Deep Hashing for Multimedia Retrieval

Mingsheng Long

National Engineering Laboratory for Big Data Software
School of Software
Tsinghua University

https://github.com/thuml

International Conference on Data Science, 2017

Outline

Multimedia Retrieval

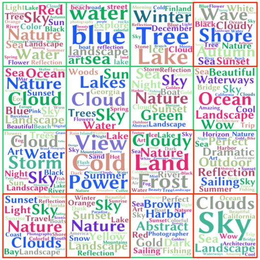
- Nearest Neighbor (NN) similarity retrieval across modalities
  - Database: $\mathcal{X}^{img} = \{\mathbf{x}_1^{img}, \ldots, \mathbf{x}_N^{img}\}$ and Query: $\mathbf{q}^{img}$, $\mathbf{q}^{txt}$
  - NN: $\mathrm{NN}\left(\mathbf{q}^{img}\right) = \min_{\mathbf{x}^{img} \in \mathcal{X}^{img}} d\left(\mathbf{x}^{img}, \mathbf{q}^{img}\right)$
  - Cross-modal NN: $\mathrm{NN}\left(\mathbf{q}^{txt}\right) = \min_{\mathbf{x}^{img} \in \mathcal{X}^{img}} d\left(\mathbf{x}^{img}, \mathbf{q}^{txt}\right)$

Image Query

['lake']

Precision: 0.625

Top 16 Returned Tags



(a) $I \rightarrow T$ (Image Query on Text DB)

Tags Query

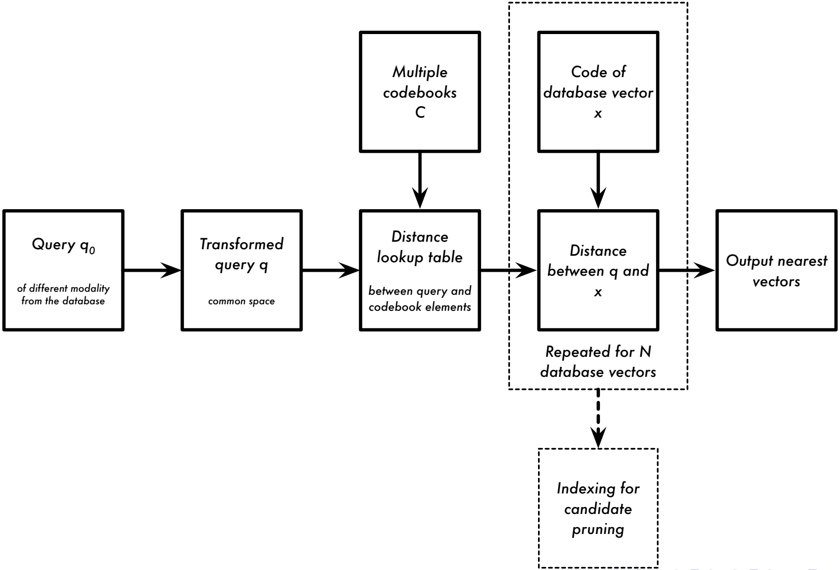
['sky sun']

Precision: 0.625

Top 16 Returned Images

(b) $T \rightarrow I$ (Text Query on Image DB)

Search Pipeline



Approximate Nearest Neighbor (ANN) Search

Exact nearest neighbor search

- Linear scan: $O(ND)$
- Costly and impractical for large scale high-dimensional cases

Approximate nearest neighbor search

- Compression
  - Reduce the cost of distance computation
  - Time complexity: $O(ND'), D' \ll D$
- Pruning
  - Reduce the number of distance computations
  - Time complexity: $O(N'D), N' \ll N$

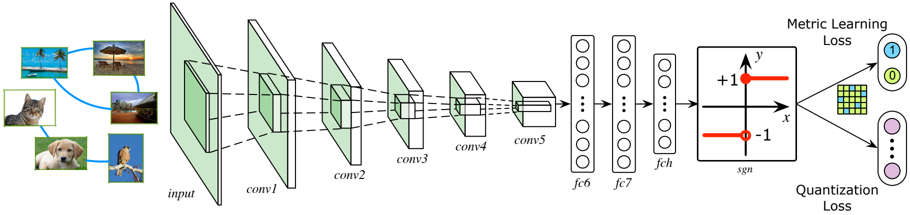
Hashing Approaches

- Hamming Embedding
  - Compression: continuous -> binary codes (1100) -> points in Hamming space
  - Pruning: Hamming Radius within 2/3/4
  - Limited ability and flexibility of distance approximation
- Quantization
  - Compression: continuous -> binary codes (1100) -> nearest centers in original space
  - Pruning: inverted index
  - More costly in distance computation, and more accurate than binary embedding

