Deep Supervised Hashing with Spherical Embedding

West Virginia University

Stanislav Pidhorskyi, Quinn Jones, Saeid Motiian, Donald Adjeroh, and Gianfranco Doretto

ACCV 2018 · 2–6 December 2018 · Perth Western Australia

Introduction

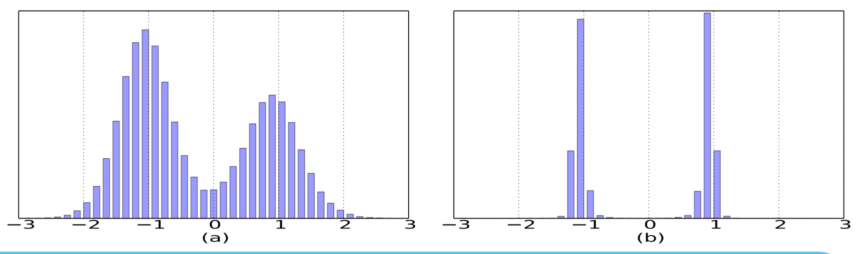
Problem: Learning mapping from images to compact binary codes preserving semantic similarity.
Why: To allow fast nearest neighbor search in large-scale image databases.
How:
- Using a novel relaxation strategy that keeps problem well-posed and does not introduce extra binarization priors.
- Encouraging balanced codes.
- Maximizing mean average precision as an objective.

Existing deep hashing techniques

Discrete nature of the codomain → NP-complete problem

Existing solutions:
- Relax the native space into the continuous counterpart, add binarization priors [1, 2, 3, 4, 5]
- Joint learning of image representations and quantization codebooks [6, 7]

Binarization priors complicate the training and might lead to performance reduction



Using conventional quantization algorithms on not constrained image representation increases the complexity of the learning process and imposes restrictions on the hash codes

Overview

Images:
$\mathcal{I} = \{I_1, \cdots, I_N\}$
Labels:
$\mathcal{Y} = \{y_1, \cdots, y_N\}$

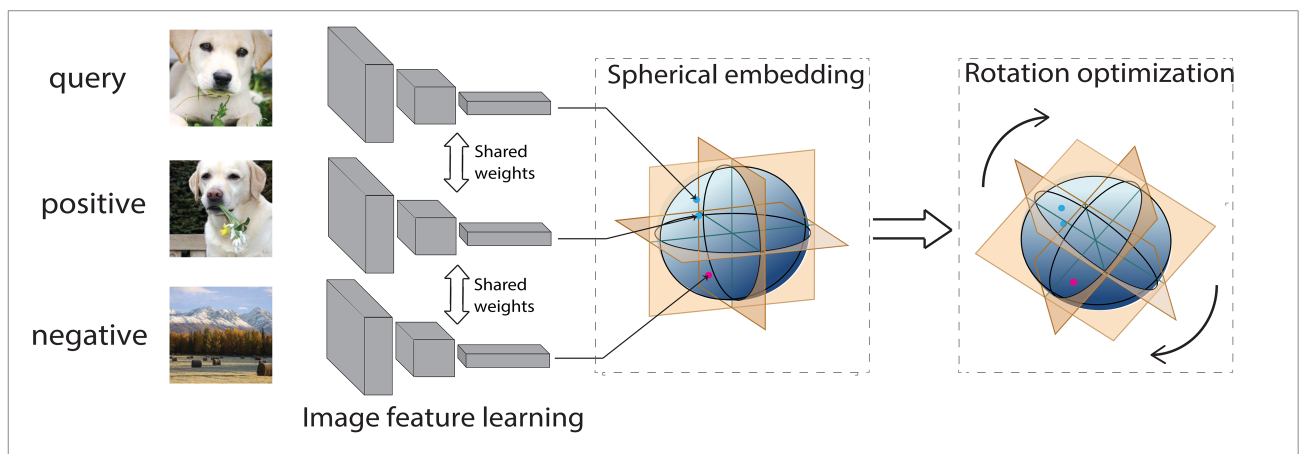
Find: $h(I)$: $\mathcal{I} \rightarrow \mathbf{b}$

Such that, for all $\mathcal{T} = \{(i,j,k) | y_i = y_j \neq y_k\}$

$d_H(\mathbf{b}_i, \mathbf{b}_j) < d_H(\mathbf{b}_i, \mathbf{b}_k) \quad \forall (i,j,k) \in \mathcal{T}$

General triplet based loss function: $L(h) = \sum_{(i,j,k)\in\mathcal{T}} \ell(d_H(\mathbf{b}_i, \mathbf{b}_j) - d_H(\mathbf{b}_i, \mathbf{b}_k))$

