Clustering Complex Data with Group-Dependent Feature Selection

Yen-Yu Lin[1,2], Tyng-Luh Liu[1], and Chiou-Shann Fuh[2]

[1]Institute of Information Science, Academia Sinica, Taiwan; [2]Dept. of CSIE, National Taiwan University, Taiwan

1. Summary

- We propose a new clustering technique that considers multiple feature representations and cluster-specific feature selection for handling complex data.
- Our approach:
  - Associate each cluster with a classifier that implements multiple kernel learning (MKL) in a boosting way.
  - Each cluster-specific classifier is applied to feature selection to best separate data of the cluster from the rest.
  - Integrate the multiple, correlative training tasks of the cluster-specific classifiers into the clustering procedure.
  - It supports both unsupervised and semi-supervised clustering.

2. The Proposed Approach

2.1 Problem Definition

- Our goal is to divide a given dataset $D = \{x_i\}_{i=1}^N$ into C clusters.
- Multiple image descriptors are used, and kernel matrices serve as the unified feature representation.
- Two sets of variables are optimized in our clustering formulation.
  - $Y \in \{0,1\}^{N \times C}$: the partition matrix used to represent the clustering result.
  - $\{f_c\}_{c=1}^C$: the cluster-specific classifiers. f_c will select features such that data of cluster c are coherent to each other, and distinct to the rest.
- The cause-and-effect factor:
  - Learning classifiers $\{f_c\}_{c=1}^C$ requires data labels provided by partition matrix Y.
  - Partition matrix Y is determined by considering the cluster structure revealed via $\{f_c\}_{c=1}^C$.
  - Thus, we cast them as a joint optimization problem:

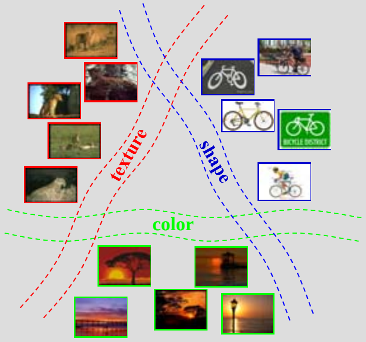
$$\min_{Y, \{f_c\}_{c=1}^C} \sum_{c=1}^{C} \text{Loss}(f_c; \{x_i, y_{ic}\}_{i=1}^N) \quad \text{--> LogLoss}$$

$$\begin{aligned}
\text{s.t.}\quad & Y \in \{0,1\}^{N \times C}, \\
& y_{i,:} e_C = 1, \text{ for } i = 1, 2, \dots, N; && \text{--> partition matrix} \\
& \ell \le e_N^\top y_{:,c} \le u, \text{ for } c = 1, 2, \dots, C; && \text{--> cluster sizes} \\
& y_{i,:} = y_{j,:}, \text{ if } (i,j) \in S; && \text{--> must-links} \\
& y_{i,:} \neq y_{j,:}, \text{ if } (i,j) \in S'. && \text{--> cannot-links}
\end{aligned}$$

2.2 Optimization

- An alternative optimization procedure is used to solve the joint optimization problem.
  - By fixing Y, each cluster-specific classifier f_c is trained by using a boosting-based MKL. f_c is derived by selecting features to best separate data residing in cluster c from the rest.
  - By fixing $\{f_c\}_{c=1}^C$, partition matrix Y is optimized by solving a binary integer programming (BIP) problem.

Cluster-dependent Feature Selection



Our Approach

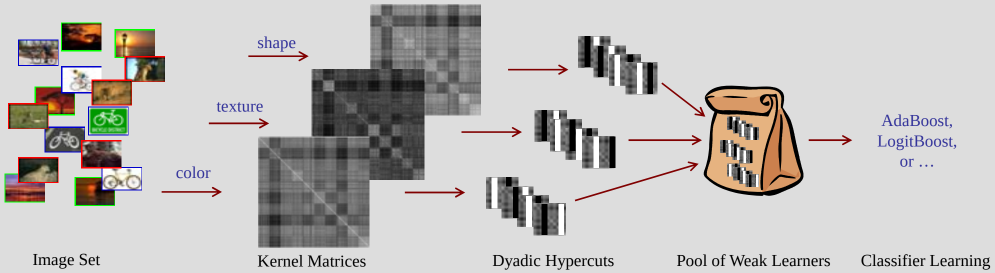
- Learn a classifier f_c for each cluster c.
- f_c will select features cross different descriptors to best separate data of cluster c from the rest.

On Learning Cluster-specific Classifiers $\{f_c\}_{c=1}^C$ via a Boosting-based MKL

On Getting the Clustering Result Y via Binary Integer Programming

- Assign data into clusters such that the total induced loss of all cluster-specific classifiers is minimized.

Boosting-based Multiple Kernel Learning for Feature Selection



3. Multiple Kernel Learning via Boosting

- We carry out multiple kernel learning in a boosting way.
  - Transfer the discriminant power of each kernel into a set of weak learners, called dyadic hypercuts [Moghaddam et al., NIPS'02].
  - Learn the boosting classifier over weak learners generated from all the kernels.

