Efficient Training of Very Deep Neural Networks for Supervised Hashing

Ziming Zhang [1,2,3] Yuting Chen [1] Venkatesh Saligrama [1]

[1] Data Science & Machine Learning Lab, College of Engineering, Boston University, Boston, USA
[2] School of Automation, Huazhong University of Science and Technology, Wuhan, China
[3] MOE Key Laboratory of Image Information Processing and Intelligent Control, Huazhong University of Science and Technology, Wuhan, China

Motivation:

Deep Neural Networks

- Excellent performance
- Slow training speed:
  - VGG [Simonyan and Zisserman] on ILSVRC2014 with 16 conv layers takes approximately one month using 4 GPUs
- Vanishing gradient problem in back propagation (using the chain rule)
- Can we train DNNs faster?
  - Our network: 64 fully connected layers, 1024 nodes per layer, 1 GPU, training time ~3 hours
  - State-of-the-art for supervised hashing.

SOL

Optimization:

- Introduce auxiliary variables:

$$\min_{\Theta,\tilde{\Theta},W,\{z_{i,m}\}} \Omega(\Theta,W)+\sum_i \ell(W^T z_{i,M}, y_i)$$
$$s.t.\ z_{i,0}=x_i, \tilde{\theta}_i^{(m)}=\theta^{(m)}, 1\le m\le M, \forall i$$
$$z_{i,m}=f_m(z_{i,m-1};\theta^{(m)}), 1\le m\le M, \forall i$$

- Augmented Lagrangian (or Adaptive Direction Method of Multipliers):

$$\min_{\Theta,\tilde{\Theta},W,\{z_{i,m}\},\{u_{i,m}\}} \Omega(\Theta,W)+\sum_i \ell(W^T z_{i,M}, y_i) + \frac{\beta}{2}\sum_{i,m=1}\left\|z_{i,m}-f_m\left(z_{i,m-1};\tilde{\theta}_i^{(m)}\right)+u_{i,m}\right\|_2^2 + \frac{\gamma}{2}\sum_{i,m=1}\left\|\theta^{(m)}-\tilde{\theta}_i^{(m)}+v_{i,m}\right\|_2^2$$
$$s.t.\ z_{i,0}=x_i, \forall i$$

- Gradients w.r.t. variables

$$\frac{\partial g}{\partial \tilde{\theta}_i^{(m)}}=\frac{\partial \Omega}{\partial \tilde{\theta}_i^{(m)}}-\beta\sum_i\left(\boxed{z_{i,m}}-f_m\left(\boxed{z_{i,m-1}};\tilde{\theta}_i^{(m)}\right)+u_{i,m}\right)\frac{\partial f_m}{\partial \tilde{\theta}_i^{(m)}} - \gamma\sum_i\left(\theta^{(m)}-\tilde{\theta}_i^{(m)}+v_{i,m}\right)$$

$$\frac{\partial g}{\partial z_{i,M}}=\frac{\partial \ell}{\partial z_{i,M}}+\beta\left(\boxed{z_{i,M}}-f_M\left(\boxed{z_{i,M-1}};\tilde{\theta}_i^{(m)}\right)+u_{i,M}\right)$$

$$\frac{\partial g}{\partial z_{i,m}}=-\beta\frac{\partial f_{m+1}}{\partial z_{i,m}}\left(\boxed{z_{i,m+1}}-f_{m+1}\left(\boxed{z_{i,m}};\tilde{\theta}_i^{(m+1)}\right)+u_{i,m+1}\right)+\beta\left(\boxed{z_{i,m}}-f_m\left(\boxed{z_{i,m-1}};\tilde{\theta}_i^{(m)}\right)+u_{i,m}\right)$$

- **Supervised hashing**
  - Goal: learn hash codes for linear classification [Shen et al.]
  - General problem:

$$\min_{\Theta,W,B} \Omega(\Theta,W)+\sum_i \ell(W^T b_i, y_i), s.t., b_i=\mathrm{sgn}(F(x_i,\Theta)), \forall i$$

OPT

Method

Train DNN with optimization problem:

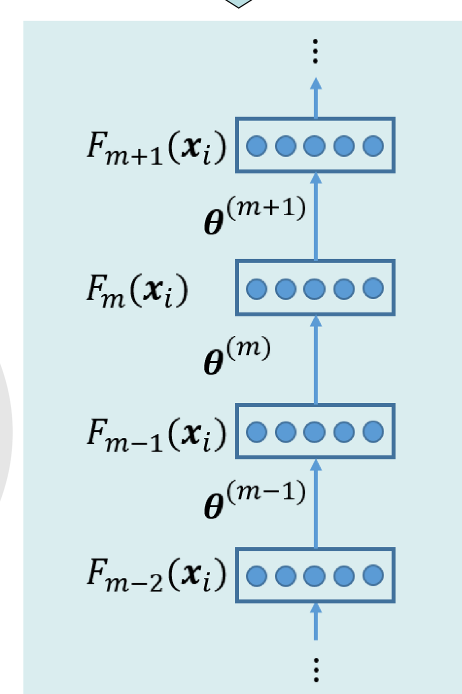
$$\min_{\Theta,W} \Omega(\Theta,W)+\sum_i \ell(W^T F_M(x_i), y_i)$$
$$s.t.\ F_0(x_i)=x_i, F_m(x_i)=f_m(F_{m-1}(x_i);\theta^{(m)}), 1\le m\le M, \forall i$$

$\Theta=\{\theta^{(m)}\}_{m=1,\cdots,M}$: network weights

$\theta^{(m)}\in R^{D_m\times D_{m-1}}$: layer weights

$f_m: R^{D_{m-1}}\to R^{D_m}$: activation function

e.g. $f_m(x_i;\theta^{(m)})=\max\{0,\theta^{(m)}x_i\}$

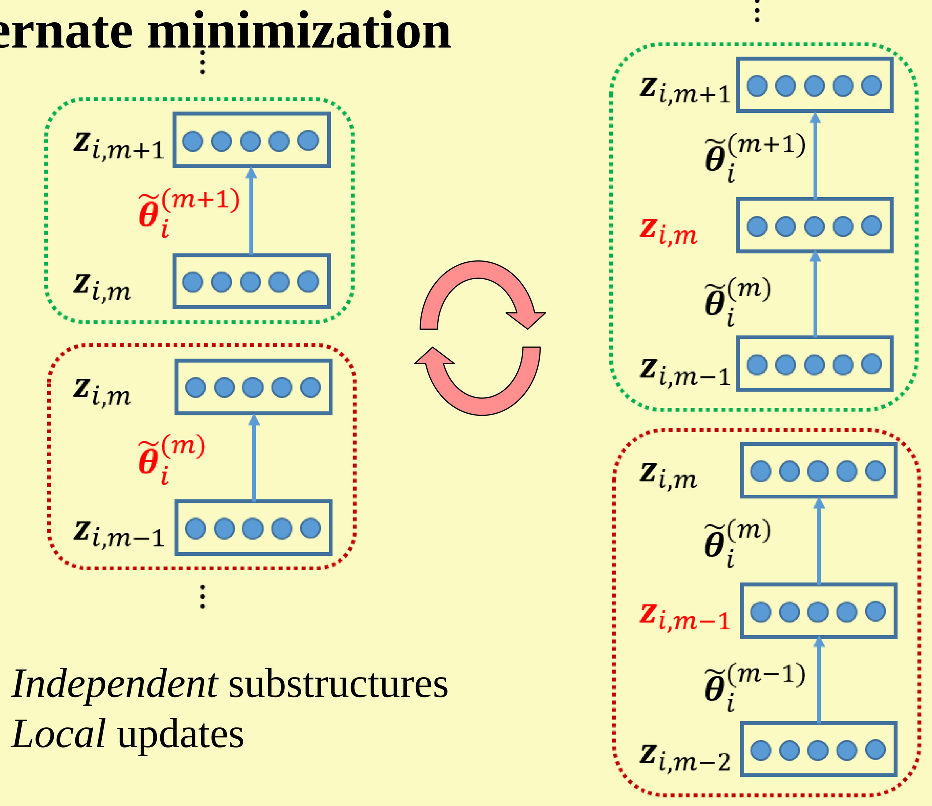


Decomposes the training process

- Break long term dependency: $z_{i,m}=F_m(x_i), \forall i,m$
- Dependency between loss ℓ and regularizer Ω: $\tilde{\theta}_i^{(m)}=\theta^{(m)}, \forall i, \forall m$