Outline

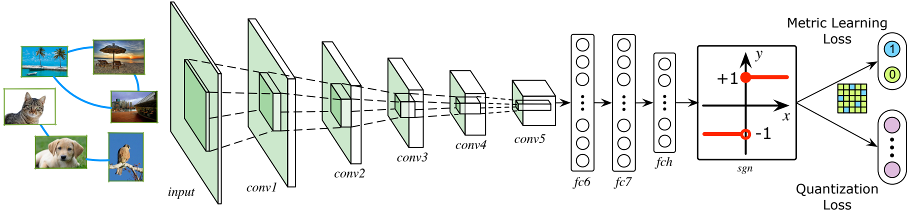
1. Introduction
2. Deep Hashing: A Bayesian Framework
3. Single-Modal Retrieval
   - Deep Quantization Network (AAAI 2016)
   - Deep Hashing Network (AAAI 2016)
   - HashNet (ICCV 2017)
4. Cross-Modal Retrieval
   - Deep Visual-Semantic Hashing (KDD 2016)

Deep Hashing



- Training data: $\mathcal{D} = \{(\mathbf{x}_i, \mathbf{x}_j, s_{ij}) : s_{ij} \in \mathcal{S}\}$ where $\mathbf{x}_i \in \mathbb{R}^D, i = 1, \ldots, N$
- $s_{ij} = 1$ if $\mathbf{x}_i$ and $\mathbf{x}_j$ are similar, $s_{ij} = 0$ if $\mathbf{x}_i$ and $\mathbf{x}_j$ are dissimilar
- Learn *nonlinear* hash function via deep network $f : \mathbf{x} \mapsto \mathbf{h} \in \{-1, 1\}^K$
- End-to-end pipeline: feature learning (metric learning) + hash coding

Maximum A Posterior (MAP) Framework

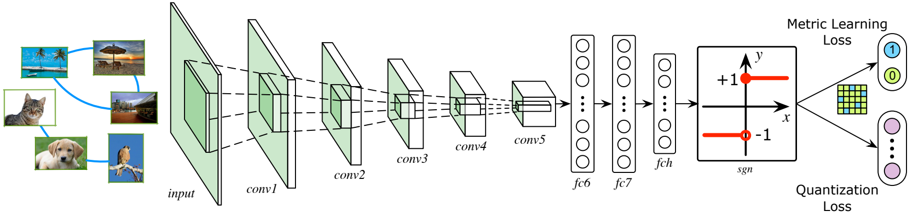


The logarithm MAP of $\mathbf{H} = [\mathbf{h}_1, \ldots, \mathbf{h}_N]$ given $\{(\mathbf{x}_i, \mathbf{x}_j, s_{ij}) : s_{ij} \in \mathcal{S}\}$ is

$$\begin{aligned} \log P(\mathbf{H}|\mathcal{S}) &\propto \log P(\mathcal{S}|\mathbf{H}) P(\mathbf{H}) \\ &= \sum_{s_{ij} \in \mathcal{S}} w_{ij} \log P(s_{ij}|\mathbf{h}_i, \mathbf{h}_j) + \sum_{i=1}^{N} \log P(\mathbf{h}_i) \end{aligned} \tag{1}$$

where $P(\mathbf{h}_i)$ is the prior, $P(\mathcal{S}|\mathbf{H}) = \prod_{s_{ij} \in \mathcal{S}} [P(s_{ij}|\mathbf{h}_i, \mathbf{h}_j)]^{w_{ij}}$ is the weighted likelihood function, and w_{ij} is the weight for each training pair $(\mathbf{x}_i, \mathbf{x}_j, s_{ij})$.

Maximum A Posterior (MAP) Framework



$P(s_{ij}|\mathbf{h}_i, \mathbf{h}_j)$ is the conditional probability of similarity label s_{ij} given a pair of hash codes $\mathbf{h}_i$ and $\mathbf{h}_j$, which can be defined by the Bernoulli distribution