3.1 Weak Learners: Dyadic Hypercuts

- A dyadic hypercut is specified by a kernel and a pair of training samples of opposite labels:

$$h(x) = \text{sign}(k(x_p, x) - k(x_n, x) - \mu).$$

- Dyadic hypercuts capture useful information in the kernel.

3.2 Classifier Learning with Multiple Kernels

- Learn the classifiers by one of the boosting algorithms, such as AdaBoost, LogitBoost, or AnyBoost.
- It supports incremental/on-line classifier learning.

4. Experimental Results

- The proposed approach is evaluated on two vision applications: visual object categorization and face image grouping.

4.1 Visual Object Categorization

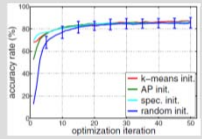
- Following the setting in [Dueck et al., ICCV'07], we select the same twenty object categories form the Caltech-101 dataset.
- We randomly pick thirty images from each category to form a set of 600 images.
- Five kinds of image descriptors are implemented, and they result in the following five kernel matrices:
  - GB: Based on the geometric blur descriptor.
  - SIFT: Based on the SIFT descriptor.
  - SS: Based on the self-similarity descriptor.
  - C2: Based on the biologically inspired features.
  - PHOG: Based on the PHOG descriptor.
- Clustering performances are evaluated by accuracy (ACC) and normalized mutual information (NMI).
- When each descriptor (kernel) is considered individually, …

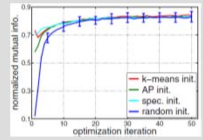
kernel	k-means	Affinity Prop.	Spectral Clus.	Ours
GB	68.0 / 0.732	52.5 / 0.578	69.5 / 0.704	**75.0 / 0.742**
SIFT	62.5 / 0.680	59.8 / 0.638	62.5 / 0.668	**69.6 / 0.706**
SS	**65.7 / 0.659**	55.7 / 0.574	63.3 / 0.655	62.1 / 0.639
C2	37.8 / 0.417	47.5 / 0.517	**57.7 / 0.585**	51.2 / 0.550
PHOG	53.3 / 0.547	43.3 / 0.464	**61.0 / 0.624**	55.2 / 0.569

  - In form of [ACC (%) / NMI]
- When all descriptors (kernels) are considered jointly, …

kernels	CE + k-means	CE + Affinity Prop.	CE + Spectral Clus.	Ours
All	73.8 / 0.737	63.3 / 0.654	77.3 / 0.758	**85.7 / 0.833**

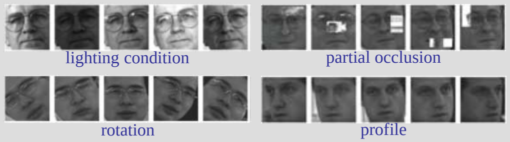
  - CE: Cluster Ensemble [Strehl et al., JMLR'02]
- Our approach works with different initializations.





4.2 Face Image Grouping

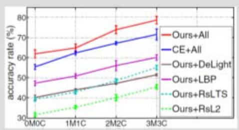
- The CMU PIE database is used.
  - We divide the 68 people into four equal-size disjoint groups, each of which contains images reflecting one kind of variations.

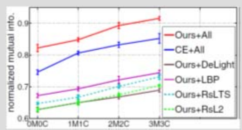


- Four kinds of image descriptors are implemented:
  - DeLight: Based on the delighting algorithm [Gross et al., 2003].
  - LBP: The rotation-invariant LBP operator [Ojala et al., 2000].
  - RsLTS: Least trimmed squares with 20% outliers allowed.
  - RsL2: Pixel intensities with Euclidean distance.
- Clustering performances [ACC (%) / NMI]:

method	kernel(s)	dataset (number of classes)				
		All (68)	Lighting (17)	Rotation (17)	Occlusion (17)	Profile (17)
Ours	DeLight	40.2 / 0.628	**91.4 / 0.974**	21.0 / 0.435	25.5 / 0.508	23.0 / 0.487
	LBP	**47.3 / 0.672**	71.1 / 0.886	**59.9 / 0.744**	30.0 / 0.500	**28.2 / 0.512**
	RsLTS	39.3 / 0.647	35.4 / 0.518	32.9 / 0.495	**61.4 / 0.757**	27.6 / 0.492
	RsL2	31.6 / 0.628	50.9 / 0.685	27.6 / 0.464	19.5 / 0.352	28.4 / 0.509
CE	All	55.4 / 0.746	92.6 / 0.975	43.8 / 0.657	55.4 / 0.695	29.8 / 0.535
Ours	All	**61.9 / 0.822**	**93.6 / 0.985**	**57.8 / 0.730**	**64.8 / 0.781**	**31.6 / 0.554**

- Our approach also works with partially labeled data.
  - Randomly generate must-links and cannot-links for each subject.





  - *x*M*y*C stands for *x* **m**ust-links and *y* **c**annot-links per subject.