Fast Scalable Supervised Hashing

Xin Luo[1], Liqiang Nie[2], Xiangnan He[3]
Ye Wu[1], Zhen-Duo Chen[1], Xin-Shun Xu[1]

[1]Lab of **M**achine **I**ntelligence & **M**edia **A**nalysis,
School of Software, Shandong University, China
[2]School of Computer Science and Technology,Shandong University, China
[3]National University of Singapore, Singapore

Outline

- Introduction
- Proposed Method
  - Overall Objective Function
  - Optimization Algorithm
  - Out-of-Sample Extension
- Experiments
- FSSH_deep
- Conclusion & Future Work

Introduction

- Nearest Neighbor Search（NNS）

- Given a query point q, NNS returns the points closest (most similar) to q in the database.
- Underlying many machine learning, information retrieval, and computer vision problems.

Introduction

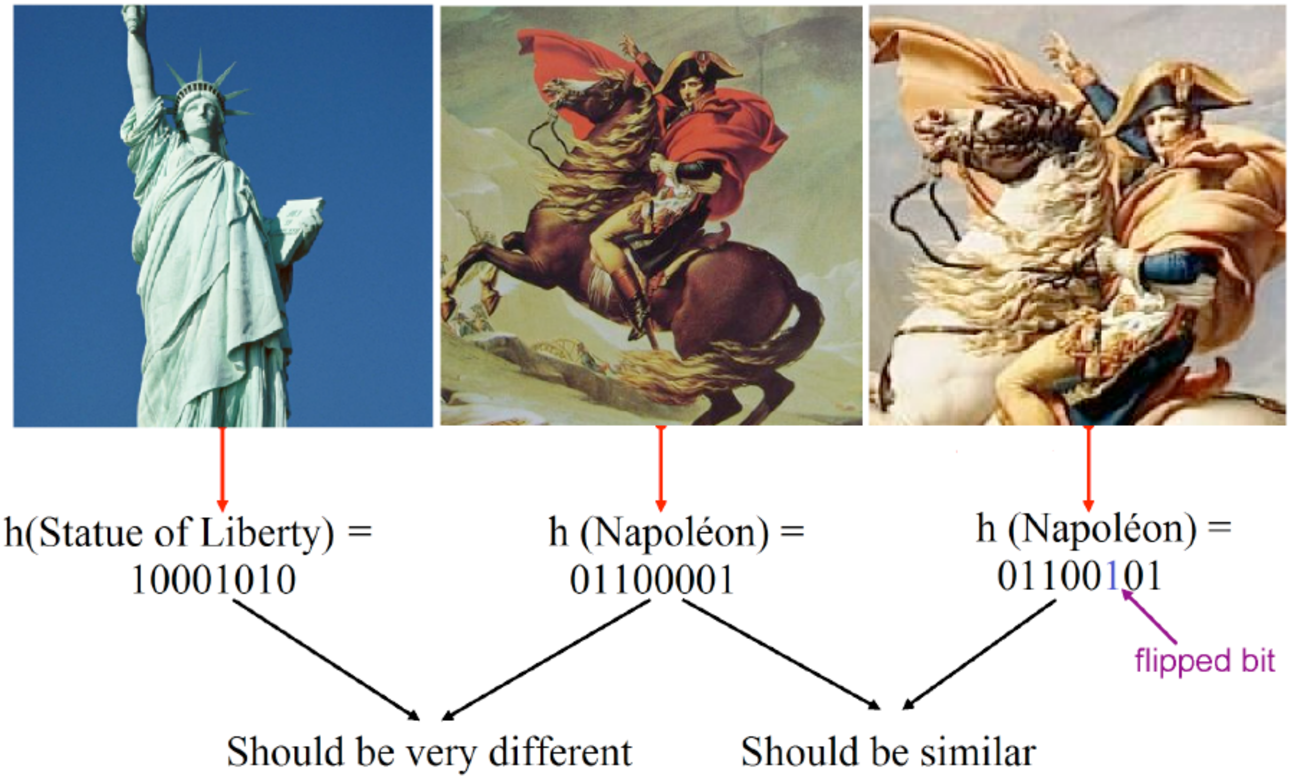
- Challenges in large-scale data applications:
  - Expensive storage cost
  - Slow query speed

- data on the Internet increases explosively
- curse of dimensionality problem

- One popular solution is the hashing based approximate nearest neighbor (ANN) technique.

