A Graph Regularized Deep Neural Network for Unsupervised Image Representation Learning

Shijie Yang[1,2], Liang Li[2], Shuhui Wang[2], Weigang Zhang[1,3] ,Qingming Huang[1,2]

[1]University of Chinese Academy of Sciences , [2]Institute of Computing Technology, Chinese Academy of Sciences,[3] Harbin Institute of Technology

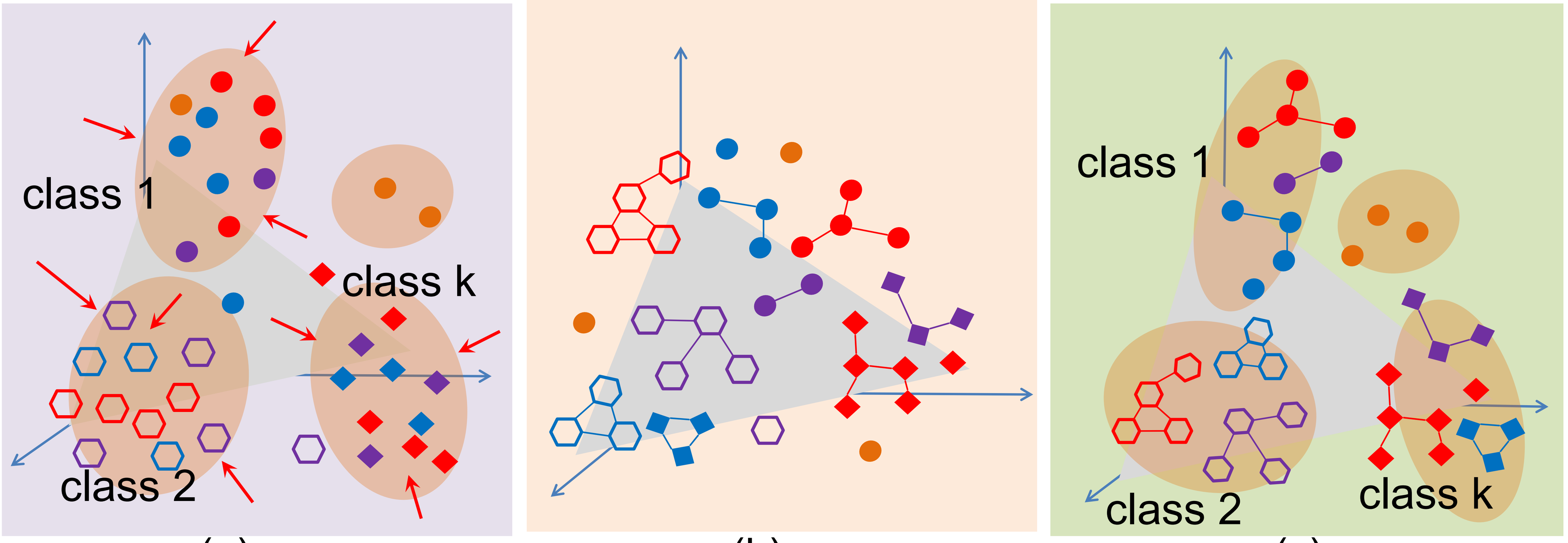
shijie.yang@vipl.ict.ac.cn, liang.li@vipl.ict.ac.cn, wangshuhui@ict.ac.cn , wgzhang@hit.edu.cn, qmhuang@ucas.ac.cn

IEEE 2017 Conference on Computer Vision and Pattern Recognition

CVPR July 21-26 HONOLULU 2017

Motivation:

- Deep Auto-Encoder (DAE) has shown its promising power in high-level representation learning.
  - *Fig.1(a) : DAE extracts the high-level salient factors, following a global reconstruction criterion.*
- A stream of manifold learning methods benefit from the local invariance theory.
  - *Fig.1(b): It emphasizes on preserving the local geometric structure, and infer the subspace according to the affinity propagations.*
- We propose a graph regularized deep neural network (GR-DNN) to endue traditional DAEs with the ability of retaining local geometric structure.
- *Fig.1(c) :* The discriminative feature embedding is learned to
  - ***extract the global high-level semantic abstractions***
  - ***preserve the geometric structure within local manifold tangent space***



(a) (b) (c)

Figure 1: Toy examples of the learned 3-D embedding subspace: (a) traditional DAEs, (b) local invariance learners, (c) our GR-DNN. Different shapes denote the labels, and different colors denote the neighbor relations.