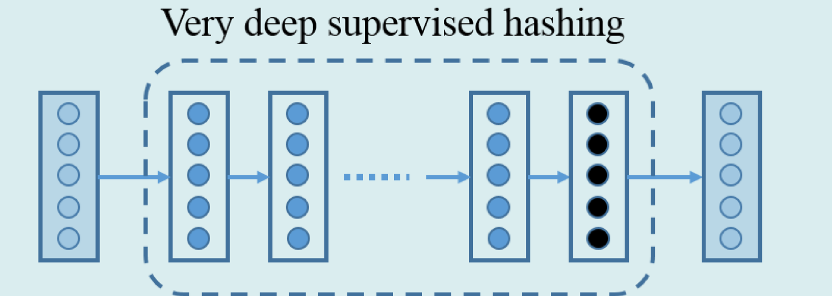
θ updates **z updates**

Alternate minimization



- ✓ *Independent* substructures
- ✓ *Local* updates

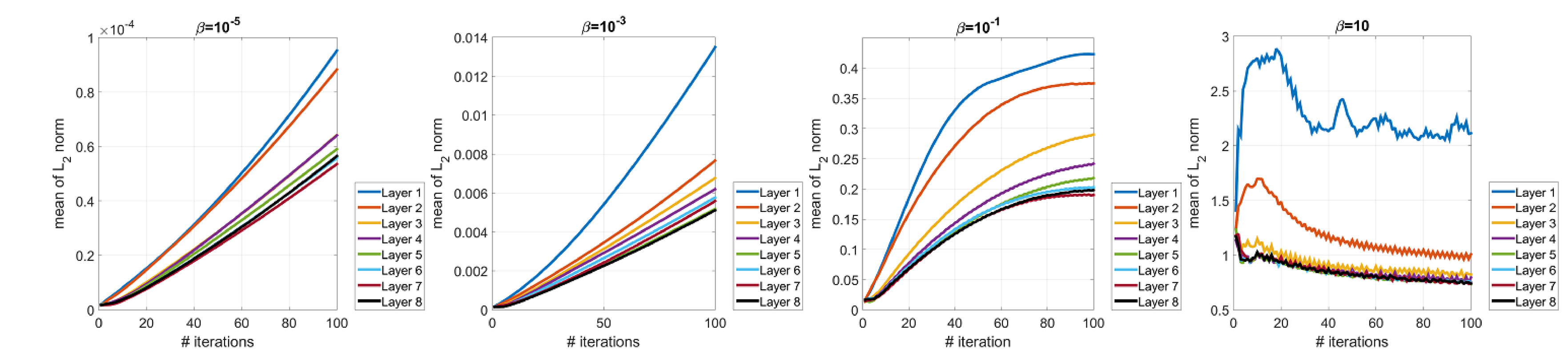
Application: Hashing



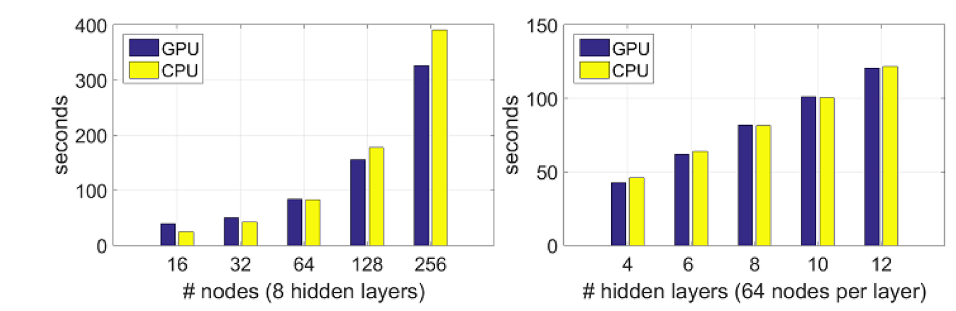
ANLYS

Toy Example

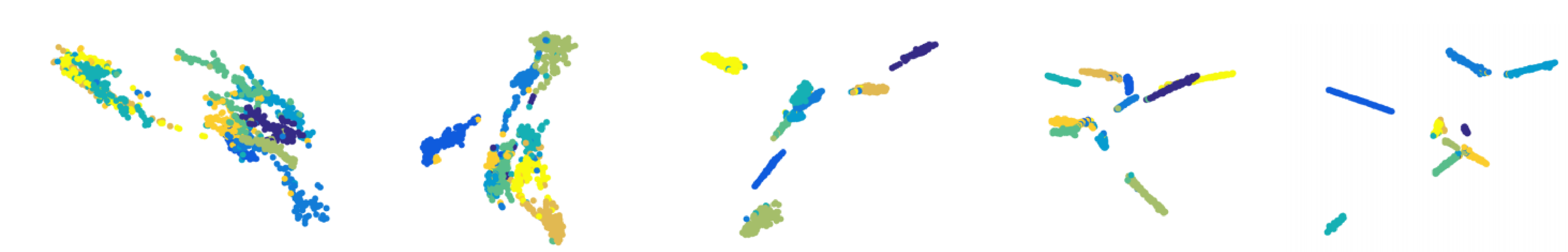
- MNIST dataset: 8 layers, 64 nodes per layer
  - Empirical convergence



- Computational complexity
  - For two-layer substructure: $O(D_m D_{m+1})$