Introduction

Illustration comes from http://cs.nju.edu.cn/lwj/L2H.html



Similarity Preserving

Introduction

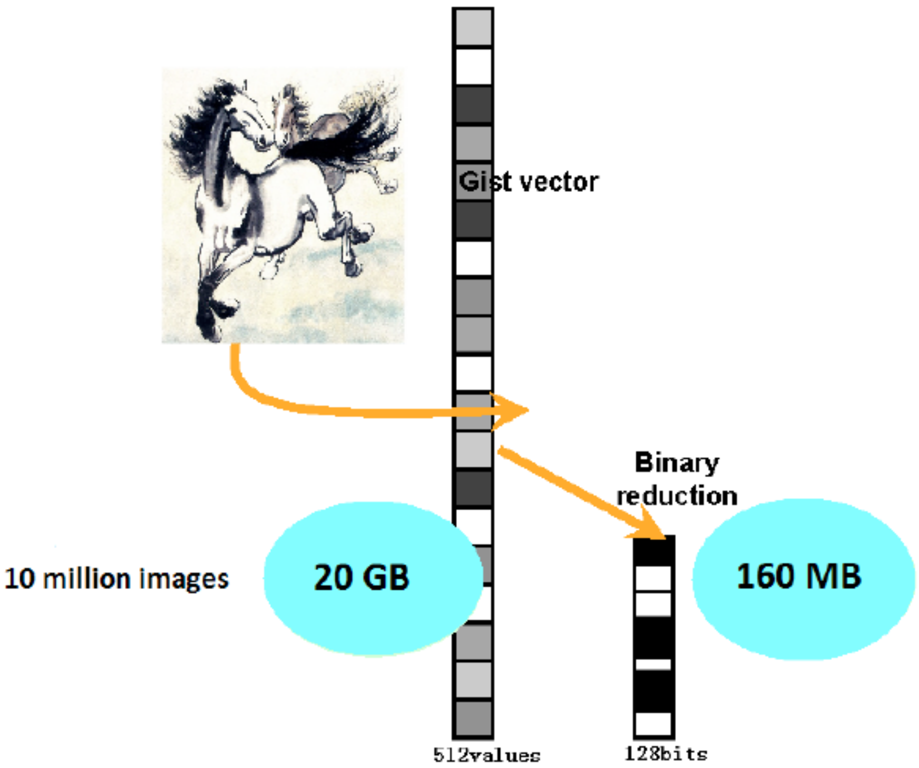
Illustration comes from http://cs.nju.edu.cn/lwj/L2H.html

- advantages of hashing:
  - ✓ fast query speed
  - ✓ low storage cost

1	0	0	0	1	0	1	0
0	1	1	0	0	0	0	1
0	1	1	0	0	1	0	1

XOR

Hamming distance



Introduction

MIMA

- According whether to use semantic information:
  - unsupervised hashing
  - supervised hashing (better retrieval accuracy)

- We propose a novel supervised hashing method, named, **F**ast **S**calable **S**upervised **H**ashing (FSSH).

Introduction

MIMA

- Two commonly used objective functions:

$$\| r\mathbf{S} - \mathbf{B}\mathbf{B}^\top \|_F^2 \quad s.t. \quad \mathbf{B} \in \{-1,1\}^{n\times r} \qquad (1)$$

$$\| \mathbf{L} - \mathbf{G}\mathbf{B} \|_F^2 \quad s.t. \quad \mathbf{B} \in \{-1,1\}^{n\times r} \qquad (2)$$

$\mathbf{S} \in \{-1,1\}^{n\times n}$	instance pairwise semantic similarity
$S_{ij} = 1$	instance i and instance j are semantically similar
$S_{ij} = -1$	i and j are semantically dissimilar
$\mathbf{B} \in \{-1,1\}^{n\times r}$	r-bit binary hash codes for n instances
$\|\cdot\|_F$	Frobenius norm

$\mathbf{L} \in \{0,1\}^{n\times c}$	labels for n instances $\mathbf{L} = [\mathbf{l}_1, \mathbf{l}_2, \cdots, \mathbf{l}_n]$
$\mathbf{l}_{ik} = 1$	instance i is in class k
$\mathbf{l}_{ik} = 0$	instance i is not in class k
$\mathbf{G}$	a projection from labels to hash codes
c	the number of classes