Contribution:

- A graph regularized deep neural network is proposed to leverage DAEs with the local invariant theory.
- The proposed deep-structured graph regularizer achieves both the lower computational complexity and superior learning performance, compared with traditional graph Laplacian regularizer.

Methodology:

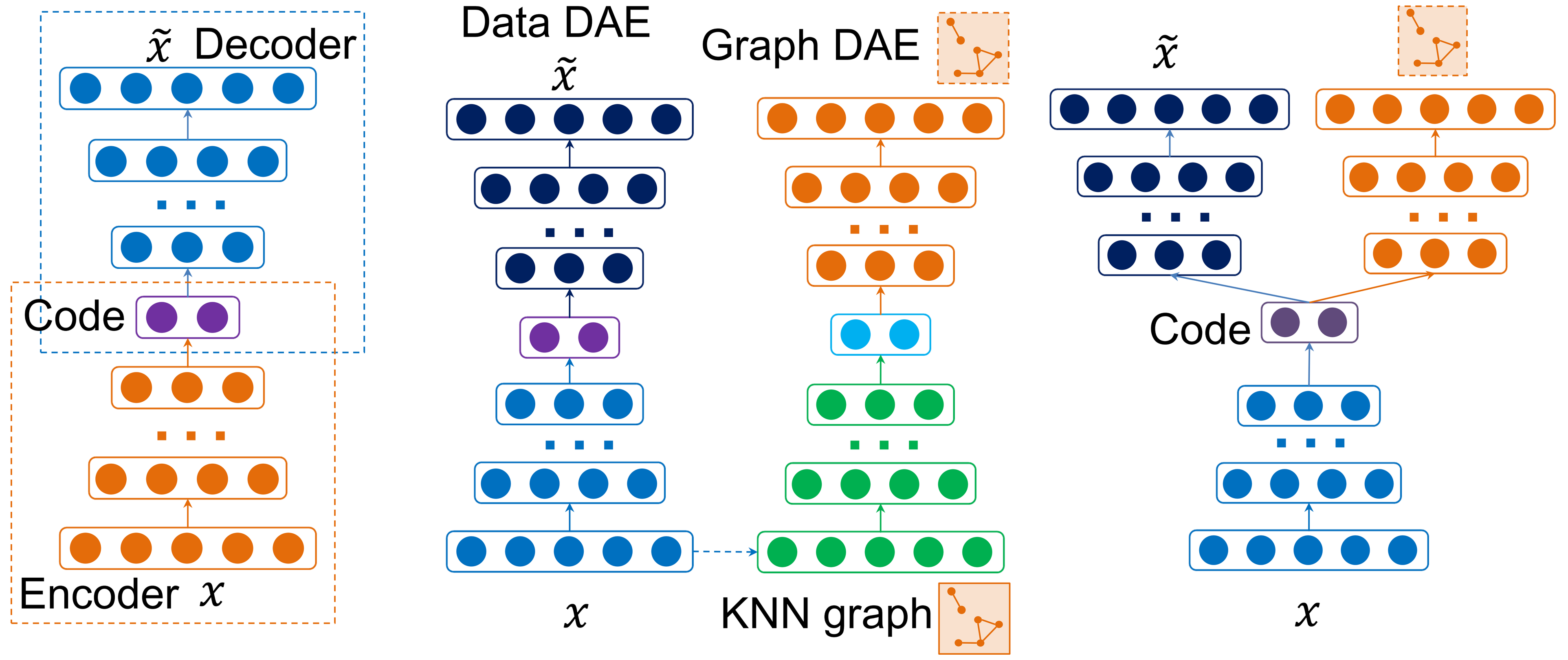
- GR-DNN is composed of one encoder and two decoders (*Fig. 2(c)*)
  - *Decoder 1: reconstruct the original input feature vectors*
  - *Decoder 2: reconstruct the pre-constructed K-NN affinity graph* $\mathbf{S} \in \mathbb{R}^{n\times n}$ *(Eq. (1))*

$$[\mathbf{S}]_{ij}=\begin{cases}\exp\left(\frac{-\|\mathbf{x}_i-\mathbf{x}_j\|^2}{\sigma}\right) & \text{, if } \mathbf{x}_i \text{ and } \mathbf{x}_j \text{ are connected}\\ 0 & \text{, otherwise}\end{cases} \quad \textbf{Eq. (1)}$$

