ICIP 2017
End-to-end Learning Binary Representation via Direct Binary Embedding

Liu Liu, **Hairong Qi**

Department of Electrical Engineering and Computer Science
University of Tennessee, Knoxville

lliu25@vols.utk.edu;
hqi@utk.edu

September 15, 2017

Overview

Massive Datasets

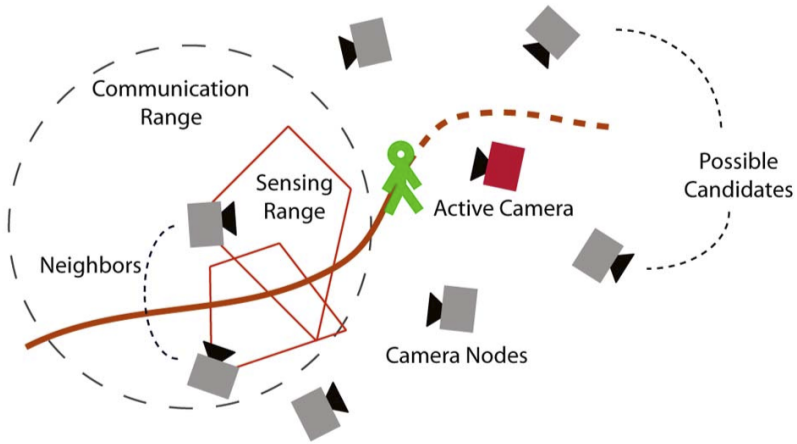
Modern media brought massive *visual dataset*.

- Facebook has about 300 million photo uploads per day
- Instagram Stories has over 250 million active daily user
- ImageNet has over 13 million images, with over 21k categories

Resource Constraint

Resource-constrained environment

- smart camera networks (SCN) often deployed in harsh communication environment
- on-board computation and storage resource is limited
- distributed object/scene recognition



Binary Representation

Focus on learning efficient representation for visual content.

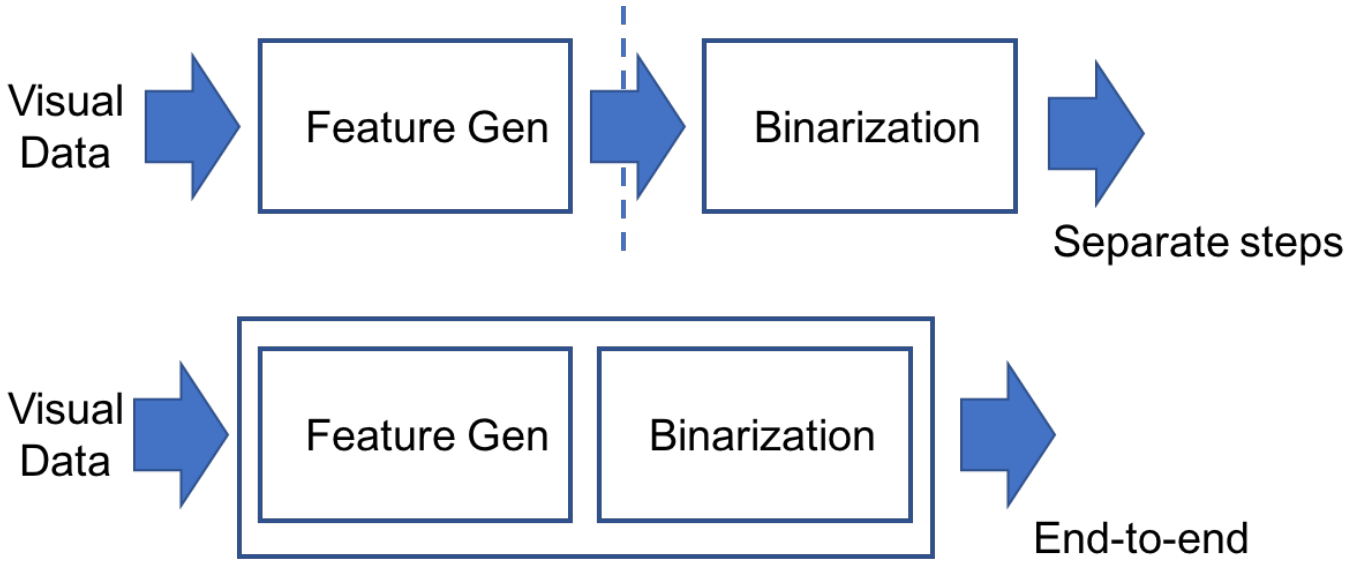
- Project high-dimensional visual data into low-dimensional embedding space
- Binarize the embedding in Hamming space

