i, z_j) = \langle z_i, z_j \rangle / \|z_i\| \|z_j\| \in [-1, 1]$with $s_{ij} \in \{-1, 1\}$, hence making our loss well-specified for preserving the similarity conveyed in $\mathcal{S}$.

Objective Function

$$\min_{\Theta, C, H} L + \lambda Q, \tag{2}$$

Pairwise Cosine Loss

$$L = \sum_{s_{ij} \in \mathcal{S}} \left(s_{ij} - \frac{\langle z_i^l, z_j^l \rangle}{\|z_i^l\| \, \|z_j^l\|} \right)^2, \tag{3}$$

Product Quantization Loss

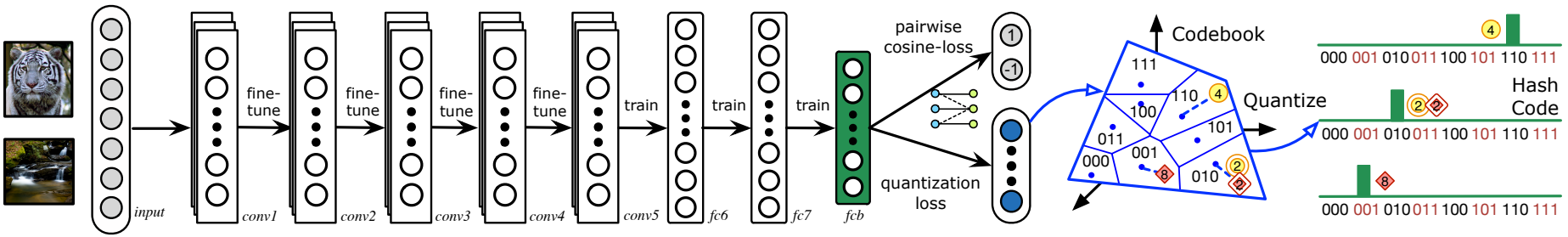
$$Q = \sum_{i=1}^{N} \left\| z_i^l - C h_i \right\|_2^2, \tag{4}$$

$$C = \operatorname{diag}(C_1, C_2, \ldots, C_M) = \begin{bmatrix} C_1 & 0 & \cdots & 0 \\ 0 & C_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & C_M \end{bmatrix}.$$

Deep Quantization Network

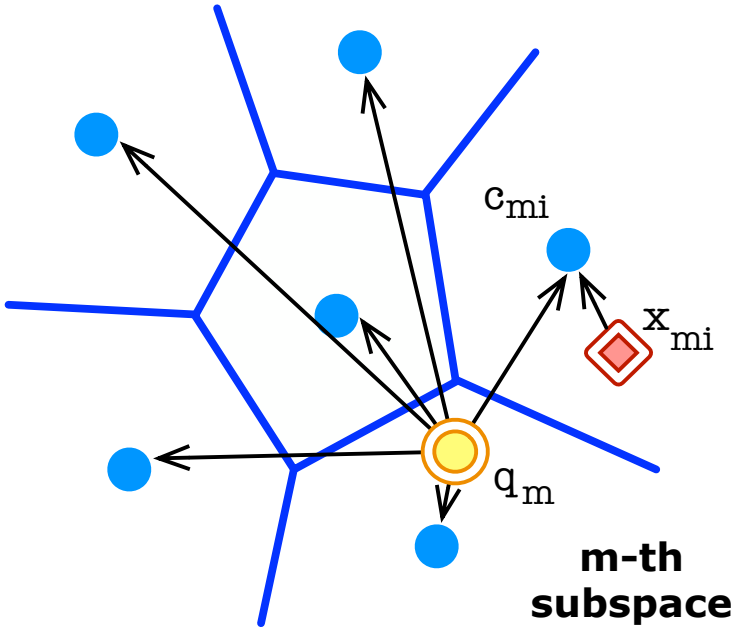
Key Contributions

- An end-to-end deep quantization framework using Alexnet for deep representation learning
- Firstly minimize quantization error with deep representation learning, which significantly improve quantizability
- Devise a pairwise cosine loss to better link the cosine distances with similarity labels



Approximate Nearest Neighbor Search

Asymmetric Quantizer Distance (AQD)



$$\mathrm{AQD}(q, x_i) = \sum_{m=1}^{M} \left\| z^l_{qm} - C_m h_{im} \right\|_2^2 \quad (5)$$

q: query, x_i: raw feature of db point i

z^l_q: deep representation of query q

h_{im}: binary code of x_i in m-th subspace.

$C_m h_{im}$: compressed representation of x_i in m-th subspace

Look-up Tables

- For each query, pre-compute $M \times K$ Look-up table
- Each query entails M table lookups and additions

Theoretical Analysis

Theorem (Error Bound)

The error of using AQD (5) *to approximate original Euclidean distance is bounded by the product quantization error* (4)

$$\left|AQD\left(q, x_i\right) - d\left(q, x_i\right)\right| \leqslant \left\|z_i^l - Ch_i\right\|_2 + |\epsilon|. \tag{6}$$

where $d\left(q, x_i\right) = \left\|z_q^l - z_i^l\right\|_2$.
The theorem can be easily proved by triangle inequality.