Introduction

MIMA

- Two commonly used objective functions:

$$\| r\mathbf{S} - \mathbf{B}\mathbf{B}^\top \|_F^2 \quad s.t. \quad \mathbf{B} \in \{-1, 1\}^{n \times r} \qquad (1)$$

$$\| \mathbf{L} - \mathbf{G}\mathbf{B} \|_F^2 \quad s.t. \quad \mathbf{B} \in \{-1, 1\}^{n \times r} \qquad (2)$$

complexity of $\mathbf{S}$ $O(n^2)$ limitations learn the hash codes bit by bit

$\mathbf{S} \in \{-1, 1\}^{n \times n}$	instance pairwise semantic similarity	$\mathbf{L} \in \{0, 1\}^{n \times c}$	labels for n instances $\mathbf{L} = [\mathbf{l}_1, \mathbf{l}_2, \cdots, \mathbf{l}_n]$
$S_{ij} = 1$	instance i and instance j are semantically similar	$\mathbf{l}_{ik} = 1$	instance i is in class k
$S_{ij} = -1$	i and j are semantically dissimilar	$\mathbf{l}_{ik} = 0$	instance i is not in class k
$\mathbf{B} \in \{-1, 1\}^{n \times r}$	r-bit binary hash codes for n instances	$\mathbf{G}$	a projection from hash codes to labels
$\|\cdot\|_F$	Frobenius norm	c	the number of classes

Proposed Method

MIMA

- Motivations:
  - How to generate hash codes fast?
  - How to make the model scalable to large-scale data?
  - How to guarantee precise hash codes?

Proposed Method

MIMA

- Motivations:
  - How to generate hash codes fast?
  - How to make the model scalable to large-scale data?
  - How to guarantee precise hash codes?

↓

- simultaneously update all bits of hash codes
- avoid the direct use of large matrix **S**
- consider both semantic and visual information

Proposed Method

MIMA

- Overall Objective Function

$$\min_{\mathbf{B},\mathbf{G},\mathbf{W}} \| \mathbf{S} - \big(\phi(\mathbf{X})\mathbf{W}\big)(\mathbf{LG})^{\top} \|_F^2 + \mu \| \mathbf{B} - \mathbf{LG} \|_F^2 + \theta \| \mathbf{B} - (\phi(\mathbf{X})\mathbf{W}) \|_F^2 \qquad (3)$$

$$s.t. \quad \mathbf{B} \in \{-1, 1\}^{n \times r}$$

Proposed Method - optimization

MIMA

- We present an iterative optimization algorithm, in which each iteration contains three steps, i.e., **W** Step, **G** Step, and **B** Step.

- More specifically,
  - **W** step: fix **G** and **B**, update **W**;
  - **G** step: fix **W** and **B**, update **G**;
  - **B** step: fix **W** and **G**, update **B**.

Proposed Method - optimization

MIMA

- **W** Step

$$\min_{\mathbf{W}} \| \mathbf{S} - \big(\phi(\mathbf{X})\mathbf{W}\big)(\mathbf{LG})^{\top} \|_F^2 + \theta \| \mathbf{B} - \phi(\mathbf{X})\mathbf{W} \|_F^2 \quad (4)$$

Proposed Method - optimization

MIMA

- **W** Step

$$\min_{\mathbf{W}} \| \mathbf{S} - \big(\phi(\mathbf{X})\mathbf{W}\big)(\mathbf{LG})^\top \|_F^2 + \theta \| \mathbf{B} - \phi(\mathbf{X})\mathbf{W} \|_F^2 \tag{4}$$

setting the derivative regarding W to zero

$$\mathbf{W} = \big(\phi(\mathbf{X})^\top \phi(\mathbf{X})\big)^{-1}\big(\phi(\mathbf{X})^\top \mathbf{SLG} + \theta\phi(\mathbf{X})^\top \mathbf{B}\big)(\mathbf{G}^\top\mathbf{L}^\top\mathbf{LG} + \theta\mathbf{I}_{r\times r})^{-1} \tag{5}$$

Proposed Method - optimization

MIMA

- **W** Step

$$\min_{\mathbf{W}} \| \mathbf{S} - \left(\phi(\mathbf{X})\mathbf{W}\right)(\mathbf{LG})^{\top} \|_F^2 + \theta \| \mathbf{B} - \phi(\mathbf{X})\mathbf{W} \|_F^2 \qquad (4)$$

setting the derivative regarding W to zero

$$\mathbf{W} = \left(\phi(\mathbf{X})^{\top}\phi(\mathbf{X})\right)^{-1}\left(\phi(\mathbf{X})^{\top}\boxed{\mathbf{S}}\mathbf{LG} + \theta\phi(\mathbf{X})^{\top}\mathbf{B}\right)\left(\mathbf{G}^{\top}\mathbf{L}^{\top}\mathbf{LG} + \theta\mathbf{I}_{r\times r}\right)^{-1} \qquad (5)$$

