Sequential Projection Learning for Hashing with Compact Codes

Jun Wang†, Sanjiv Kumar‡, Shih-Fu Chang†

† Columbia University, New York, USA

‡ Google Research, New York, USA

International Conference on Machine Learning

Haifa, Israel June 21 - 24

Background and Related Work

Nearest neighbor search for large databases

- Exhaustive nearest neighbor search not practical for large scale databases with high dimensional points
- Approximate nearest neighbor (ANN) search achieves sub-linear query time
- Popular ANN search techniques include tree-based approaches and hashing approaches

Related work

$$h_k(\mathbf{x}) = \mathrm{sgn}(f(\mathbf{w}_k^\top \mathbf{x} + b_k))$$

General form of hash functions

- Different choices of projections:
  - random projections
    locality sensitive hashing (***LSH***, Indyk et al. 98)
    Shift Invariant Kernel based Hashing (***SIKH***, Raginsky et al. 09)
  - principal projections
    spectral hashing (***SH***, Weiss, et al. 08)
- Different forms of hash functions
  - identity function: LSH & Boosted SSC (***BSSC***, Shakhnarovich, 05)
  - sinusoidal function: SH & SIKH
- Other related work
  - Restricted boltzmann machines (***RBMs***, Hinton et al. 06)
  - Jian et al. 08 (metric learning)
  - Kulis et al. 09 & Mu et al. 10 (kernerlized)

Main issues

- Existing hashing methods mostly rely on random or principal projections, which are not very compact or accurate
- Simple metrics are usually not enough to express semantic similarity

Semi-Supervised Hashing - Formulation

Setting and notations

$$h_k(\mathbf{x}) = \mathrm{sgn}(\mathbf{w}_k^\top \mathbf{x} + b_k),\ b_k = mean(\mathbf{w}_k^\top \mathbf{x})$$

In the setting of SSH, one is given the data $\mathbf{X}$, and a subset $\mathbf{X}_l \in \mathbb{R}^{D\times l}$ with pair-wise labels:

$(\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{M}$ neighbor pair
$(\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{C}$ nonneighbor-pair

Define the pairwise label matrix $\mathbf{S} \in \mathbb{B}^{l\times l}$ as:

$$S_{ij} = \begin{cases} 1 & : (\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{M} \\ -1 & : (\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{C} \\ 0 & : otherwise. \end{cases}$$

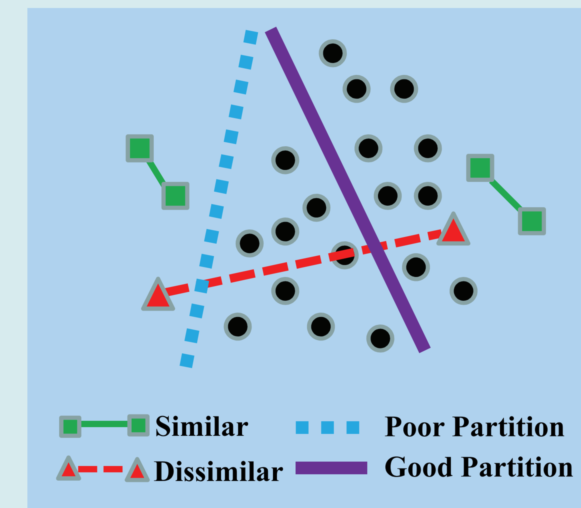
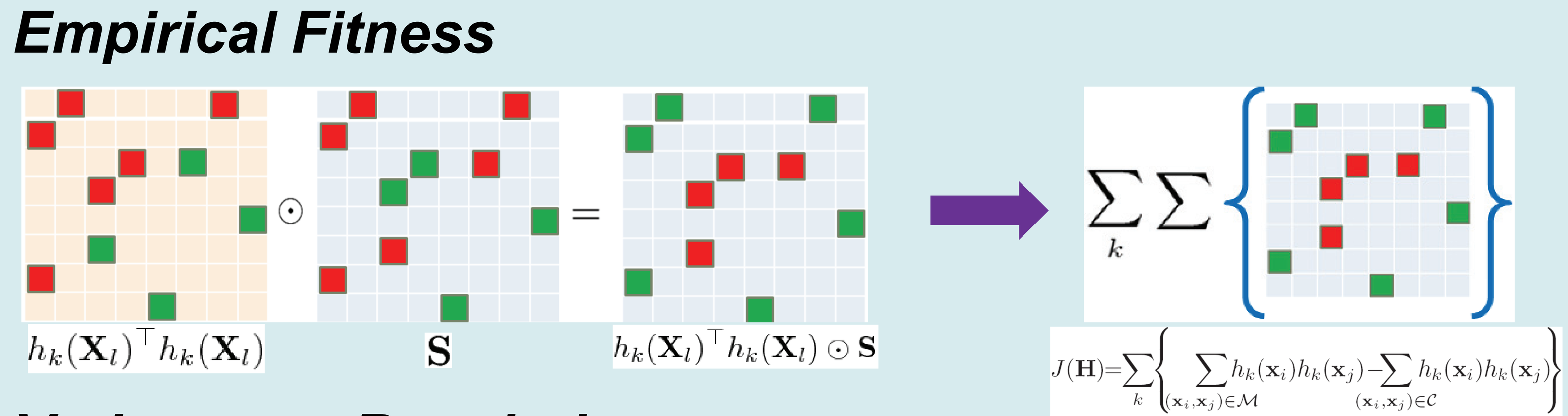


