Learning a Self-Expressive Network for Subspace Clustering

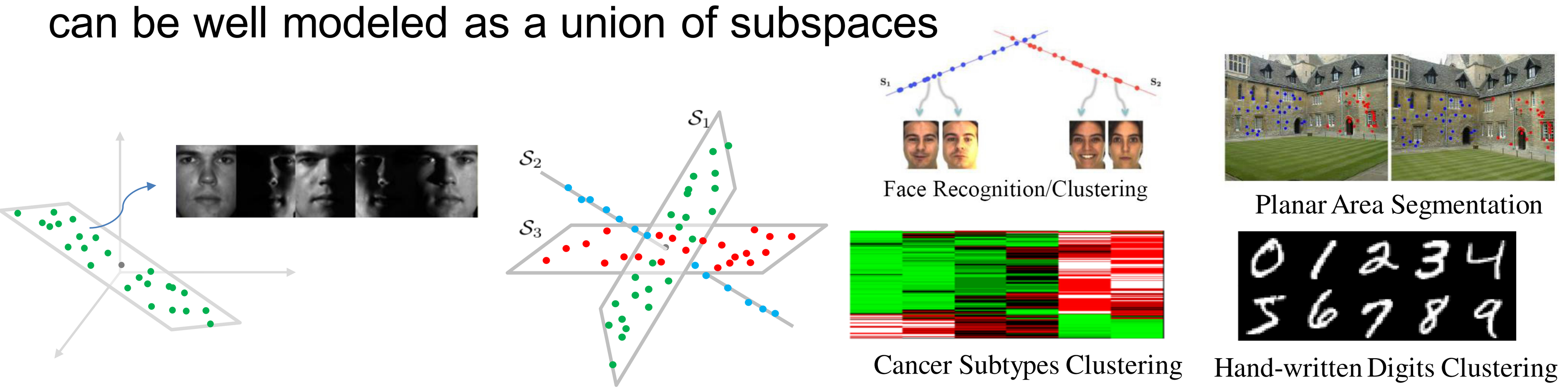
Shangzhi Zhang[1], Chong You[2], Rene Vidal[3], Chun-Guang Li[1]

[1] Beijing University of Posts & Telecommunications [2] University of California, Berkeley [3] Johns Hopkins University

CVPR VIRTUAL JUNE 19-25

Introduction

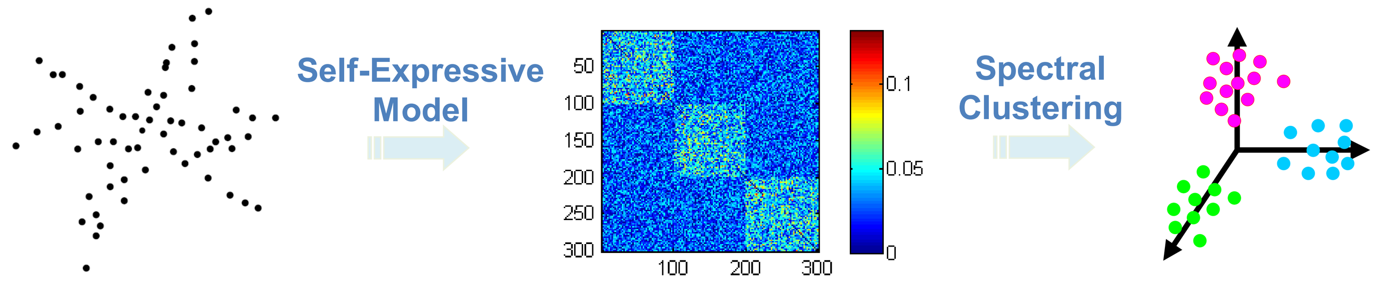
- High dimensional visual data with a specific semantic pattern can be modeled by a linear subspace. Complex high dimensional data with multiple patterns can be well modeled as a union of subspaces