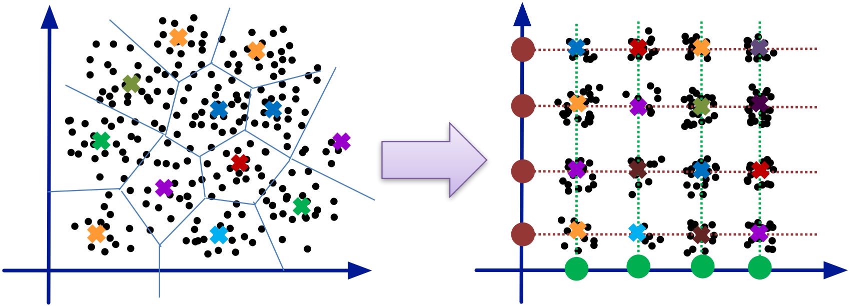
$$
\begin{aligned}
P\left(s_{ij}|\mathbf{h}_i, \mathbf{h}_j\right) &= \begin{cases} \sigma\left(\mathrm{d}\left(\mathbf{h}_i, \mathbf{h}_j\right)\right), & s_{ij} = 1 \\ 1 - \sigma\left(\mathrm{d}\left(\mathbf{h}_i, \mathbf{h}_j\right)\right), & s_{ij} = 0 \end{cases} \\
&= \sigma(\mathrm{d}\left(\mathbf{h}_i, \mathbf{h}_j\right))^{s_{ij}}\left(1 - \sigma\left(\mathrm{d}\left(\mathbf{h}_i, \mathbf{h}_j\right)\right)\right)^{1-s_{ij}}
\end{aligned} \tag{2}
$$

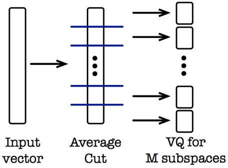
where $\mathrm{d}\left(\mathbf{h}_i, \mathbf{h}_j\right)$ is Hamming distance, and σ is specific probability function

Outline

Deep Quantization Network

Main idea: improve **quantizability** such that data can be clustered easily





$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \vdots \\ \boldsymbol{x}_M \end{bmatrix} \approx \bar{\mathrm{x}} = \begin{bmatrix} \boldsymbol{p}_{1i_1} \\ \boldsymbol{p}_{2i_2} \\ \vdots \\ \boldsymbol{p}_{Mi_M} \end{bmatrix} \leftarrow \begin{matrix} \{p_{11}, p_{12}, \cdots, p_{1K}\} \\ \{p_{21}, p_{22}, \cdots, p_{2K}\} \\ \vdots \\ \{p_{M1}, p_{M2}, \cdots, p_{MK}\} \end{matrix}$$

Codebook

Deep Quantization Network (DQN)

Quantizability: feature vectors should exhibit cluster/manifold structures

- Gaussian prior over hash representations $\mathbf{h}_i$ (continuous relaxation)

$$p(\mathbf{h}_i) = \frac{1}{\sqrt{2\pi}\gamma} \exp\left(- \left\| \mathbf{h}_i - \sum_{m=1}^{M} \mathbf{C}_m \mathbf{b}_{mi} \right\|^2 / 2\gamma^2 \right) \tag{3}$$

Likelihood function: taking inner product to quantify Hamming distance

- Based on a widely-adopted connection: $\mathrm{d}(\mathbf{h}_i, \mathbf{h}_j) = \frac{1}{2}(K - \langle \mathbf{h}_i, \mathbf{h}_j \rangle)$

$$\begin{aligned} p(s_{ij} | \mathbf{h}_i, \mathbf{h}_j) &= \begin{cases} \sigma(\langle \mathbf{h}_i, \mathbf{h}_j \rangle), & s_{ij} = 1 \\ 1 - \sigma(\langle \mathbf{h}_i, \mathbf{h}_j \rangle), & s_{ij} = 0 \end{cases} \\ &= \sigma(\langle \mathbf{h}_i, \mathbf{h}_j \rangle)^{s_{ij}} (1 - \sigma(\langle \mathbf{h}_i, \mathbf{h}_j \rangle))^{1 - s_{ij}} \end{aligned} \tag{4}$$

where $\sigma(x) = \frac{1}{1+e^{-x}}$ is the sigmoid function to turn logit into probability

Deep Quantization Network (DQN)

Overall optimization problem: Metric Learning + Product Quantization

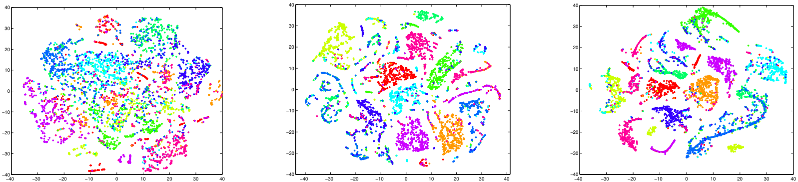
$$
\begin{aligned}
&\min_{\Theta} C = L + \lambda Q \\
&L = \sum_{s_{ij} \in \mathcal{S}} \left(\log \left(1 + \exp \left(\langle \mathbf{h}_i, \mathbf{h}_j \rangle \right) \right) - s_{ij} \langle \mathbf{h}_i, \mathbf{h}_j \rangle \right) \\
&Q = \sum_{i=1}^{N} \left\| \mathbf{h}_i - \sum_{m=1}^{M} \mathbf{C}_m \mathbf{b}_{mi} \right\|^2
\end{aligned}
\tag{5}
$$