Our Approach



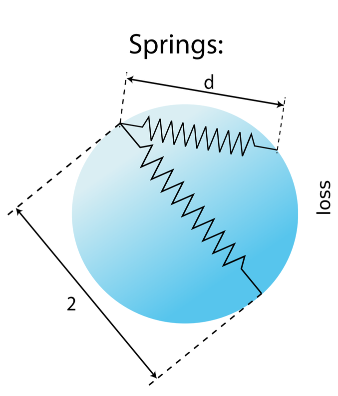
1. Learning spherical embedding → • No binary priors • Unified loss formulation • Rotation invariant • Encourages balancedness
2. Find rotation that leads to optimal → • Objective is retrieval mAP • Small number of parameters • Allows objective with no gradients available.
3. Novel triplet loss - **spring loss** → Overcomes optimization difficulties for hashes with low number of bits.

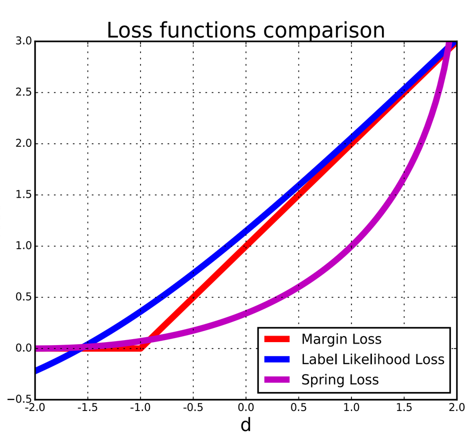
Relaxation:
$h(I) = \mathrm{sgn}[\tilde{h}(I)]$ where $\tilde{h}(I)$ is spherical embedding
constrain $\mathbf{s} = \tilde{h}(I)$ to be defined on (B-1) dimensional sphere
$\mathbf{s} \doteq \tilde{\mathbf{b}} / \|\tilde{\mathbf{b}}\|$

The objective for learning $h(I)$ becomes: $L(\tilde{h}) = \sum_{(i,j,k)\in\mathcal{T}} \ell(\mathbf{s}_i^\top \mathbf{s}_k - \mathbf{s}_i^\top \mathbf{s}_j)$

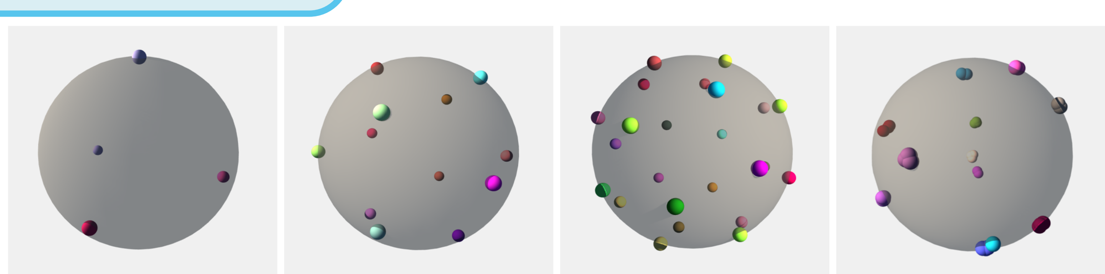
If $\tilde{h}(I)$ is optimal → so is $R\tilde{h}(I)$ → loss rotation invariant

Margin loss:
$$\ell(d_{i,j,k}) = \max\{0, d_{i,j,k} + \alpha\}$$
Label Likelihood Loss
$$\ell(d_{i,j,k}) = d_{i,j,k} + \alpha + \log(1 + e^{-d_{i,j,k} - \alpha})$$
Spring Loss
$$\ell(d_{i,j,k}) = (2 - \sqrt{2 - d_{i,j,k}})^2$$