(a) Linear Subspace (b) Union of Subspaces (c) Applications

Subspace Clustering: segment data points into each subspace

Self-Expressiveness based subspace clustering



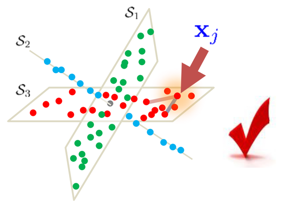
- **Step1**: Perform **Self-Expressive Model** to find self-expressive coefficients which are used to build an affinity matrix

$$\min_{c_{ij}} \frac{\gamma}{2}\|x_j - \sum_{i\neq j} c_{ij}x_i\|_2^2 + \sum_{i\neq j} r(c_{ij})$$

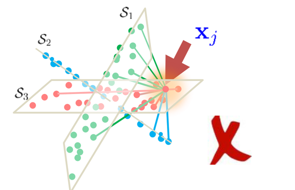
  - ✓ Imposing proper regularization term $r(c_{ij})$:
    - **subspace preserving property**

$$r(c_{ij}) = \begin{cases} \|c_{ij}\|_1 \\ \|c_{ij}\|_2^2 \\ \|c_{ij}\|_1 + \|c_{ij}\|_2^2 \end{cases}$$

  - ✓ Indue an affinity matrix A via:
    - $a_{ij} = |c_{ij}| + |c_{ji}|$ or symmetrized 3-nereast neighbor graph



(a) subspace preserving



(b) not subspace preserving

- **Step2**: Apply **Spectral Clustering**

Drawbacks of Previous Methods

- **Scalability issue**: $O(N^2)$ variables to solve, time consuming on large-scale datasets
  - ✓ **Prior work**: Dictionary based methods: use a small dictionary (e.g. SSSC, OLRSC, ESC); Decomposition based methods: greedy (e.g. SSC-OMP), active support (e.g. EnSC), dropout (e.g. S^3COMP)
- **Generalization**: Previous methods lack of the ability to handle out-of-sample data

Our Proposal

- **Basic idea: Reparametrize Self-Expressive Model into Self-Expressive Network (SENet)**

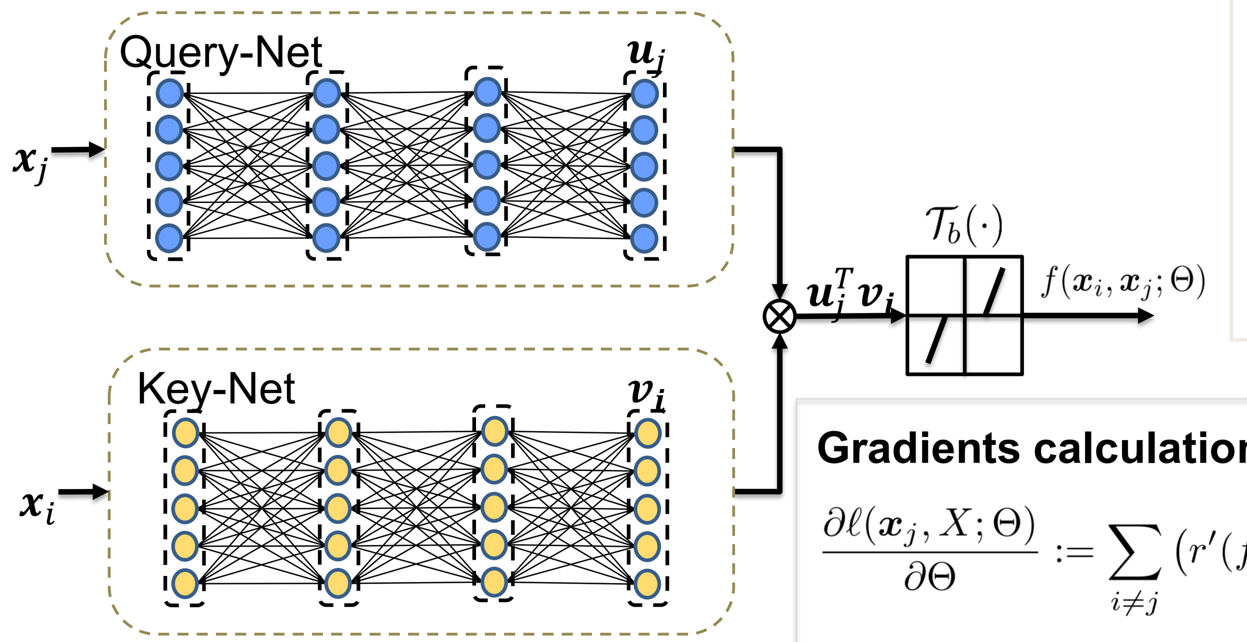
(EnSC, You et al. CVPR'16)