Deep Quantization Network (DQN)

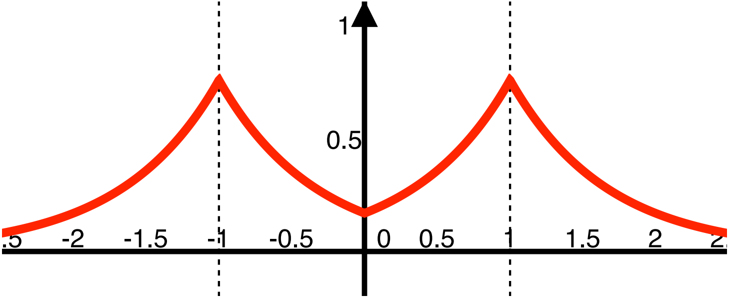
Experimental results and ablation study: end-to-end quantizability is key

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**

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	0.755	0.763	0.764	0.766	0.533	0.537	0.542	0.545	0.806	0.815	0.821	0.831
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**



Deep Hashing Network (DHN)



Bimodal Laplacian prior (unnormalized) over continuous representations $\mathbf{h}_i$

$$p(\mathbf{h}_i) = \frac{1}{2\epsilon} \exp\left(-\frac{\||\mathbf{h}_i| - \mathbf{1}\|_1}{\epsilon}\right) \tag{6}$$

The prior puts the largest probability density on the discrete values $\{-1, 1\}$.

Deep Hashing Network (DHN)

Overall optimization problem: Metric Learning + Iterative Quantization

$$\min_{\Theta} C = L + \lambda Q$$
$$L = \sum_{s_{ij} \in \mathcal{S}} \left(\log\left(1 + \exp\left(\langle \mathbf{h}_i, \mathbf{h}_j \rangle\right)\right) - s_{ij} \langle \mathbf{h}_i, \mathbf{h}_j \rangle\right)$$
$$Q = \sum_{s_{ij} \in \mathcal{S}} \left(\left\| |\mathbf{h}_i| - \mathbf{1} \right\|_1 + \left\| |\mathbf{h}_j| - \mathbf{1} \right\|_1\right) \quad (7)$$

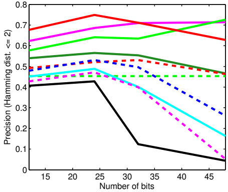
$$Q = \sum_{s_{ij} \in \mathcal{S}} \sum_{k=1}^{K} \left(\log\cosh\left(|h_{ik}| - 1\right) + \log\cosh\left(|h_{jk}| - 1\right)\right)$$

Deep Hashing Network (DHN)

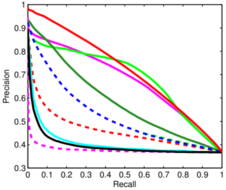
Ablation study: minimizing quantization loss improves the hashing quality

- The first work that jointly minimizes quantization loss in deep hashing

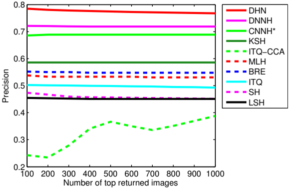
Method	NUS-WIDE (MAP)				CIFAR-10 (MAP)				Flickr (MAP)			
	12 bits	24 bits	32 bits	48 bits	12 bits	24 bits	32 bits	48 bits	12 bits	24 bits	32 bits	48 bits
DHN-B	**0.760**	**0.779**	**0.788**	**0.789**	**0.606**	**0.599**	0.597	0.592	**0.842**	**0.850**	**0.851**	**0.856**
DHN	0.708	0.735	0.748	0.758	0.555	0.594	**0.603**	**0.620**	0.810	0.828	0.829	0.841
DHN-Q	0.632	0.667	0.683	0.703	0.532	0.551	0.574	0.569	0.784	0.797	0.801	0.804
DHN-E	0.611	0.643	0.664	0.670	0.485	0.512	0.521	0.535	0.751	0.764	0.763	0.766



(c) P@H=2



(d) PR



(e) P@N

HashNet

Motivation

- Learn exactly binary hash codes by optimizing sign activation functions
- Learn discriminative metrics by tackling the data imbalance problem