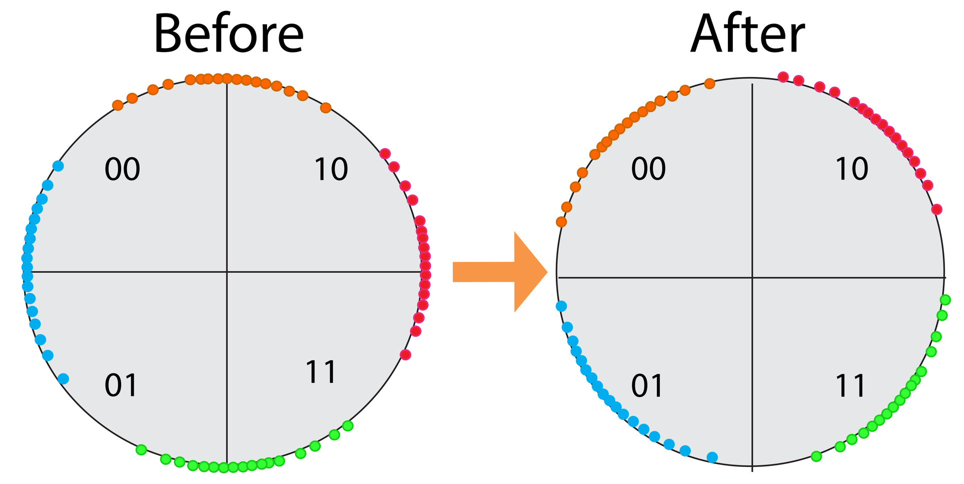
From left to right:
4 classes, spring loss
12 classes, spring loss
24 classes, spring loss
12 classes, margin loss



Quantization:
Given image I, hash code computed as: $\mathbf{b} = \mathrm{sgn}(R\tilde{h}(I))$
where R - rotation matrix

loss rotation invariant → estimating the rotation that maximizes the mAP

Rotation:
$$\hat{R} = \arg\max_R \mathrm{mAP}(R)$$
$\mathrm{mAP}(\cdot)$ is not a smooth function, gradient is zero almost everywhere.
↓
Optimization with random search



Algorithm:

1. Pick random rotation matrix $R^{(0)}$
2. Compute perturbation matrix $Q^{(i)}$
3. $R^{(i+1)} = Q^{(i)} R^{(i)}$
4. $R^{(i+1)} = \begin{cases} Q^{(i)} R^{(i)}, & \text{if } \mathrm{mAP}(R^{(i+1)}) > \mathrm{mAP}(R^{(i)}) \\ R^{(i)}, & \text{otherwise} \end{cases}$

Perturbation matrix $Q^{(i)}$

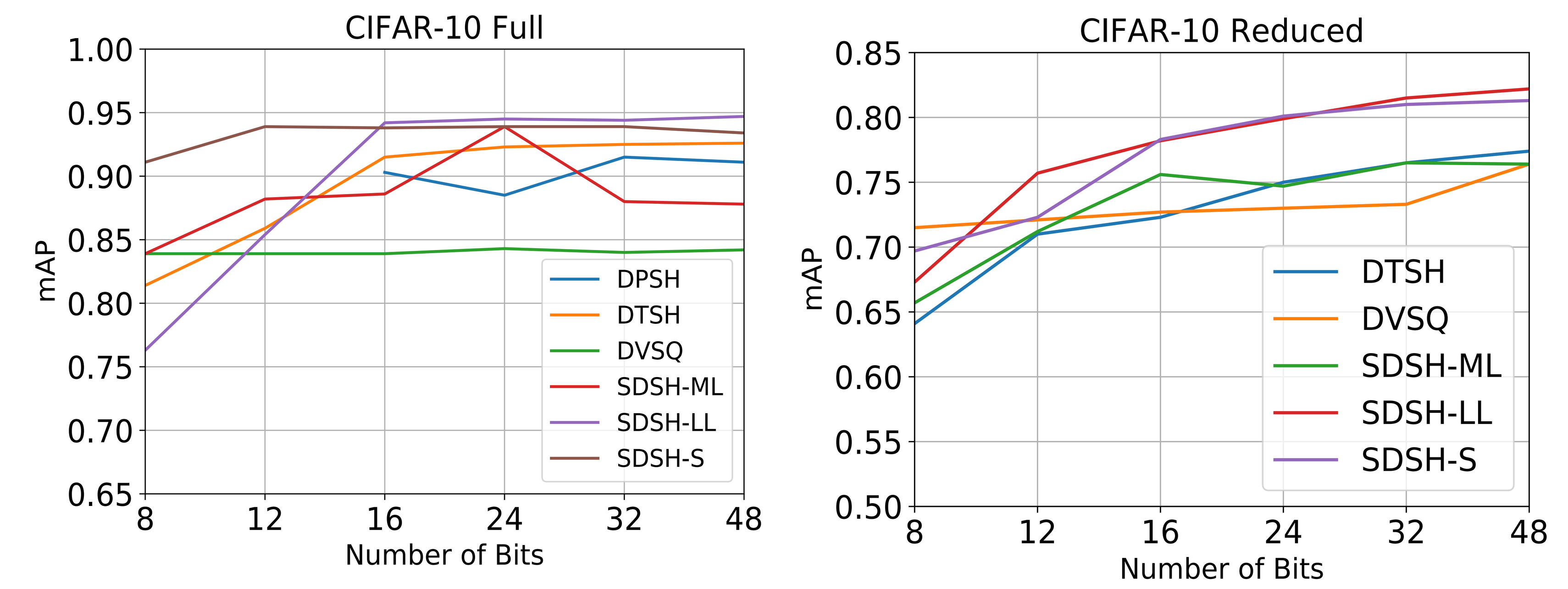
$\theta^{(i)} \leftarrow f(i)$ Rotation magnitude, linear annealing schedule