$$\min_{c_{ij}} \frac{\gamma}{2}\|\boldsymbol{x}_j - \sum_{i\neq j} c_{ij}\boldsymbol{x}_i\|_2^2 + \sum_{i\neq j} r(c_{ij}), \text{ where } r(c_{ij}) = \lambda|c_{ij}| + \frac{1-\lambda}{2}c_{ij}^2$$

$$\min_{\Theta} \frac{\gamma}{2}\|\boldsymbol{x}_j - \sum_{i\neq j} f(\boldsymbol{x}_i,\boldsymbol{x}_j;\Theta)\boldsymbol{x}_i\|_2^2 + \sum_{i\neq j} r(f(\boldsymbol{x}_i,\boldsymbol{x}_j;\Theta))$$

SENet: learn coefficients c_{ij} via a neural network $f(\boldsymbol{x}_i, \boldsymbol{x}_j)$

- **Network Instantiation —— Query-Key Structure**



$$f(\boldsymbol{x}_i,\boldsymbol{x}_j;\Theta) \coloneqq \mathcal{T}_b(\boldsymbol{u}_j^T\boldsymbol{v}_i) \in \mathbb{R}$$
$$\boldsymbol{u}_j \coloneqq \boldsymbol{u}(\boldsymbol{x}_j;\Theta_u) \in \mathbb{R}^p$$
$$\boldsymbol{v}_i \coloneqq \boldsymbol{v}(\boldsymbol{x}_i;\Theta_v) \in \mathbb{R}^p$$
$$\mathcal{T}_b(\cdot) \coloneqq sgn(\cdot)\max\{0, |\cdot| - b\}$$

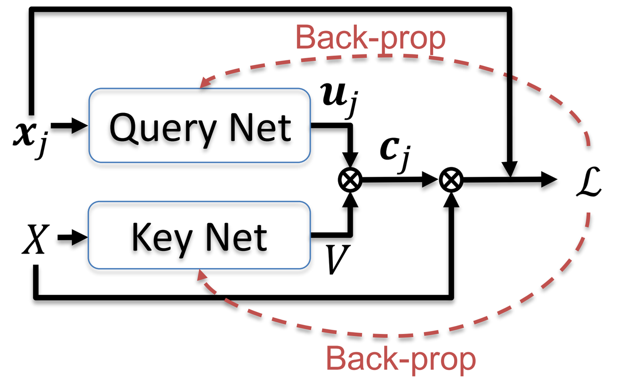
Gradients calculation:

$$\frac{\partial \ell(\boldsymbol{x}_j, X;\Theta)}{\partial \Theta} \coloneqq \sum_{i\neq j}\left(r'(f(\boldsymbol{x}_i,\boldsymbol{x}_j;\Theta)) - \boldsymbol{x}_i^\top \boldsymbol{q}_j\right)\frac{\partial f(\boldsymbol{x}_i,\boldsymbol{x}_j;\Theta)}{\partial \Theta},$$

where residual $\boldsymbol{q}_j \coloneqq \gamma(\boldsymbol{x}_j - \sum_{i\neq j} f(\boldsymbol{x}_i,\boldsymbol{x}_j;\Theta)\boldsymbol{x}_i)$.