Proposed Method - optimization

MIMA

- **W** Step

$$\min_{\mathbf{W}} \| \mathbf{S} - \big(\phi(\mathbf{X})\mathbf{W}\big)(\mathbf{LG})^\top \|_F^2 + \theta \| \mathbf{B} - \phi(\mathbf{X})\mathbf{W} \|_F^2 \tag{4}$$

setting the derivative regarding W to zero

$$\mathbf{W} = \big(\phi(\mathbf{X})^\top \phi(\mathbf{X})\big)^{-1}\big(\phi(\mathbf{X})^\top \boxed{\mathbf{S}}\mathbf{LG} + \theta\phi(\mathbf{X})^\top \mathbf{B}\big)\big(\mathbf{G}^\top\mathbf{L}^\top\mathbf{LG} + \theta\mathbf{I}_{r\times r}\big)^{-1} \tag{5}$$

$$\mathbf{W} = \mathbf{C}^{-1}\big(\mathbf{AG} + \theta\phi(\mathbf{X})^\top\mathbf{B}\big)\big(\mathbf{G}^\top\mathbf{DG} + \theta\mathbf{I}_{r\times r}\big)^{-1} \tag{6}$$

where $\underline{\mathbf{A} = \phi(\mathbf{X})^\top\mathbf{SL}}$, $\mathbf{C} = \phi(\mathbf{X})^\top\phi(\mathbf{X})$, $\mathbf{D} = \mathbf{L}^\top\mathbf{L}$.

an intermediate term $\mathbf{A} \in \mathbb{R}^{m\times c}$

m << n, c << n
$O(mc)$

Proposed Method - optimization

MIMA

- **G** Step

$$\min_{\mathbf{G}} \| \mathbf{S} - \big(\phi(\mathbf{X})\mathbf{W}\big)(\mathbf{LG})^\top \|_F^2 + \mu \| \mathbf{B} - \mathbf{LG} \|_F^2 \qquad (7)$$

setting the derivative regarding W to zero

$$\mathbf{G} = (\mathbf{L}^\top\mathbf{L})^{-1}\big(\mu\mathbf{L}^\top\mathbf{B} + \mathbf{L}^\top\mathbf{S}^\top\phi(\mathbf{X})\mathbf{W}\big)\big(\mathbf{W}^\top\phi(\mathbf{X})^\top\phi(\mathbf{X})\mathbf{W} + \mu\mathbf{I}_{r\times r}\big)^{-1} \quad (8)$$

$$\mathbf{G} = \mathbf{D}^{-1}(\mu\mathbf{L}^\top\mathbf{B} + \mathbf{A}^\top\mathbf{W})(\mathbf{W}^\top\mathbf{C}\mathbf{W} + \mu\mathbf{I}_{r\times r})^{-1} \qquad (9)$$

where $\mathbf{A} = \phi(\mathbf{X})^\top\mathbf{SL}$, $\mathbf{C} = \phi(\mathbf{X})^\top\phi(\mathbf{X})$, $\mathbf{D} = \mathbf{L}^\top\mathbf{L}$.

Proposed Method - optimization

MIMA

- **B** Step

$$\min_{\mathbf{B}} \mu \| \mathbf{B} - \mathbf{LG} \|_F^2 + \theta \| \mathbf{B} - \phi(\mathbf{X})\mathbf{W} \|_F^2 \quad (10)$$

$$s.t. \quad \mathbf{B} \in \{-1, 1\}^{n \times r}$$

Then, we transform the above equation into,

$$= (\mu + \theta) \| \mathbf{B} \|_F^2 - 2\mu Tr(\mathbf{B}^\top \mathbf{LG})$$

$$+ \mu \| \mathbf{LG} \|_F^2 - 2\theta Tr\left(\mathbf{B}^\top \phi(\mathbf{X})\mathbf{W}\right) + \theta \| \phi(\mathbf{X})\mathbf{W} \|_F^2 \quad (11)$$

where Tr( ) is the trace norm.

Proposed Method - optimization

- **B** Step

$$\min_{\mathbf{B}} \ \mu \| \mathbf{B} - \mathbf{LG} \|_F^2 + \theta \| \mathbf{B} - \phi(\mathbf{X})\mathbf{W} \|_F^2 \tag{10}$$
$$s.t. \quad \mathbf{B} \in \{-1, 1\}^{n \times r}$$

Then, we transform the above equation into,

$$\begin{aligned} &= (\mu + \theta) \| \mathbf{B} \|_F^2 - 2\mu Tr(\mathbf{B}^\top \mathbf{LG}) \\ &+ \mu \| \mathbf{LG} \|_F^2 - 2\theta Tr\left(\mathbf{B}^\top \phi(\mathbf{X})\mathbf{W}\right) + \theta \| \phi(\mathbf{X})\mathbf{W} \|_F^2 \end{aligned} \tag{11}$$

where Tr( ) is the trace norm.