$P^{(i)} \leftarrow$ random unitary matrix Using SVD

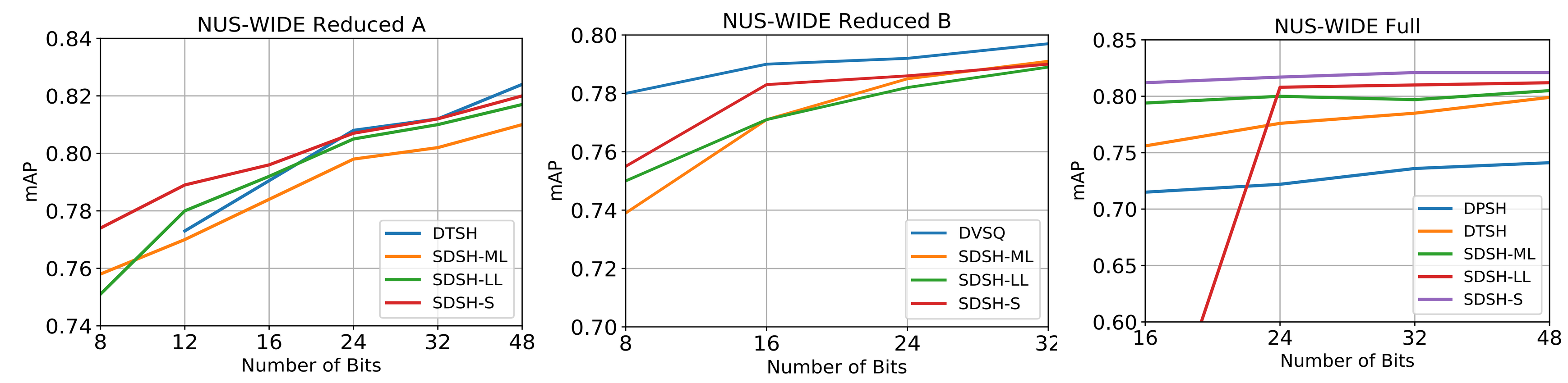
$$E(\theta) = \begin{pmatrix} \cos(\theta) & -\sin(\theta) & & & \\ \sin(\theta) & \cos(\theta) & & & \\ & & 1 & & \\ & & & \ddots & \\ & & & & 1 \end{pmatrix}$$