Why binary?

- binary representation is computationally efficient
- much less storage (comparing to floating number)
- versatile for different tasks: retrieval, classification, etc.

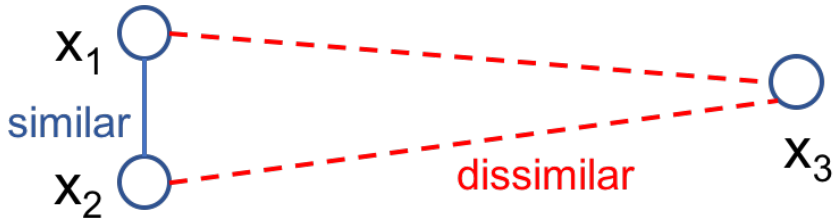
End-to-end Learning

- conventional approach: generating feature step + binary embedding
- end-to-end approach: learning binary embedding for visual content together with feature learning
- usually achieved by deep learning approaches



Learning Discriminative Representation

Traditionally, binary representation is learned as hash code for retrieval purpose, pairwise similarity is exploited.



Problem: the uniqueness of each class is lost when using similarity as supervision.
Approach: use labels as supervision directly

Problem Formulation

$$\min_{W,F} \frac{1}{N}\sum_{i=1}^{N}\left(\mathcal{L}(\mathbf{W}^\top \mathbf{b}_i, y_i) + \lambda\|\mathbf{b}_i - F(I_i;\Omega)\|_2^2\right) \quad (1)$$

$$\text{s.t. } \mathbf{b}_i = \mathit{thresold}(F(I_i;\Omega), 0.5)$$

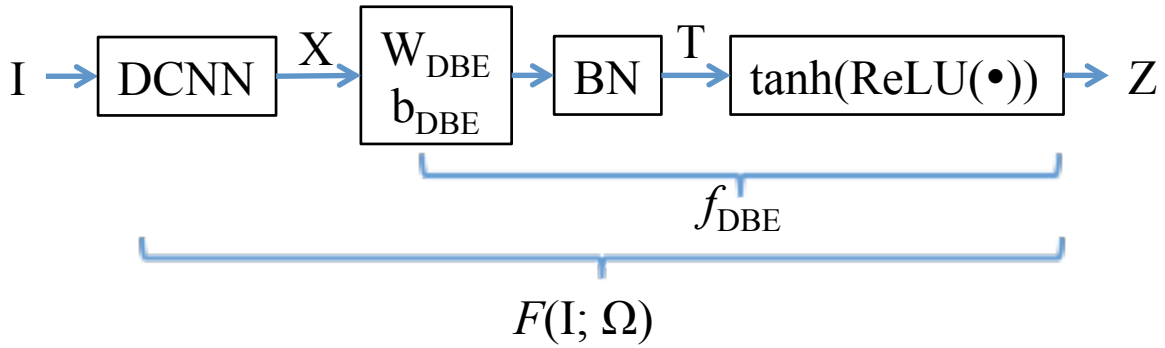
$$F(I,\Omega) = f_{\text{DBE}}(f_n(\cdots f_2(f_1(I;\omega_1);\omega_2)\cdots;\omega_n)\omega_{\text{DBE}}), \quad (2)$$