constants

Proposed Method - optimization

MIMA

- **B** Step

$$\min_{\mathbf{B}} -Tr\left(\mathbf{B}^{\top}\left(\mu\mathbf{LG} + \theta\phi(\mathbf{X})\mathbf{W}\right)\right)$$
$$s.t. \quad \mathbf{B} \in \{-1, 1\}^{n\times r} \tag{12}$$

Thus, B can also be solved with a closed-form solution stated as follows,

$$\mathbf{B} = sgn\left(\mu\mathbf{LG} + \theta\phi(\mathbf{X})\mathbf{W}\right) \tag{13}$$

Proposed Method - optimization

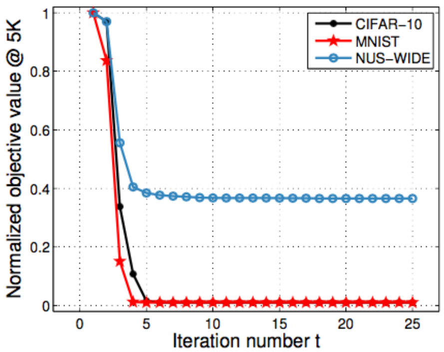
MIMA

Algorithm 1 Optimization algorithm of FSSH.

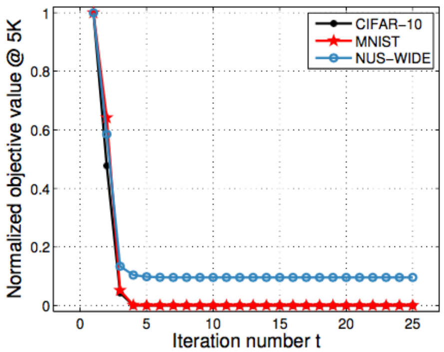
Input: The label matrix $\mathbf{Y} \in \{0,1\}^{n \times c}$, the intermediate term $\mathbf{A} = \phi(\mathbf{X})^\top \mathbf{SL}$ computed offline, the balance parameters μ and θ, iteration number t and the hash code length r.

1: Randomly initialize $\mathbf{W}$, $\mathbf{G}$ and $\mathbf{B}$.
2: **for** $iter = 1 \rightarrow t$ **do**
3:     **W step:** Fix $\mathbf{G}$ and $\mathbf{B}$, update $\mathbf{W}$ using Eq. (8).
4:     **G step:** Fix $\mathbf{W}$ and $\mathbf{B}$, update $\mathbf{G}$ using Eq. (10).
5:     **B step:** Fix $\mathbf{W}$ and $\mathbf{G}$, update $\mathbf{B}$ using Eq. (13).
6: **end for**

Output: $\mathbf{W}$ and $\mathbf{B}$.



(a) FSSH_os



(b) FSSH_ts

Proposed Method -
Out-of-Sample Extension

FSSH_os	simultaneously learns its hash functions and hash codes
FSSH_ts	uses linear regression as the hash function
FSSH_deep	adopts deep network as the hash function

Suppose $\mathbf{X}_{query}$ and $\mathbf{B}_{query}$ are the original features and corresponding hash codes of the queries.

FSSH_os

$$\mathbf{B}_{query} = sgn(\phi(\mathbf{X}_{query})\mathbf{W}) \tag{14}$$

Proposed Method - Out-of-Sample Extension

MIMA

FSSH_ts

$$\| \mathbf{B} - \phi(\mathbf{X})\mathbf{P} \|_F^2 + \lambda_e \| \mathbf{P} \|_F^2 \tag{15}$$

where λ_e is a balance parameter, $\| \mathbf{P} \|_F^2$ is the regularization term, and $\phi(\mathbf{X})$ is the RBF kernel features.