Insights

The error of using AQD is statistically bounded by DQN quantization loss (4), which indicates that DQN is more accurate than *sign* thresholding methods which do not control the quantization error.

Experiment Setup

- **Datasets:** pre-trained on ImageNet, fined-tuned on Nus-wide, Cifar-10 and MIRFlickr25k
- **Protocols:** MAPs, Precision-Recall Curve, Precision Top-R Curve
- **Parameter selection:** cross-validation by jointly assessing
  - test errors of joint loss function

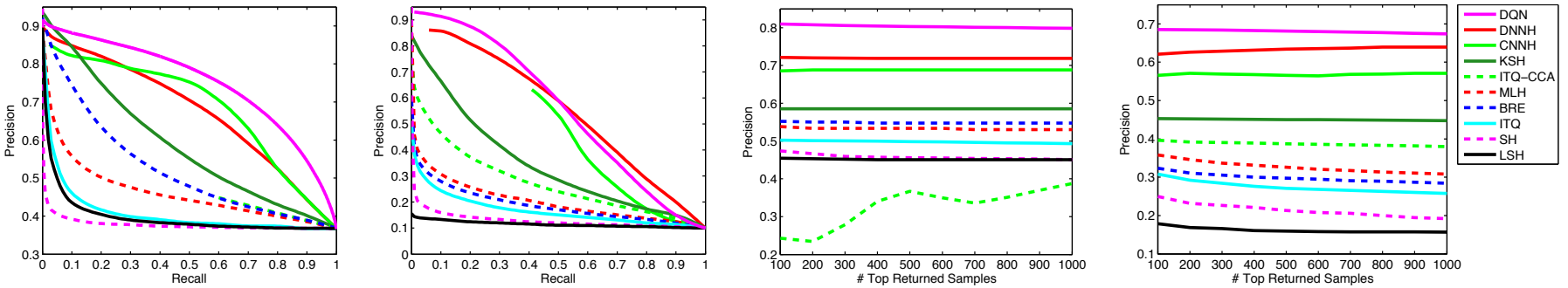
(Fei-Fei et al. 2012) (Jia et al. 2014) (Krizhevsky et al. 2009)

Results and Discussion

Learning Hash Codes by end-to-end deep hashing approach

- Product Quantization, Pairwise Cosine Loss with Alexnet (DQN)
  vs. Triplet Deep Hash with NiN structure (DNNH)
  vs. Best Shallow Hash with deep fc7 features (KSH-D)

Dataset	NUS-WIDE				CIFAR-10				Flickr			
	12 bits	24 bits	32 bits	48 bits	12 bits	24 bits	32 bits	48 bits	12 bits	24 bits	32 bits	48 bits
KSH	0.556	0.572	0.581	0.588	0.303	0.337	0.346	0.356	0.690	0.702	0.702	0.706
KSH-D	0.673	0.705	0.717	0.725	0.502	0.534	0.558	0.563	0.777	0.786	0.792	0.793
CNNH	0.617	0.663	0.657	0.688	0.484	0.476	0.472	0.489	0.749	0.761	0.768	0.776
DNNH	0.674	0.697	0.713	0.715	0.552	**0.566**	0.558	**0.581**	0.783	0.789	0.791	0.802
DQN	**0.768**	**0.776**	**0.783**	**0.792**	**0.554**	0.558	**0.564**	0.580	**0.839**	**0.848**	**0.854**	**0.863**



(a) NUS-WIDE (b) CIFAR-10 (c) NUS-WIDE (d) CIFAR-10

Empirical Analysis

DQN_{2+}: two step method, Similarity Preserving + OPQ
DQN_{ip}: replace pairwise cosine loss with Inner-Product loss

Dataset	NUS-WIDE				CIFAR-10				Flickr			
	12 bits	24 bits	32 bits	48 bits	12 bits	24 bits	32 bits	48 bits	12 bits	24 bits	32 bits	48 bits
DQN_{2+}	0.750	0.754	0.756	0.764	0.528	0.534	0.538	0.541	0.804	0.809	0.815	0.829
DQN_{ip}	0.623	0.646	0.655	0.673	0.506	0.513	0.519	0.529	0.748	0.756	0.759	0.775
DQN	**0.768**	**0.776**	**0.783**	**0.792**	**0.554**	**0.558**	**0.564**	**0.580**	**0.839**	**0.848**	**0.854**	**0.863**

Key Observations

- DQN outperforms DQN_{2+}, indicating product quantization error with representation learning can boost the quantizability.
- DQN outperforms DQN_{ip} by large margins, indicating the superiority of cosine similarity and the inconsistency of inner-product loss.

Summary

- A deep quantization network (DQN) for efficient image retrieval
- Three important contributions
  - An end-to-end deep quantization framework using Alexnet for deep representation learning
  - Firstly minimize quantization error with deep representation learning, which significantly improve quantizability
  - Devise a pairwise cosine loss to better link the cosine distances with similarity labels
- Open Problems
  - Inverted multi-index for Deep Quantization Networks
  - Deeper convolutional neural networks for better representation learning