We account for the data imbalance between similar and dissimilar pairs by

$$w_{ij} = c_{ij} \cdot \begin{cases} |\mathcal{S}| / |\mathcal{S}_1|, & s_{ij} = 1 \\ |\mathcal{S}| / |\mathcal{S}_0|, & s_{ij} = 0 \end{cases} \tag{8}$$

- To trade-off precision vs. recall, $\mathcal{S}_1 = \{s_{ij} \in \mathcal{S} : s_{ij} = 1\}$ is the set of similar pairs, and $\mathcal{S}_0 = \{s_{ij} \in \mathcal{S} : s_{ij} = 0\}$ is the set of dissimilar pairs,
- c_{ij} is continuous similarity, i.e. $c_{ij} = \frac{\mathbf{y}_i \cap \mathbf{y}_j}{\mathbf{y}_i \cup \mathbf{y}_j}$ if labels $\mathbf{y}_i$ and $\mathbf{y}_j$ of $\mathbf{x}_i$ and $\mathbf{x}_j$ are given, $c_{ij} = 1$ if only s_{ij} is given

HashNet

Optimization problem: Cost-Sensitive Metric Learning + Sign Activation

$$\min_{\Theta} \sum_{s_{ij} \in \mathcal{S}} w_{ij} \left(\log \left(1 + \exp \left(\alpha \left\langle \mathbf{h}_i, \mathbf{h}_j \right\rangle \right) \right) - \alpha s_{ij} \left\langle \mathbf{h}_i, \mathbf{h}_j \right\rangle \right), \tag{9}$$

$$h = \operatorname{sgn}(z) = \begin{cases} +1, & \text{if } z \geqslant 0 \\ -1, & \text{otherwise} \end{cases} \tag{10}$$

$$\lim_{\beta \to \infty} \tanh(\beta z) = \operatorname{sgn}(z) \tag{11}$$

Input: A sequence $1 = \beta_0 < \beta_1 < \ldots < \beta_m = \infty$
for stage $t = 0$ to m

- Train HashNet with $\tanh(\beta_t z)$ as activation
- Set converged HashNet as next stage initialization

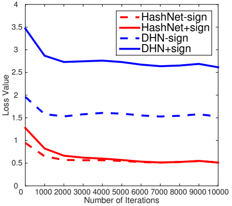
Output: HashNet with $\operatorname{sgn}(z)$ as activation, $\beta_m \to \infty$

Overall optimization problem: Metric Learning + Continuation Method

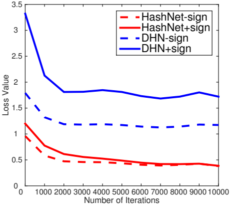
Method	ImageNet				NUS-WIDE				MS COCO			
	16 bits	32 bits	48 bits	64 bits	16 bits	32 bits	48 bits	64 bits	16 bits	32 bits	48 bits	64 bits
HashNet	**0.5059**	**0.6306**	**0.6633**	**0.6835**	**0.6623**	**0.6988**	**0.7114**	**0.7163**	**0.6873**	**0.7184**	**0.7301**	**0.7362**
DHN	0.3106	0.4717	0.5419	0.5732	0.6374	0.6637	0.6692	0.6714	0.6774	0.7013	0.6948	0.6944
DNNH	0.2903	0.4605	0.5301	0.5645	0.5976	0.6158	0.6345	0.6388	0.5932	0.6034	0.6045	0.6099
CNNH	0.2812	0.4498	0.5245	0.5538	0.5696	0.5827	0.5926	0.5996	0.5642	0.5744	0.5711	0.5671
SDH	0.2985	0.4551	0.5549	0.5852	0.4756	0.5545	0.5786	0.5812	0.5545	0.5642	0.5723	0.5799
KSH	0.1599	0.2976	0.3422	0.3943	0.3561	0.3327	0.3124	0.3368	0.5212	0.5343	0.5343	0.5361
ITQ-CCA	0.2659	0.4362	0.5479	0.5764	0.4598	0.4052	0.3732	0.3467	0.5659	0.5624	0.5297	0.5019
ITQ	0.3255	0.4620	0.5170	0.5520	0.5086	0.5425	0.5580	0.5611	0.5818	0.6243	0.6460	0.6574
BRE	0.0628	0.2525	0.3300	0.3578	0.5027	0.5290	0.5475	0.5546	0.5920	0.6224	0.6300	0.6336
SH	0.2066	0.3280	0.3951	0.4191	0.4058	0.4209	0.4211	0.4104	0.4951	0.5071	0.5099	0.5101
LSH	0.1007	0.2350	0.3121	0.3596	0.3283	0.4227	0.4333	0.5009	0.4592	0.4856	0.5440	0.5849