Then, the optimal **P** can be computed as,

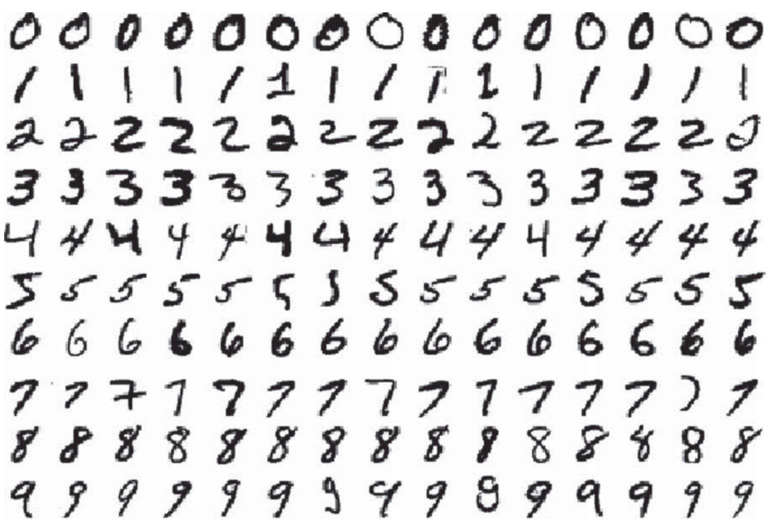
$$\mathbf{P} = \left(\phi(\mathbf{X})^\top \phi(\mathbf{X}) + \lambda_e \mathbf{I}\right)^{-1} \phi(\mathbf{X})^\top \mathbf{B} \tag{16}$$

$$\mathbf{B}_{query} = sgn(\phi(\mathbf{X}_{query})\mathbf{P}) \tag{17}$$

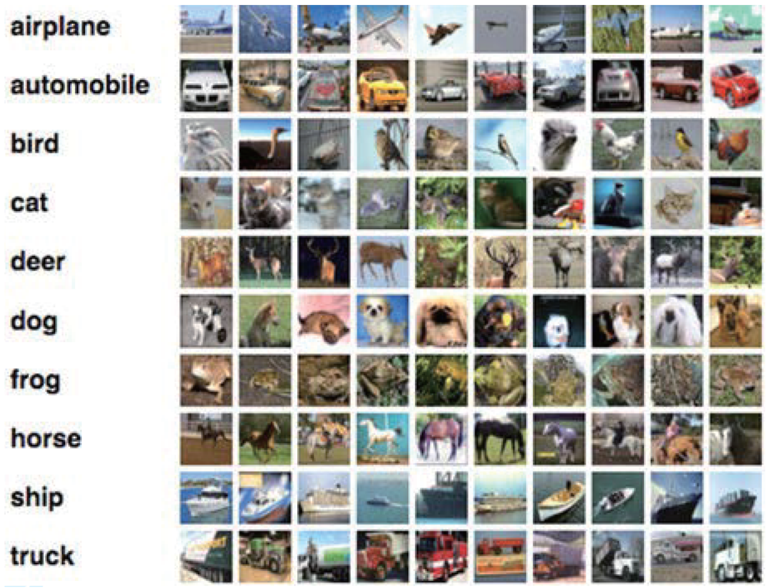
Experiments

MIMA

- Datasets
  - MNIST
  - CIFAR-10
  - NUS-WIDE



MNIST



CIFAR-10

Experiments

- Compared supervised methods
  - one-step hashing: KSH, SDH, FSDH.
  - two-step hashing: TSH, LFH, COSDISH.
- Evaluation Metrics (accuracy)
  - Mean Average Precision (MAP),
  - Top-N Precision curves,
  - Precision-Recall curves.
- Time cost (efficiency)