- Objective:

$$Cost = \underbrace{\Delta(\mathbf{X},\tilde{\mathbf{X}})+\gamma\Phi}_{traditional\,(regularized)\,DAE} + \underbrace{\eta\Delta(\mathbf{S},\tilde{\mathbf{S}})}_{graph\ regularizer} \quad \textbf{Eq. (2)}$$

- Training:
  - *The path-wise pre-training* the Data-DAE and Graph-DAE (*Fig. 2(b)*)
  - *The joint fine-tuning*(*Fig. 2(c)*): select the Data-DAE's encoder as the final encoder
- Anchor graph (AG) for large-scale data
  - *Select a small set of representative anchor points* $\{\mathbf{a}_1, \mathbf{a}_2, ..., \mathbf{a}_{N_a}\}$
  - *Build* $\hat{\mathbf{S}} \in \mathbb{R}^{n\times N_a}$ *instead of* $\mathbf{S} \in \mathbb{R}^{n\times n}$
  - *Now we only consider* $O(nN_a)$ *distances*



(a) DAE (b) Pre-training of GR-DNN (c) Joint training

Figure 2: Illustrations of the network structures.

Experiments:

- Datasets: *COIL20, YaleB face, MNIST, MNIST2(noisy MNIST)*
- Comparison Methods: *Raw , NMF , GNMF, DAE, D-DAE: Denoising-DAE, C-DAE: Contractive-DAE, LAE: Laplacian auto-encoders, GR-DNN(DAE), GR-DNN(D-DAE), GR-DNN(C-DAE).*
- K-NN Search Results:

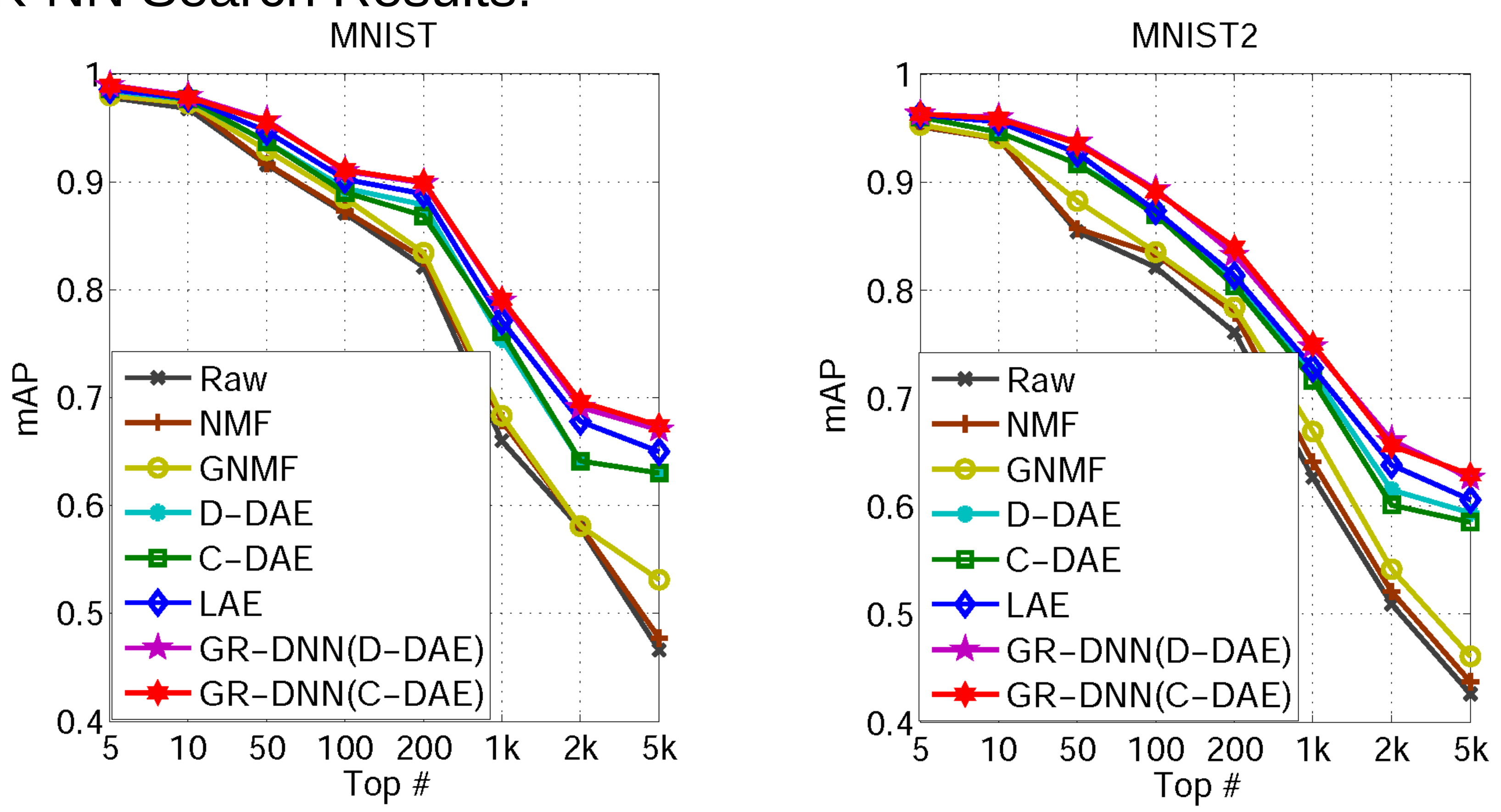


Figure 3: The top # mAP score on MNIST, MNIST2.

- Visualization:

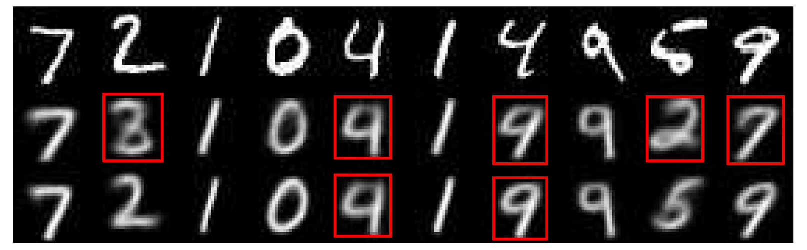


Figure 4: The top, middle and bottom panel respectively shows the original samples, the reconstructed samples by D-DAE and GR-DNN(D-DAE).

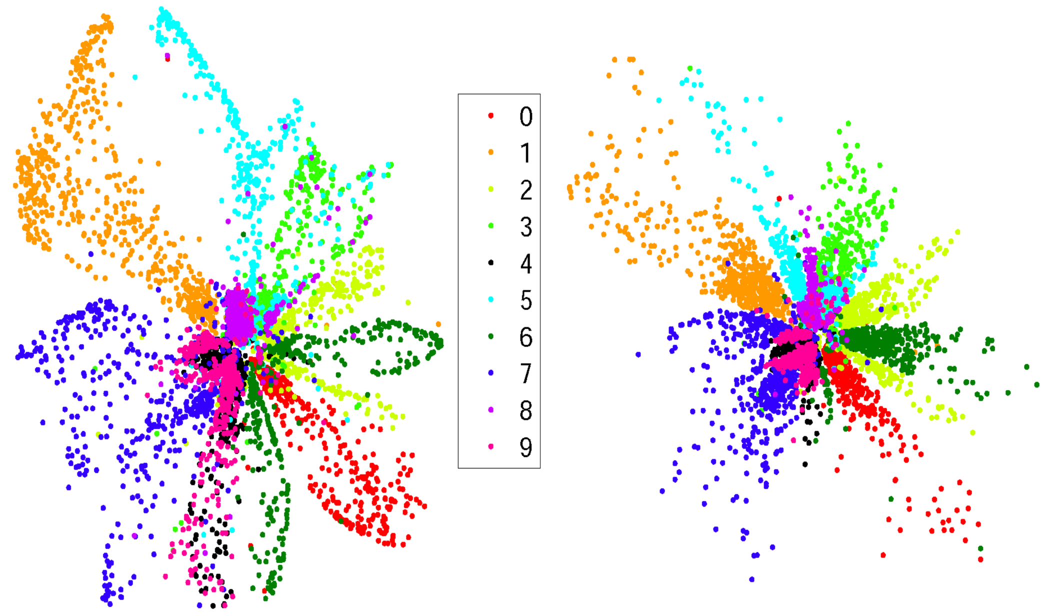


Figure 5: The left and right panel shows the 2D codes produced by D-DAE and GR-DNN(D-DAE) respectively.

Conclusion:

- GR-DNN can enhance traditional DAEs and extract more discriminative features with local geometric structure preserved.
- Future works: learning compact hash codes for retrieval task, and extending GR-DNN to multi-view learning scenarios.

Code (Theano-based) : https://github.com/ysjakking/GR-DNN