Method	ImageNet				NUS-WIDE				MS COCO			
	16 bits	32 bits	48 bits	64 bits	16 bits	32 bits	48 bits	64 bits	16 bits	32 bits	48 bits	64 bits
HashNet+C	**0.5059**	**0.6306**	**0.6633**	**0.6835**	**0.6646**	**0.7024**	**0.7209**	**0.7259**	**0.6876**	**0.7261**	**0.7371**	**0.7419**
HashNet	**0.5059**	**0.6306**	**0.6633**	**0.6835**	0.6623	0.6988	0.7114	0.7163	0.6873	0.7184	0.7301	0.7362
HashNet-W	0.3350	0.4852	0.5668	0.5992	0.6400	0.6638	0.6788	0.6933	0.6853	0.7174	0.7297	0.7348
HashNet-sgn	0.4249	0.5450	0.5828	0.6061	0.6603	0.6770	0.6921	0.7020	0.6449	0.6891	0.7056	0.7138

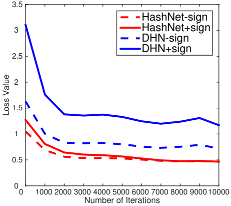
HashNet



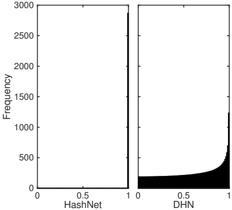
(f) ImageNet



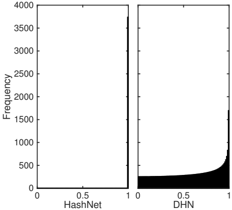
(g) NUS-WIDE



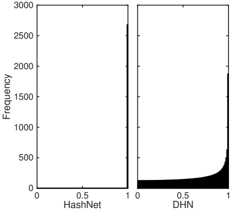
(h) COCO



(i) ImageNet

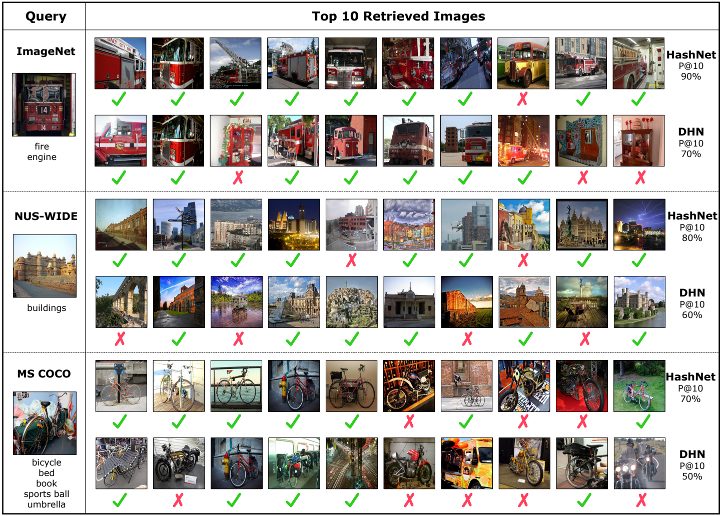


(j) NUS-WIDE



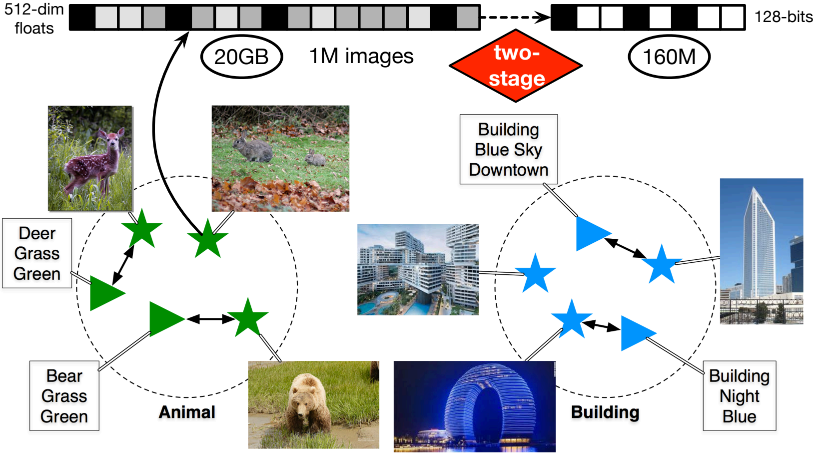
(k) COCO

HashNet


Query
Top 10 Retrieved Images
ImageNet
fire engine
HashNet P@10 90%
DHN P@10 70%
NUS-WIDE
buildings
HashNet P@10 80%
DHN P@10 60%
MS COCO
bicycle bed book sports ball umbrella
HashNet P@10 70%
DHN P@10 50%