Similar continuous relaxation:

$$\min_{W,F} \frac{1}{N}\sum_{i=1}^{N}\left(\mathcal{L}(\mathbf{W}^\top F(I_i;\Omega), y_i) + \lambda||2F(I_i;\Omega) - \mathbf{1}| - \mathbf{1}|^2\right) \quad (3)$$

Direct Binary Embedding

$$\mathbf{Z} = f_{\text{DBE}}(\mathbf{X}) = \tanh(\text{ReLU}(\text{BN}(\mathbf{X}\mathbf{W}_{\text{DBE}} + b_{\text{DBE}}))) \tag{4}$$



The benefits of DBE layer approximating binary code are three-fold:

1. batch normalization mitigates training with saturating nonlinearity, and potentially promotes more effective binary representation.
2. ReLU activation is sparse and learns bit '0' inherently.
3. tanh activation bounds the ramping of ReLU activation and learns bit '1' effectively without jeopardizing the sparsity of ReLU.

Direct Binary Embedding

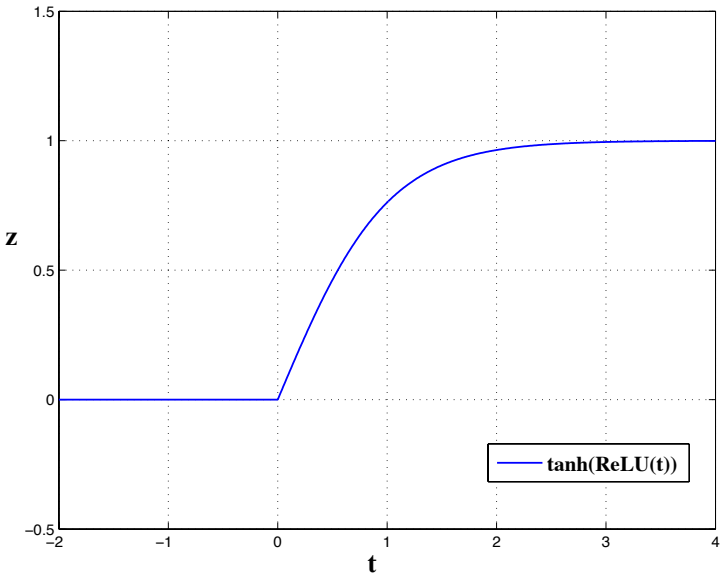


Figure: activation

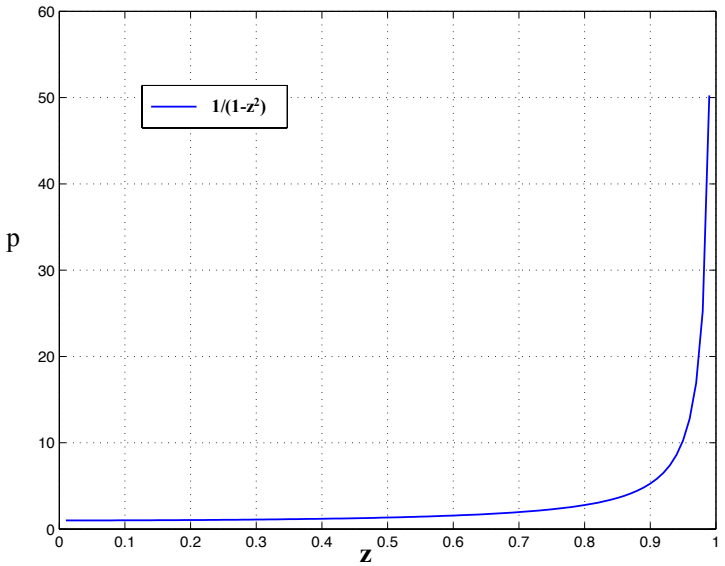


Figure: Probabilistic distribution

Classification

Multiclass classification:

$$\min_{\mathbf{W},F} \; -\frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{C}\mathbb{1}(y_i)\log\frac{e^{\mathbf{w}_k^\top F(I_i;\Omega)}}{\sum_{j=1}^{C}e^{\mathbf{w}_j^\top F(I_i;\Omega)}} \tag{5}$$

$$\text{s.t. } F(\mathbf{I},\Omega) = f_{\text{DBE}}(f_n(\cdots f_2(f_1(\mathbf{I};\omega_1);\omega_2)\cdots;\omega_n)\omega_{\text{DBE}})$$