  - Total complexity for the network: $O(\sum_{m=0}^{M} D_m D_{m+1} N)$



- Per layer visualization



(a) Original raw pixel features (b) Layer-1 output features (c) Layer-2 output features (d) Layer-4 output features (e) Layer-8 output features

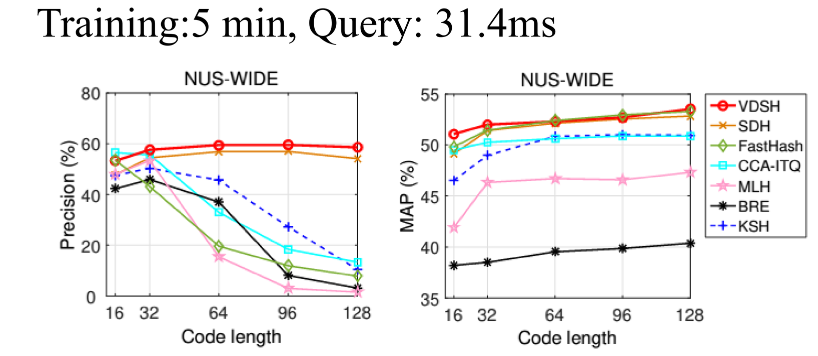
EXP

Experiments on Image Retrieval

- Dataset

	Content	# Classes	# Images	No. Query
CIFAR-10	Natural	10	60k	100 * 10
MNIST	Digits	10	70k	100 * 10
NUS-WIDE	Natural	81	~270k	100 * 21