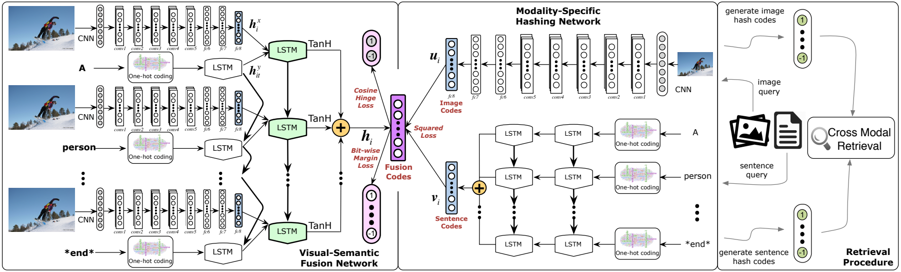
Outline

Deep Visual-Semantic Hashing



- Previous work: separate pipeline for cross-modal feature embedding and binary encoding → large information loss, unbalanced encoding

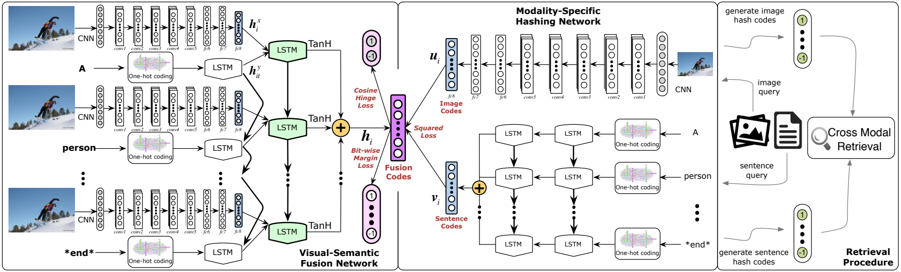
Deep Visual-Semantic Hashing



$$\mathbf{h}_{it} = f\left(\mathbf{h}_i^x + \mathbf{h}_{it}^y\right) \tag{12}$$

$$\mathbf{h}_i = \frac{\sum_{t=1}^{T} \pi_{it}\mathbf{h}_{it}}{\sum_{t=1}^{T} \pi_{it}} = \frac{\sum_{t=1}^{T} \pi_{it} f\left(\mathbf{h}_i^x + \mathbf{h}_{it}^y\right)}{\sum_{t=1}^{T} \pi_{it}} \tag{13}$$

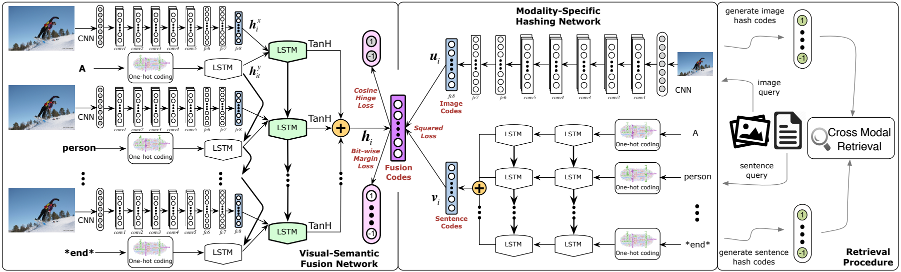
Deep Visual-Semantic Hashing



$$L = \sum_{s_{ij} \in \mathcal{S}} \max \left(0, \mu_c - s_{ij} \frac{\mathbf{h}_i \cdot \mathbf{h}_j}{\|\mathbf{h}_i\| \, \|\mathbf{h}_j\|} \right) \tag{14}$$

$$Q = \sum_{i=1}^{N} \sum_{k=1}^{K} \max \left(0, \mu_b - |h_{ik}| \right) \tag{15}$$

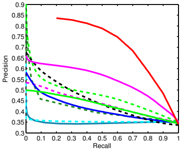
Deep Visual-Semantic Hashing



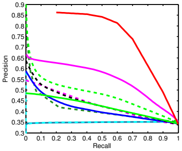
$$L^x = \frac{1}{2N} \sum_{i=1}^{N} \left(\mathbf{u}_i - \frac{\sum_{t=1}^{T} \pi_{it} \mathbf{h}_{it}}{\sum_{t=1}^{T} \pi_{it}} \right)^2 \qquad L^y = \frac{1}{2N} \sum_{i=1}^{N} \frac{\sum_{t=1}^{T} \pi_{it} (\mathbf{v}_{it} - \mathbf{h}_{it})^2}{\sum_{t=1}^{T} \pi_{it}} \tag{16}$$