Multilabel classification:

$$\min_{\mathbf{W},F} \; -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{c_+}\frac{1}{c_+}\log\frac{e^{\mathbf{w}_j^\top F(I_i;\Omega)}}{\sum_{p=1}^{C}e^{\mathbf{w}_p^\top F(I_i;\Omega)}} - \nu\frac{1}{N}\sum_{i=1}^{N}\sum_{p=1}^{C}\Big[\rho\mathbb{1}(y_i) \tag{6}$$

$$\times\log\frac{1}{1+e^{\mathbf{w}_p^\top F(I_i;\Omega)}} + (1-\mathbb{1}(y_i))\log\frac{e^{\mathbf{w}_p^\top F(I_i;\Omega)}}{1+e^{\mathbf{w}_p^\top F(I_i;\Omega)}}\Big]$$

$$\text{s.t. } F(\mathbf{I};\Omega) = f_{\text{DBE}}(f_n(\cdots f_2(f_1(\mathbf{I};\omega_1);\omega_2)\cdots;\omega_n)\omega_{\text{DBE}})$$

Toy Example

MNIST with LeNet + DBE layer

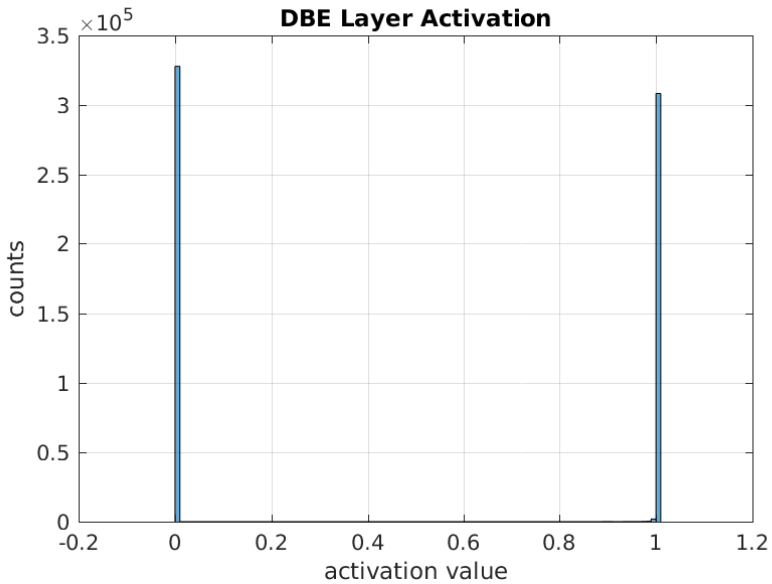


Figure: The histogram of DBE layer activation

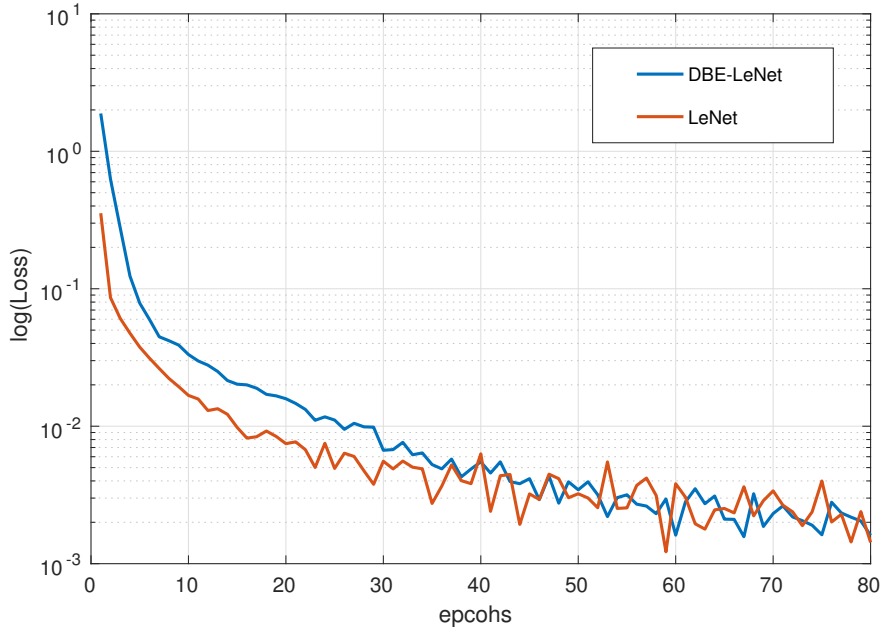


Figure: The convergence of the original LeNet and with DBE trained on MNIST

Toy Example

Method	LeNet	DBE-LeNet	SDH	FastHash
testing acc(%)	99.34	99.34	99.14	98.62

Table: The comparison of the testing accuracy on MNIST. Code-length for all hashing algorithms is 64-bit. LeNet feature (1000-d continuous vectors) is used for SDH and FastHash.

λ	0	1e-4	1e-3	1e-2	1e-1
testing acc(%)	99.34	99.34	99.30	99.26	99.01

Table: The impact on quantization error coefficient λ

Experiment

Evaluate the proposed DBE layer with the deep residual network (ResNet)
Datasets: CIFAR-10 (50K training, 10K test) and MS COCO (83K training, 40K test)
Exp. 1 **Classification**

Methods	Testing Accuracy (%)
CCA-ITQ	56.34
FastHash	57.82
SDH	67.73
DLBHC	86.73
ResNet	**92.38**
DBE (ours)	<u>92.35</u>