Illustration of semi-supervised hashing to achieve partitioning with maximum empirical fitness and entropy.

Empirical Fitness



$$J(\mathbf{H}) = \sum_k \left\{ \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{M}} h_k(\mathbf{x}_i)h_k(\mathbf{x}_j) - \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{C}} h_k(\mathbf{x}_i)h_k(\mathbf{x}_j) \right\}$$

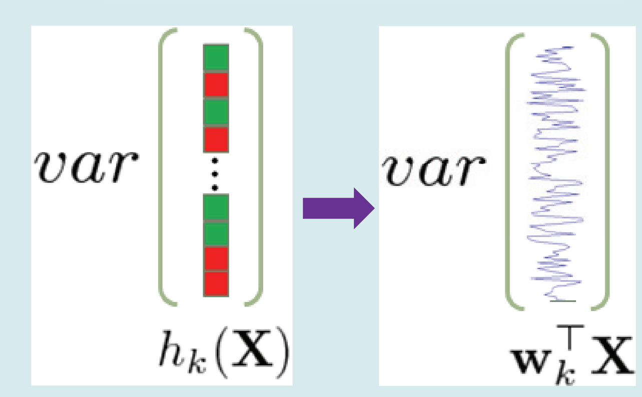
Variance as Regularizer

$$p[h_k(x)=1] = p[h_k(x)=-1] \Longleftrightarrow \max var[h_k(x)]$$

Information theoretic term: maximum variance partition equals to balanced partition

$$\max \mathrm{var}[h_k(\mathbf{x})] \geq \frac{1}{\beta}\mathrm{var}[\mathbf{w}_k^\top \mathbf{x}]$$