$$\min_{\Theta} O = L + \lambda Q + \beta \left(L^x + L^y \right) \tag{17}$$

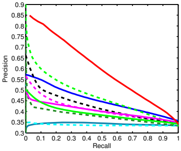
Deep Visual-Semantic Hashing



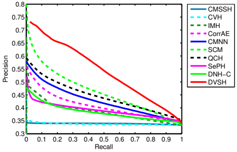
(l) $I \to T$ COCO



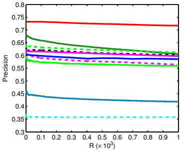
(m) $T \to I$ COCO



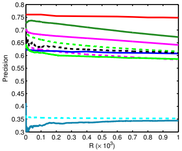
(n) $I \to T$ IAPR



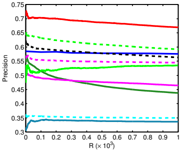
(o) $T \to I$ IAPR



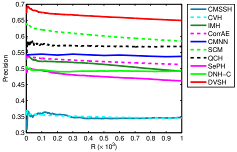
(p) $I \to T$ COCO



(q) $T \to I$ COCO



(r) $I \to T$ IAPR



(s) $T \to I$ IAPR

Deep Visual-Semantic Hashing

Task	Method	Microsoft COCO				IAPR TC-12			
		16 bits	32 bits	64 bits	128 bits	16 bits	32 bits	64 bits	128 bits
$I \to T$	SCM	0.5699	0.6002	0.6307	0.6487	**0.5880**	<u>0.6110</u>	<u>0.6282</u>	<u>0.6370</u>
	QCH	0.5723	0.5954	0.6132	0.6345	0.5259	0.5546	0.5785	0.6054
	SePH	0.5813	0.6134	0.6253	0.6339	0.5070	0.5130	0.5151	0.5309
	DNH-C	0.5353	0.5560	0.5693	0.5824	0.4801	0.5093	0.5259	0.5349
	DVSH	<u>0.5870</u>	**0.7132**	**0.7386**	**0.7552**	<u>0.5696</u>	**0.6321**	**0.6964**	**0.7236**
$T \to I$	SCM	0.5581	0.6188	0.6583	0.6858	<u>0.5876</u>	<u>0.6045</u>	<u>0.6200</u>	<u>0.6262</u>
	QCH	0.5742	0.6057	0.6375	0.6669	0.4997	0.5364	0.5652	0.5885
	SePH	**0.6127**	0.6496	0.6723	0.6929	0.4712	0.4801	0.4812	0.4955
	DNH-C	0.5250	0.5592	0.5902	0.6339	0.4692	0.4838	0.4905	0.5053
	DVSH	0.5906	**0.7365**	**0.7583**	**0.7673**	**0.6037**	**0.6395**	**0.6806**	**0.6751**

Task	Method	Microsoft COCO				IAPR TC-12			
		16 bits	32 bits	64 bits	128 bits	16 bits	32 bits	64 bits	128 bits
$I \to T$	DVSH-B	**0.6658**	**0.7408**	**0.7532**	**0.7645**	**0.6260**	**0.6761**	**0.7359**	**0.7554**
	DVSH	<u>0.5870</u>	<u>0.7132</u>	<u>0.7386</u>	<u>0.7552</u>	<u>0.5696</u>	<u>0.6321</u>	<u>0.6964</u>	<u>0.7236</u>
	DVSH-Q	0.5746	0.7019	0.7145	0.7505	0.5385	0.6113	0.6869	0.7097
	DVSH-I	0.5264	0.5745	0.6056	0.6391	0.4792	0.5035	0.5583	0.5890
	DVSH-H	0.4856	0.5244	0.5545	0.5786	0.4575	0.4975	0.5493	0.5690
$T \to I$	DVSH-B	**0.7605**	**0.8192**	**0.8034**	**0.8194**	**0.6285**	**0.6728**	**0.6922**	**0.6756**
	DVSH	<u>0.5906</u>	<u>0.7365</u>	<u>0.7583</u>	<u>0.7673</u>	<u>0.6037</u>	<u>0.6395</u>	<u>0.6806</u>	<u>0.6751</u>
	DVSH-Q	0.5530	0.7105	0.7541	0.7569	0.5684	0.6153	0.6618	0.6693
	DVSH-I	0.5185	0.5353	0.5805	0.6136	0.4903	0.5496	0.5890	0.6012
	DVSH-H	0.5025	0.5368	0.5688	0.5939	0.4396	0.4853	0.5185	0.5337