$Q^{(i)} = P^{(i)} E(\theta) P^{(i)\top}$ random perturbation with controllable

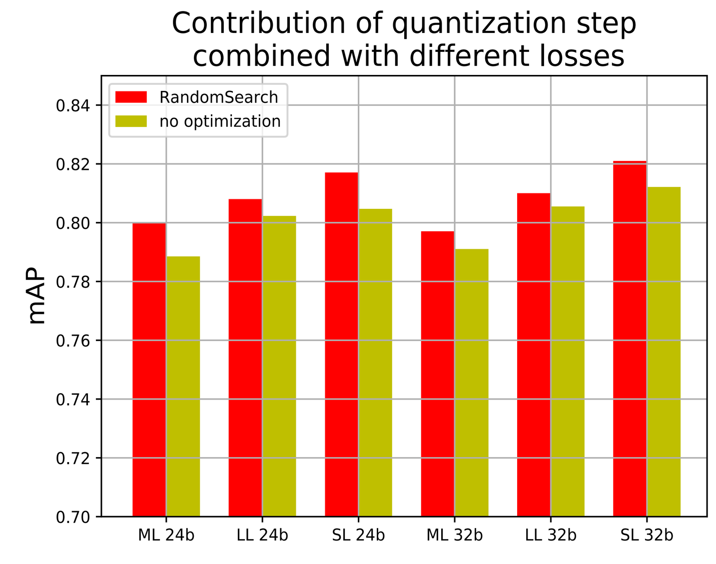
Results

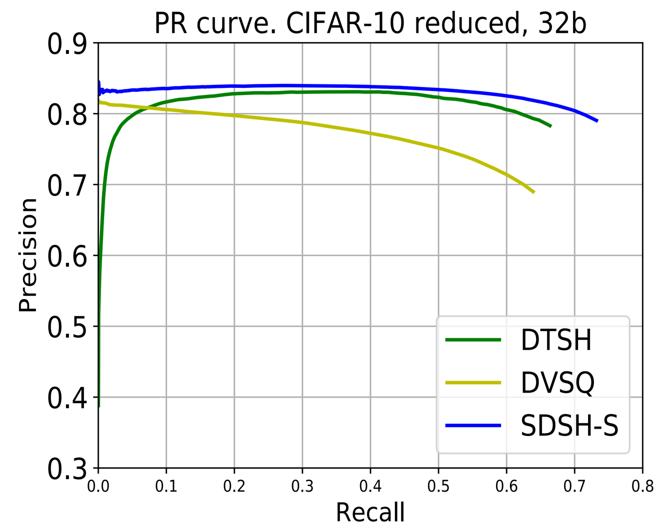


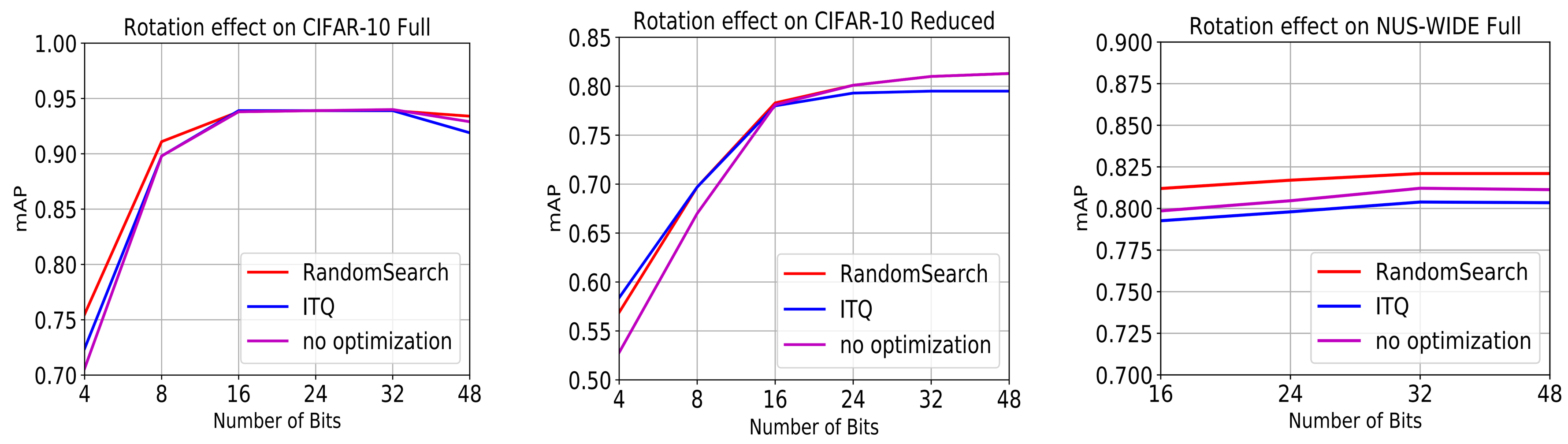
Comparison across code dimensions for CIFAR-10 dataset



Comparison across code dimension for two settings of NUS-WIDE w.r.t. the amount of samples available during training.







1. Zhu, H., Long, M.,Wang, J., Cao, Y.: Deep hashing network for efficient similarity retrieval. In: AAAI. pp. 2415–2421 (2016)
2. Liu, H., Wang, R., Shan, S., Chen, X.: Deep supervised hashing for fast image retrieval. In: CVPR. pp. 2064–2072 (2016)
3. Li, W.J., Wang, S., Kang, W.C.: Feature learning based deep supervised hashing with pairwise labels. arXiv preprint arXiv:1511.03855 (2015)
4. Wang, X., Shi, Y., Kitani, K.M.: Deep supervised hashing with triplet labels. ACCV (2016)
5. Chen, Z., Yuan, X., Lu, J., Tian, Q., Zhou, J.: Deep hashing via discrepancy minimization. In: CVPR. pp. 6838–6847 (2018)
6. Cao, Y., Long, M.,Wang, J., Zhu, H.,Wen, Q.: Deep quantization network for efficient image retrieval. In: AAAI. pp. 3457–3463 (2016)
7. Cao, Y., Long, M., Wang, J., Liu, S.: Deep visual-semantic quantization for efficient image retrieval. In: CVPR (2017)