Lower bound of maximum bit variance



Semi-Supervised Sequential Projection Learning (S3PLH) - Solution

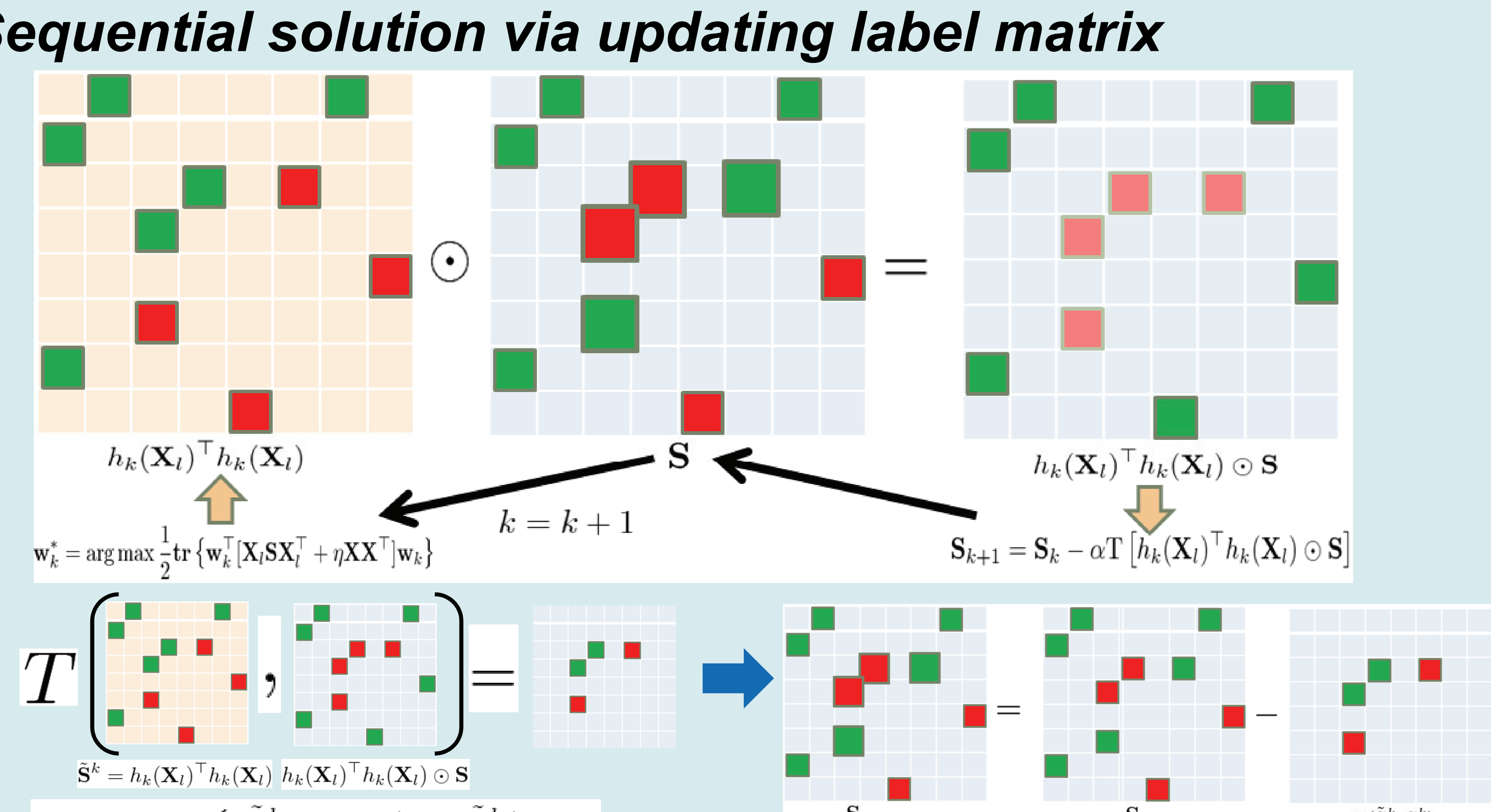
Final objective function after relaxation

$$J(\mathbf{W}) = \frac{1}{2}\mathrm{tr}\{\mathbf{W}^\top \mathbf{X}_l \mathbf{S} \mathbf{X}_l^\top \mathbf{W}\} + \frac{\eta}{2}\mathrm{tr}[\mathbf{W}^\top \mathbf{X}\mathbf{X}^\top \mathbf{W}] = \frac{1}{2}\mathrm{tr}\{\mathbf{W}^\top [\mathbf{X}_l \mathbf{S} \mathbf{X}_l^\top + \eta \mathbf{X}\mathbf{X}^\top] \mathbf{W}\}$$

$\mathbf{M} = \mathbf{X}_l \mathbf{S} \mathbf{X}_l^\top + \eta \mathbf{X}\mathbf{X}^\top$ *"adjusted" covariance matrix*

The orthogonal hash codes can be obtained by doing eigen-decomposition on **M** *- **PCAH***

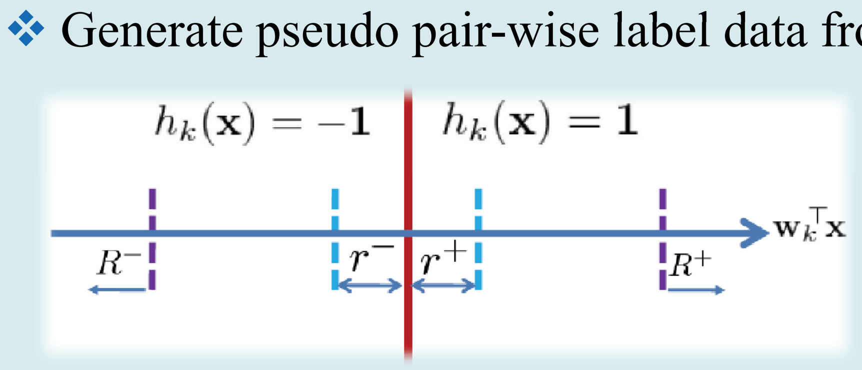
Sequential solution via updating label matrix



$$T(\tilde{\mathbf{S}}_{ij}^k, \mathbf{S}_{ij}) = \begin{cases} \tilde{\mathbf{S}}_{ij}^k & : \mathrm{sgn}(\mathbf{S}_{ij} \cdot \tilde{\mathbf{S}}_{ij}^k) < 0 \\ 0 & : \mathrm{sgn}(\mathbf{S}_{ij} \cdot \tilde{\mathbf{S}}_{ij}^k) \geq 0 \end{cases}$$