☐ Training Algorithm

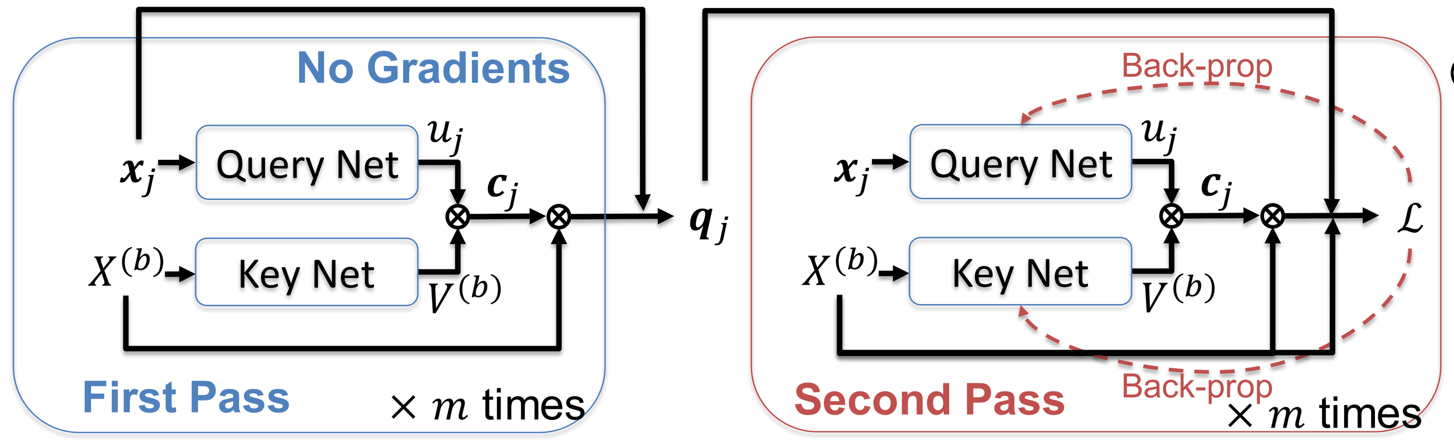
✓ Naïve SGD



(a) Naïve SGD

☒ Requires to load the entire dataset into GPU memory, O(N) memory complexity!

✓ Two-pass SGD



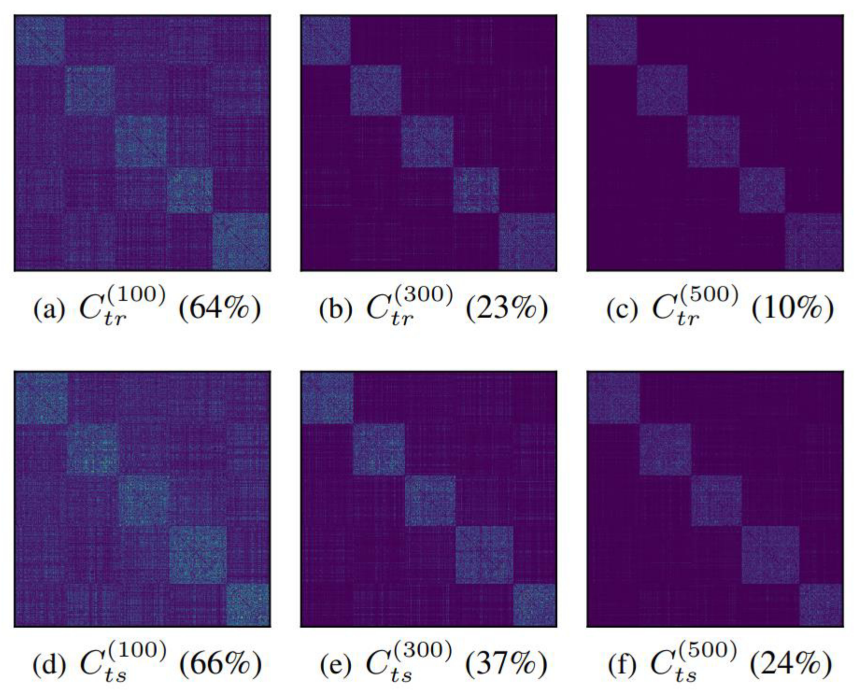
(b) Two-pass SGD

☒ Requires to load a batch of dataset into GPU memory and accumulate gradient in an on-line fashion, O(1) memory complexity!