Experiments - MAP results

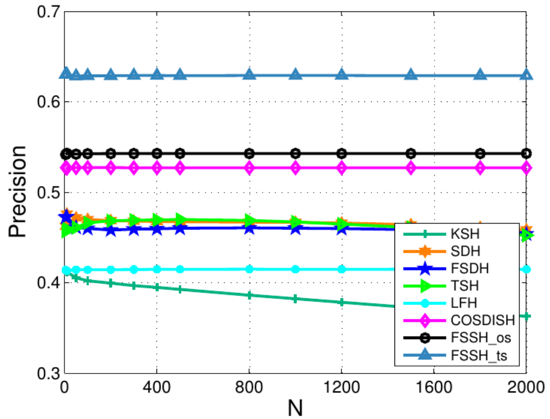
MIMA

Method	CIFAR-10				MNIST				NUS-WIDE			
	16 bits	32 bits	64 bits	96 bits	16 bits	32 bits	64 bits	96 bits	16 bits	32 bits	64 bits	96 bits
KSH	0.2626	0.2897	0.3108	0.3185	0.8017	0.8233	0.8390	0.8400	0.3703	0.3728	0.3785	0.3786
SDH	0.3525	0.3788	0.3986	0.4135	0.8718	0.8873	0.8958	0.8972	0.4113	0.4114	0.4135	0.4211
FSDH	0.3284	0.3661	0.3986	0.4110	0.8562	0.8804	0.8907	0.8962	0.4052	0.4115	0.4155	0.4166
TSH	0.3202	0.3551	0.3711	0.3843	0.9007	0.9215	0.9281	0.9323	0.4424	0.4479	0.4555	0.4545
LFH	0.3803	0.5061	0.6133	0.6288	0.5564	0.7577	0.8593	0.8575	0.5767	0.5974	0.6042	0.6124
COSDISH	0.5737	0.6109	0.6310	0.6368	0.8551	0.8754	0.8818	0.8888	0.5719	0.5913	0.5916	0.6027
FSSH_os	0.3896	0.5592	0.6497	0.6760	0.9023	0.9480	0.9360	0.9311	0.4813	0.5255	0.5867	0.6029
FSSH_ts	**0.6350**	**0.6829**	**0.7071**	**0.7108**	**0.9443**	**0.9649**	**0.9713**	**0.9721**	**0.5846**	**0.6060**	**0.6105**	**0.6225**

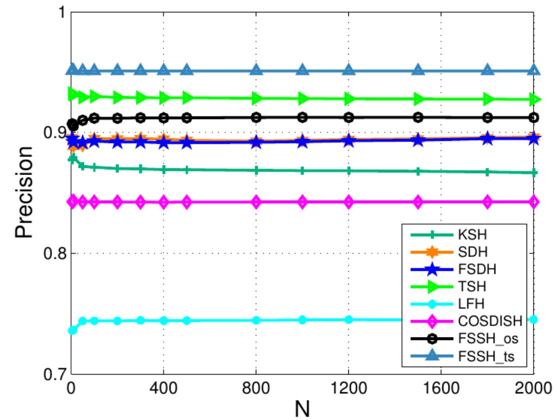
- The best MAP values of each case are shown in boldface.
- One-step hashing: KSH, SDH, FSDH, and FSSH_os.
- Two-step hashing: TSH, LFH, COSDISH, and FSSH_ts.