Unsupervised Sequential Projection Learning (USPLH) - Extension

- Generate pseudo pair-wise label data from existing hash bits



$\mathcal{M} = \{(x_i, x_j)\}$:
$h(x_i)\cdot h(x_j) = -1, |\mathbf{w}^\top(\mathbf{x}_i - \mathbf{x}_j)| \leq \epsilon$
$\mathcal{C} = \{(x_i, x_j)\}$:
$h(x_i)\cdot h(x_j) = 1, |\mathbf{w}^\top(\mathbf{x}_i - \mathbf{x}_j)| \geq \zeta$

- Use pseudo label matrix to learn projections

$$\mathbf{S}_{\mathcal{MC}}^k : \begin{array}{l} (\mathbf{x}_i, \mathbf{x}_i) \in \mathcal{M} \Rightarrow \mathbf{S}_{\mathcal{MC}}^k(i,j) = 1 \\ (\mathbf{x}_i, \mathbf{x}_i) \in \mathcal{C} \Rightarrow \mathbf{S}_{\mathcal{MC}}^k(i,j) = -1 \end{array}$$

Pseudo label matrix from kth hash bit

$$\mathbf{M}_k = \sum_{i=0}^{k-1} \lambda^{k-i} \mathbf{X}_{\mathcal{MC}}^i \mathbf{S}_{\mathcal{MC}}^i \mathbf{X}_{\mathcal{MC}}^{i\ \top} + \eta \mathbf{X}\mathbf{X}^\top$$