- First pass is to compute residual $\boldsymbol{q}_j$ in an on-line fashion. Second pass is for gradients accumulation.
- $m = \left\lceil \frac{|X|}{|X^{(b)}|} \right\rceil$, $|X|$ is the number of data points within dataset X.

Experiments

Experiments on Synthetic Datasets



(a) $C_{tr}^{(100)}$ (64%) (b) $C_{tr}^{(300)}$ (23%) (c) $C_{tr}^{(500)}$ (10%)

(d) $C_{ts}^{(100)}$ (66%) (e) $C_{ts}^{(300)}$ (37%) (f) $C_{ts}^{(500)}$ (24%)

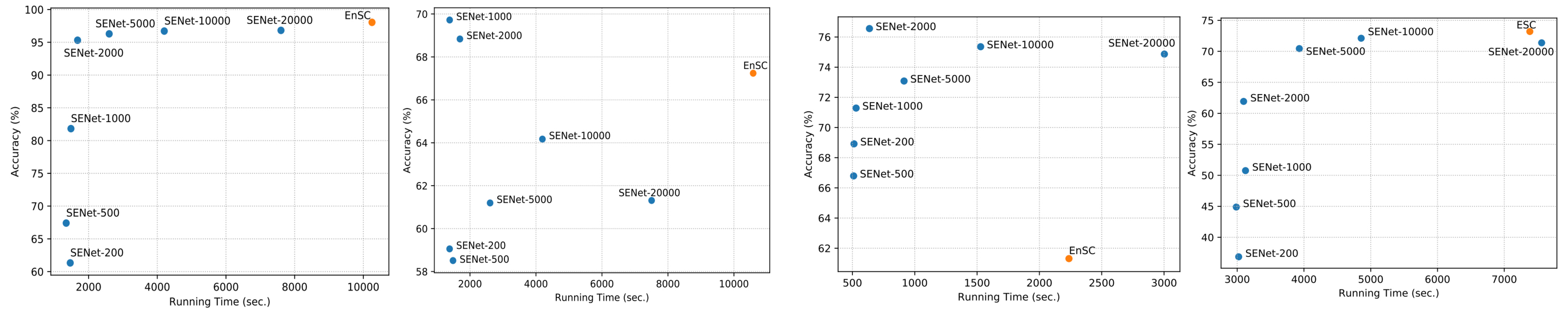
Figure 2. Visualization of self-expressive coefficients computed by SENet trained with {100, 300, 500} iterations on synthetic data where the percentage number in bracket is SRE.

N_i	Methods	$\mathcal{L}$	$\mathcal{L}_{rec}$	$\mathcal{L}_{reg}$	ACC (%)	SRE (%)	CONN
20	EnSC	135.127	0.107	132.442	72.0	49.611	0.178
	SENet	135.132	0.109	132.416	71.0	49.720	0.178
	SENet test	1830.107	72.007	29.937	65.0	58.384	0.318
100	EnSC	559.943	0.526	558.009	93.0	27.370	0.163
	SENet	559.972	0.531	558.022	92.8	27.501	0.165
	SENet test	2935.424	89.325	702.309	79.0	56.897	0.387
200	EnSC	1053.086	0.526	1040.097	96.6	20.067	0.155
	SENet	1053.369	0.531	1040.097	96.0	20.195	0.159
	SENet test	17826.273	599.099	2848.779	84.1	56.256	0.398
1000	EnSC	4884.876	2.095	4832.508	99.4	6.493	0.126
	SENet	4932.907	2.205	4877.781	99.5	9.132	0.155
	SENet test	30037.012	887.323	7853.945	92.3	36.054	0.236
2000	EnSC	9576.154	3.958	9477.197	99.7	4.580	0.108
	SENet	10025.874	4.592	9911.074	99.4	13.555	0.201
	SENet test	44458.734	1453.790	8113.975	97.4	21.863	0.220