Experiments - curves

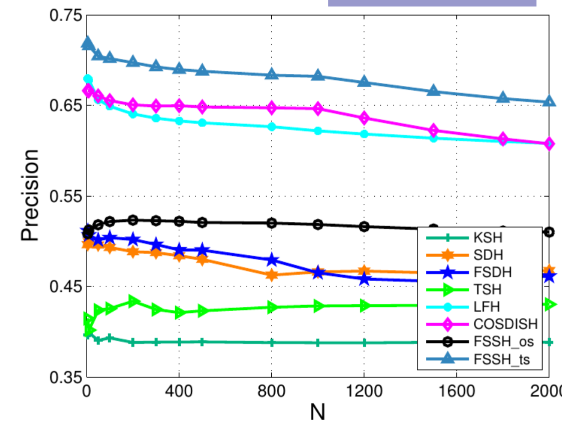
MIMA



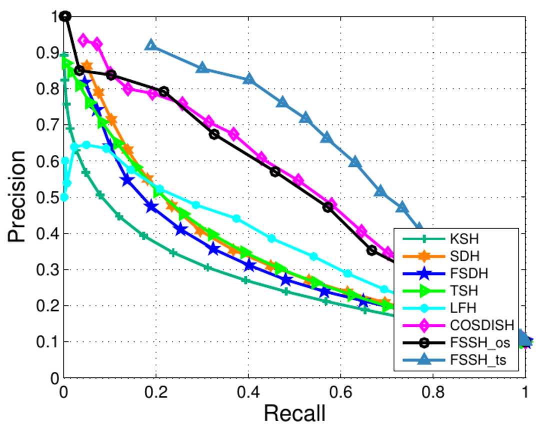
(1) CIFAR-10 top-N precision



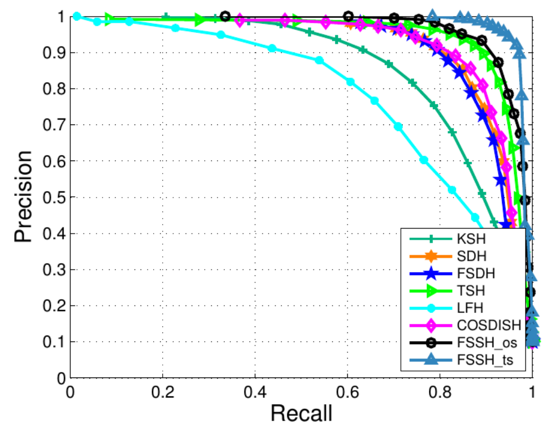
(2) MNIST top-N precision



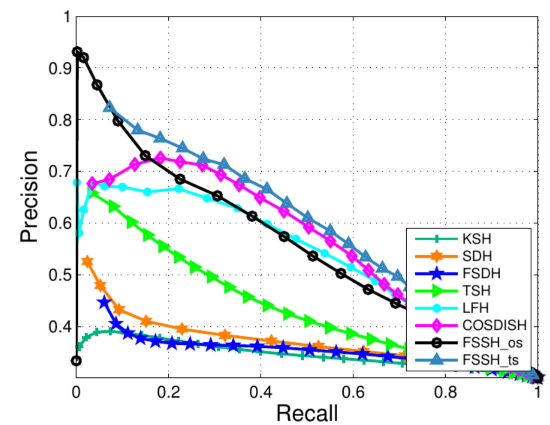
(3) NUS-WIDE top-N precision



(4) CIFAR-10 precision-recall



(5) MNIST precision-recall



(6) NUS-WIDE precision-recall

Experiments - time

MIMA

Method	CIFAR-10						MNIST					
	Training Time (second)			Test Time (millisecond)			Training Time (second)			Test Time (millisecond)		
	16 bits	32 bits	64 bits	16 bits	32 bits	64 bits	16 bits	32 bits	64 bits	16 bits	32 bits	64 bits
KSH	111.31	235.29	359.24	2.57	3.91	5.33	169.80	302.47	480.94	5.20	5.69	7.75
SDH	8.96	25.16	83.80	2.62	3.74	5.35	15.74	71.82	141.27	3.90	6.47	7.21
FSDH	6.25	6.62	6.87	2.60	3.80	5.78	9.01	9.20	9.88	4.08	5.82	7.03
TSH	105.37	206.98	340.37	2.56	4.02	6.06	173.20	343.73	483.02	4.33	5.41	7.68
LFH	4.06	7.80	13.79	2.28	3.57	5.74	7.44	12.19	18.41	2.88	4.33	5.93
COSDISH	15.99	78.99	189.59	2.45	3.89	5.54	20.88	104.08	224.01	2.75	4.19	5.65
FSSH_os	2.98	3.40	3.88	2.32	3.62	5.68	5.06	5.20	5.23	4.19	5.80	7.06
FSSH_ts	4.10	4.85	5.19	2.52	3.89	5.70	5.55	5.78	6.72	2.81	4.23	6.56

Method	NUS-WIDE (193K)			
	16 bits	32 bits	64 bits	96 bits
SDH	25.13	77.28	262.03	680.86
FSDH	17.91	19.14	20.42	26.90
LFH	16.45	19.96	28.17	37.35
COSDISH	14.06	54.22	227.01	512.14
FSSH_os	9.78	10.35	11.98	14.61
FSSH_ts	13.71	14.09	15.51	15.53

- The numbers of training images on CIFAR-10 and MNIST are 59,000 and 69,000, respectively.
- Only 2,000 samples are used for training KSH and TSH due to their large complexity.

FSSH_deep

MIMA

- FSSH_deep is one two-step variant of FSSH:
  - 1st STEP: We use features which are extracted by an off-the-shelf deep network.
  - 2nd STEP: We adopt CNN-F network as the hash function. (We train the network by solving a multi-label classi cation problem.)
- Compared deep hashing methods include DSRH, DSCH, DRSCH, DPSH, VDSH, DTSH, and DSDH.

FSSH_deep - MAP results

Method	CIFAR-10		
	24 bits	32 bits	48 bits
DSRH	0.611	0.617	618
DSCH	0.613	0.617	0.620
DRSCH	0.622	0.629	0.631
DPSH	0.781	0.795	0.807
VDSH	0.848	0.844	0.845
DTSH	0.923	0.925	0.926
DSDH	**0.940**	**0.939**	**0.939**
FSSH_deep	0.878	0.862	0.883

- All baselines are end-to-end methods.
- FSSH_deep is **not** end-to-end.
- For a fair comparison, the results of baselines are directly reported from previous works.

Conclusion & Future Work

MIMA

- We propose a novel supervised hashing method named Fast Scalable Supervised Hashing.
  - FSSH can be trained extremely fast.
  - FSSH is scalable to large-scale data.
  - FSSH generates precise hash codes.

- Three variants of FSSH are further proposed:
  - two shallow variants, i.e., FSSH_os and FSSH_ts
  - one deep variant, i.e., FSSH_deep

Conclusion & Future Work

MIMA

- Extensive experiments are conducted on three benchmark datasets. Experimental results shows the superiority of FSSH.

- In future, we plan to realize our proposed FSSH method in an end-to-end deep version to boost its performance.

Thank You !

Any Question?

Codes are available at:
https://lcbwlx.wixsite.com/fssh