NUS-WIDE: # layers: 32, # nodes: 128, Training:5 min, Query: 31.4ms

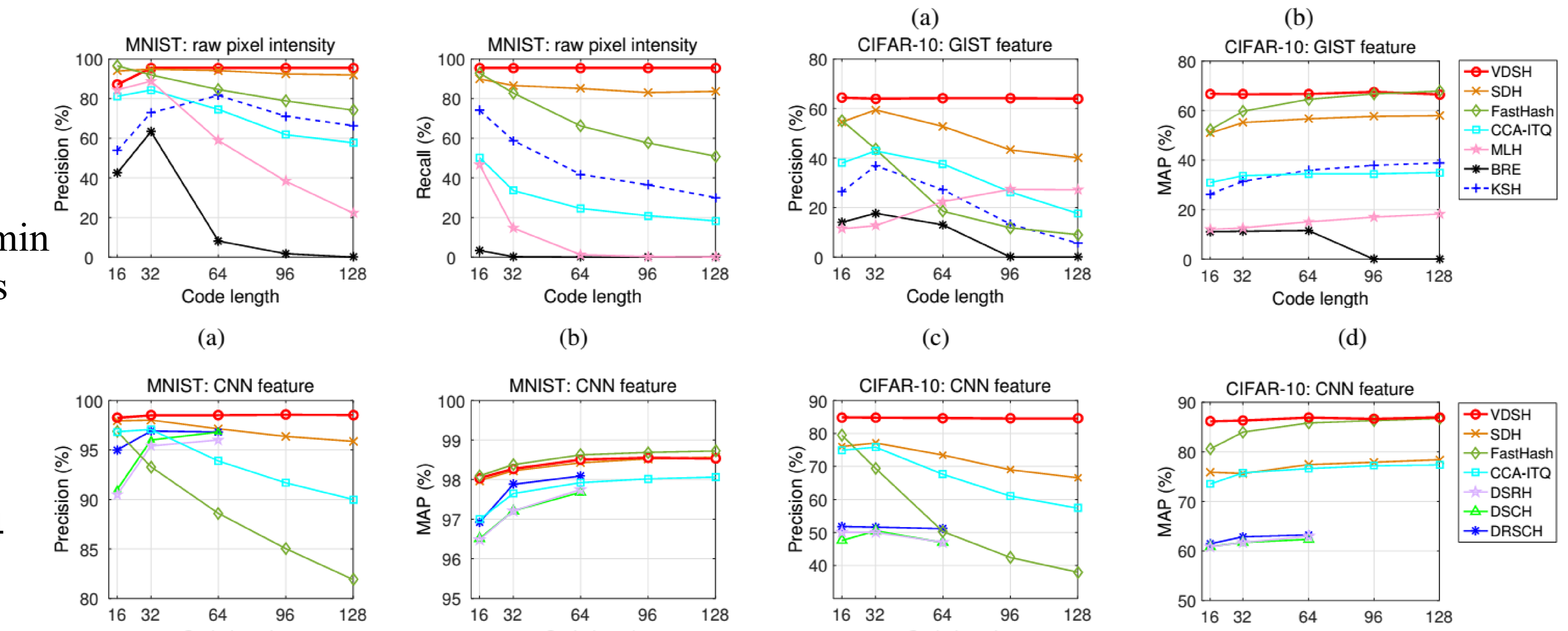


(a) (b)

- Retrieval results

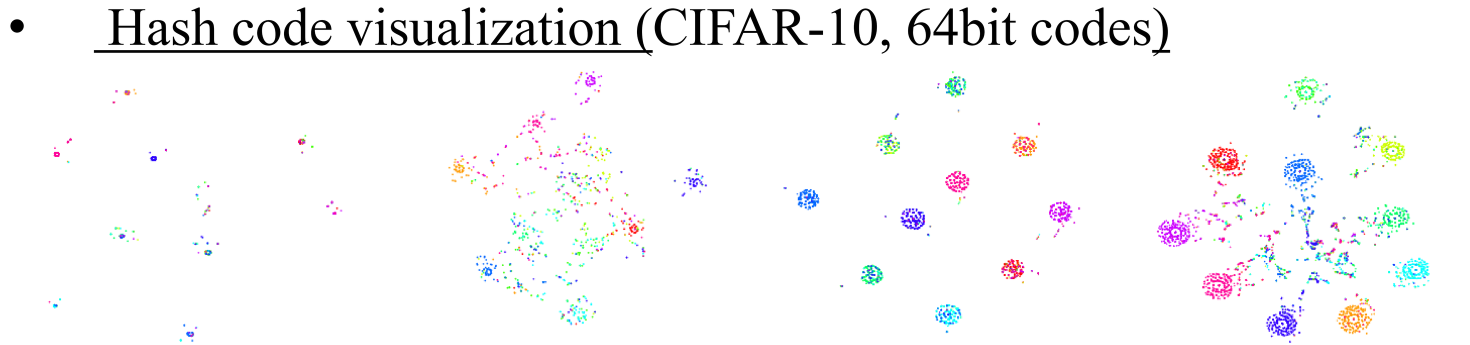
MNIST: # layers: 48 # nodes: 256 Training: 15 min Query: 6.6 ms

CIFAR-10: # layers: 16 # nodes: 1024 Training: 1 h Query: 4ms



(a) (b) (c) (d) (e) (f) (g) (h)

- Hash code visualization (CIFAR-10, 64bit codes)



(a) VDSH: GIST feature (b) SDH: GIST feature (c) VDSH: CNN feature (d) SDH: CNN feature

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.
Fumin Shen, Chunhua Shen, Wei Liu, and Heng Tao Shen. Supervised discrete hashing. In *CVPR*, 2015

Postdoc positions & code available: https://zimingzhang.wordpress.com/