Table 1. Comparing SENet to EnSC on synthetic data

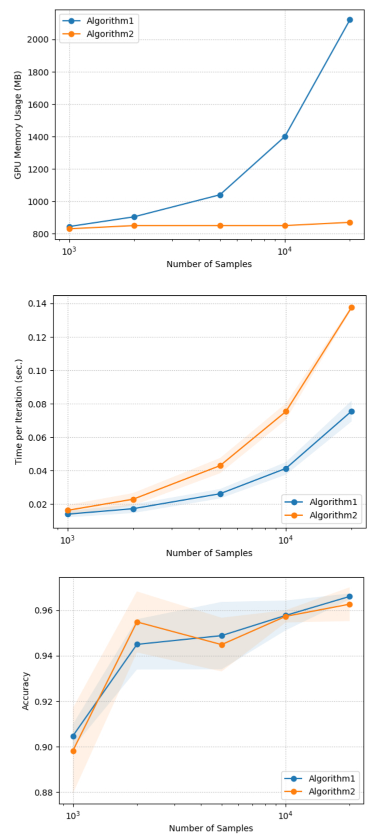
✓ SENet can **well approximate the global solution** of EnSC and can **generalize to out-of-sample data.**

Experiments on Real-World Datasets



(a) MNIST (b) Fashion-MNIST (c) CIFAR-10 (d) EMNIST

Methods	MNIST-full ACC	MNIST-full NMI	MNIST-full ARI	Fashion-MNIST-full ACC	Fashion-MNIST-full NMI	Fashion-MNIST-full ARI	CIFAR-10 ACC	CIFAR-10 NMI	CIFAR-10 ARI	EMNIST ACC	EMNIST NMI	EMNIST ARI
k-means [42]	0.541	0.507	0.367	0.505	0.578	0.403	0.525	0.589	0.276	0.459	0.438	0.316
Spectral [55]	0.728	0.856	0.667	0.625	0.700	0.494	0.455	0.574	0.256	0.662	0.769	0.654
JULE [73]	0.964	0.913	0.927	0.563	0.608	-	0.272	0.192	0.138	-	-	-
DEC [72]	0.863	0.834	-	0.518	0.546	-	0.301	0.257	0.161	-	-	-
DAC [8]	0.978	0.935	0.949	-	-	-	0.522	0.396	0.306	-	-	-
DEPICT [19]	0.965	0.917	-	0.392	0.392	-	-	-	-	-	-	-
ClusterGAN [46]	0.905	0.890	-	0.662	0.645	-	-	-	-	-	-	-
DSCDAN [75]	0.978	0.941	-	0.662	0.645	-	-	-	-	-	-	-
DCCM [70]	-	-	-	-	-	-	0.623	0.496	0.408	-	-	-
SSC-OMP [81]	0.928	0.842	0.849	0.274	0.421	0.196	0.326	0.498	0.196	0.654	0.661	0.634
NCSC [89]	0.941	0.861	0.875	**0.721**	0.686	**0.592**	-	-	-	-	-	-
EnSC [79]	**0.980**	**0.945**	**0.957**	0.672	0.705	0.565	0.613	0.601	0.430	T	T	T
ESC [77]	0.971	0.925	0.936	0.668	**0.708**	0.556	0.653	0.629	0.438	**0.732**	**0.825**	0.759
SENet	0.968	0.918	0.931	0.697	0.663	0.556	**0.765**	**0.655**	**0.573**	0.721	0.798	**0.766**



✓ SENet can **effectively handle large scale datasets** with 60k (CIFAR-10), 70k (MNIST & Fashion-MNIST) and ~190k (EMNIST) images

Code link: https://github.com/zhangsz1998/Self-Expressive-Network