"adjusted" covariance matrix with pseudo labels

Experiment and Results

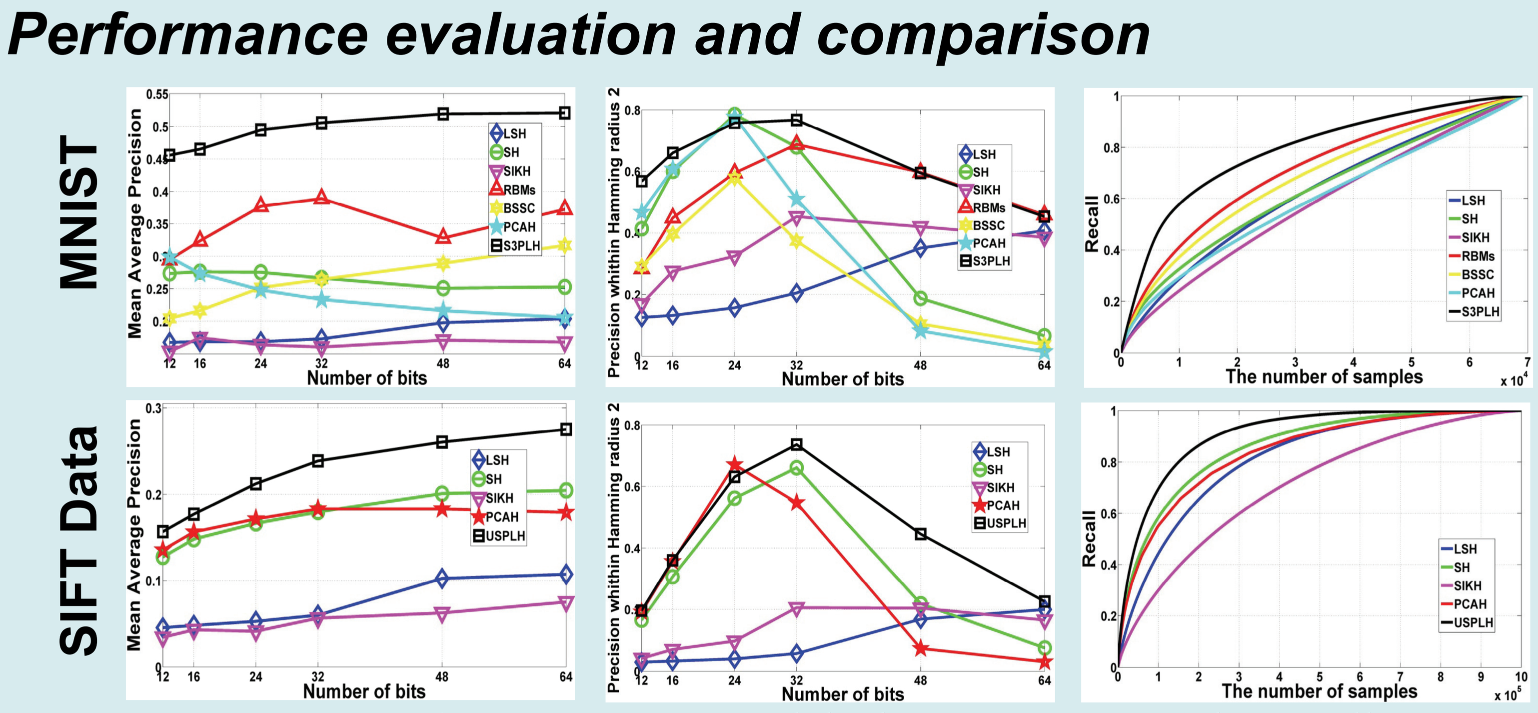
Datasets

- MNIST Digits: 70K, 1K training labels, 1K query tests (semi-supervised)
- SIFT Feature: 1 Million, 2K pseudo labels, 10K query tests (unsupervised)

Experimental protocol

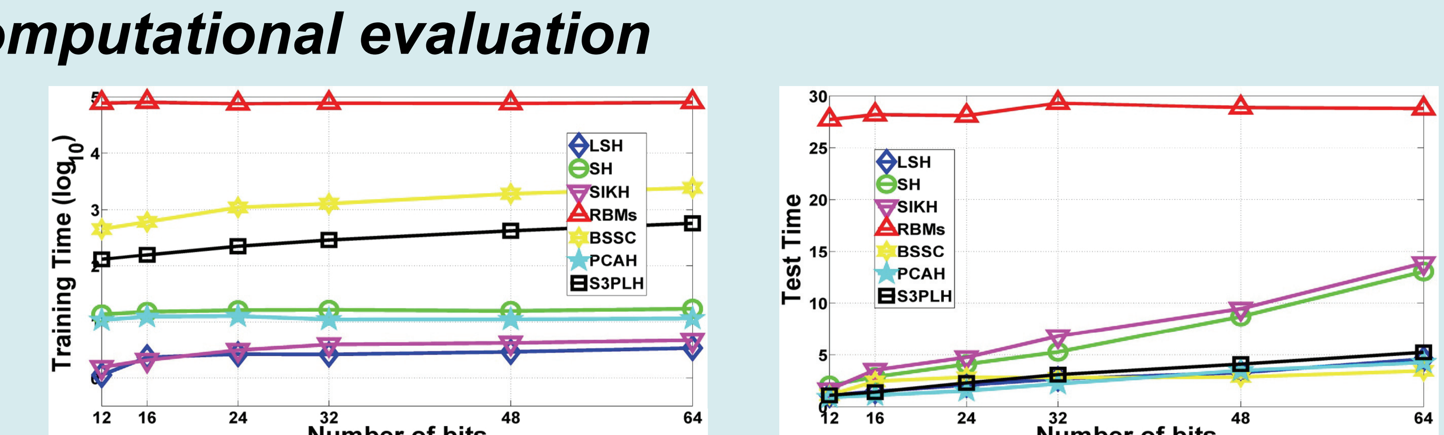
- Hash lookup (precision within hamming radius 2)
  Empty return treated as failed query with zero precision
- Hamming ranking (Mean Average Precision)
- Compared methods: *LSH, SH, SIKH, RBMs, BSSC, PCAH*

Performance evaluation and comparison



Hamming ranking *Hash lookup* *48-bit Recall curve*

Computational evaluation



Conclusion:

- A semi-supervised paradigm for hashing learning (empirical fitness with information theoretic regularization)
- Sequential learning idea for error correction
- Extension of unsupervised case
- Easy implementation and highly scalable

Future work:

- Theoretical analysis of performance guarantee
- Weighted hamming embedding

Columbia University in the City of New York

Google