Table: The testing accuracy of different methods on CIFAR-10 dataset. All binary representations have code-length of 64 bits.

Experiment

Performance w.r.t. different code lengths

Code length (bits)	16	32	48	64	128
testing acc(%)	91.63	92.04	92.20	92.35	92.36

Table: Classification accuracy of DBE on CIFAR-10 dataset across different code lengths

Experiment

Exp. 2 Natural object retrieval and multilabel image retrieval

Code length (bits)	12	24	36	48
CCA-ITQ	0.261	0.289	0.307	0.310
FastHash	0.286	0.324	0.371	0.382
SDH	0.342	0.397	0.411	0.435
DSH	0.616	0.651	0.661	0.676
DSRH	0.792	0.794	0.792	0.792
DLBHC	0.892	0.895	0.897	0.897
DBE (ours)	**0.912**	**0.924**	**0.926**	**0.927**

Table: Comparison of mean average precision (mAP) on CIFAR-10

Experiment

Code length (bits)	16	24	32	48	64
CCA-ITQ	0.477	0.481	0.485	0.490	0.494
CMFH	0.462	0.476	0.484	0.497	0.505
CCA-ACQ	0.483	0.500	0.504	0.515	0.520
DHN	0.507	0.539	0.550	0.559	0.570
DBE (ours)	**0.623**	**0.657**	**0.670**	**0.692**	**0.716**

Table: Comparison of mean average precision (mAP) on COCO

Experiment

Exp. 3 Multilabel image annotation

Method	O-P	O-R	O-F1
WARP	**59.8**	61.4	60.6
DBE-Softmax	59.1	62.1	60.3
DBE-weighted binary cross entropy	57.1	60.8	58.9
DBE-joint cross entropy	59.5	**62.7**	**61.1**

Table: Performance comparison on COCO for $K = 3$. The code length for all the DBE methods is 64-bit.

Conclusion

1. Proposed Direct Binary Embedding (DBE) layer to learn hashing functions effectively
2. Provided theoretical and experimental evidence to validate discarding continuous relaxation
3. Experiments demonstrated the effectiveness of DBE on multiple tasks, e.g., classification, retrieval, annotation.

THANK YOU