JOINT STRUCTURED GRAPH LEARNING AND UNSUPERVISED FEATURE SELECTION

Yong Peng, Leijie Zhang, Wanzeng Kong, Feiping Nie and Andrzej Cichocki
yongpeng@hdu.edu.cn

Abstract

The central task in graph-based unsupervised feature selection (GUFS) depends on two folds, one is to accurately characterize the geometrical structure of the original feature space with a graph and the other is to make the selected features well preserve such intrinsic structure. Currently, most of the existing GUFS methods use a two-stage strategy which constructs graph first and then perform feature selection on this fixed graph. Since the performance of feature selection severely depends on the quality of graph, the selection results will be unsatisfactory if the given graph is of low-quality. To this end, we propose a joint graph learning and unsupervised feature selection (JGUFS) model in which the graph can be adjusted to adapt the feature selection process. The JGUFS objective function is optimized by an efficient iterative algorithm whose convergence and complexity are analyzed in detail. Experimental results on representative benchmark data sets demonstrate the improved performance of JGUFS in comparison with state-of-the-art methods and therefore we conclude that it is promising of allowing the feature selection process to change the data graph.

Model Formulation

$$\min_{\mathbf{S},\mathbf{W},\mathbf{F}} \|\mathbf{S}-\mathbf{A}\|_F^2 + \alpha \mathbf{Tr}(\mathbf{F}^T\mathbf{L_s}\mathbf{F}) + \beta(\|(\mathbf{XW}-\mathbf{F}\|_F^2 + \gamma\|\mathbf{W}\|_{2,1}) \quad (1)$$
$$s.t.\ \mathbf{S1}=\mathbf{1}, \mathbf{S}\geq\mathbf{0}, \mathbf{F}^T\mathbf{F}=\mathbf{I}_c, \mathbf{F}\geq\mathbf{0}$$

where $\mathbf{X}\in\mathbb{R}^{n\times d}$ is the data matrix, $\mathbf{W}\in\mathbb{R}^{d\times c}$ is the projection matrix, β and γ are regularization parameters. Similar to [1, 2], we impose the non-negativity on F here

Conclusion

In this paper, we proposed a novel GUFS method, termed JGUFS, which simultaneously performs graph construction and feature selection. Instead of performing feature selection on a fixed graph, JGUFS successfully avoided the disadvantages caused by the two-stage strategy. In JGUFS, the subobjectives respectively corresponding to graph construction and unsupervised feature selection could co-evolve towards the optimum. An efficient iterative optimization method with convergence guarantee was presented to optimize the JGUFS objective. Extensive experiments were conducted on representative data sets to demonstrate the excellent performance of JGUFS in comparison with state-of-the-art methods.

References

Performance in Feature Selection

Table 1: Comparison of of clustering for different feature selection methods (ACC/NMI±std).

ACC	JAFFE	UMIST	USPS	MNIST	COIL20	WebKB	ISOLET
All-Fea	72.1±3.3	42.9±2.8	63.7±4.1	51.8±4.7	61.7±2.4	55.9±3.1	57.4±3.9
MaxVar	76.3±2.9	46.7±2.4	64.9±3.1	53.0±2.9	61.1±2.8	54.8±2.3	56.9±2.7
LapScore	77.2±3.2	45.8±3.0	64.1±3.2	53.9±3.5	62.1±2.1	56.1±2.8	56.8±2.9
MCFS	79.5±2.7	46.7±3.1	65.1±4.7	55.9±3.7	60.9±2.3	61.5±2.3	60.9±2.5
FSSL	85.6±2.2	51.9±3.3	66.5±2.4	57.1±3.8	62.5±2.8	62.3±2.7	64.9±3.1
UDFS	84.7±2.3	48.9±3.8	66.3±3.0	56.7±3.2	60.8±2.7	61.9±2.9	64.7±3.6
NDFS	86.9±2.5	51.1±3.7	66.9±2.7	58.5±2.8	63.3±2.1	62.5±3.0	65.1±3.9
JELSR	86.5±2.3	53.7±3.2	67.8±2.9	58.1±3.1	64.8±1.9	61.8±2.9	63.7±2.8
JGUFS	88.3±2.4	57.8±2.6	69.7±2.8	59.3±3.0	68.9±1.6	63.8±2.7	66.8±3.2
NMI	JAFFE	UMIST	USPS	MNIST	COIL20	WebKB	ISOLET
All-Fea	78.9±2.1	63.5±2.2	59.7±1.8	46.3±2.1	73.5±2.8	11.7±4.2	73.9±1.7
MaxVar	80.3±2.0	65.1±2.0	60.9±1.5	47.9±2.3	71.8±3.1	16.9±2.1	73.7±1.8
LapScore	81.9±1.8	64.7±2.6	60.3±1.3	48.3±2.0	73.9±2.9	13.4±3.5	72.1±1.1
MCFS	82.3±1.8	65.6±1.8	61.7±1.5	50.3±1.7	74.8±2.3	18.3±3.7	74.9±1.6
FSSL	88.6±1.3	67.7±2.0	62.3±1.3	50.8±2.1	75.1±2.7	18.5±3.5	76.8±1.7
UDFS	85.3±2.0	66.5±2.1	61.8±1.5	50.1±1.5	75.7±1.9	17.1±2.9	76.3±1.9
NDFS	87.6±1.9	68.9±2.5	61.3±1.1	51.6±1.1	77.3±1.8	17.6±2.7	78.4±1.2
JELSR	86.9±2.1	70.3±1.7	62.0±1.3	51.1±1.4	77.9±1.7	18.0±3.1	75.8±1.1
JGUFS	89.8±0.6	73.9±2.1	63.9±1.1	52.9±1.0	79.8±1.3	20.3±2.3	79.9±1.2

Optimization

With other two variables fixed, the following formula can be proved:

$$\mathcal{O}(\mathbf{F}^{t+1}, \mathbf{W}^t\mathbf{S}^t) \leq \mathcal{O}(\mathbf{F}^t, \mathbf{W}^t\mathbf{S}^t),$$
$$\mathcal{O}(\mathbf{F}^{t+1}, \mathbf{W}^{t+1}\mathbf{S}^t) \leq \mathcal{O}(\mathbf{F}^{t+1}, \mathbf{W}^t\mathbf{S}^t)$$
$$\mathcal{O}(\mathbf{F}^{t+1}, \mathbf{W}^{t+1}\mathbf{S}^{t+1}) \leq \mathcal{O}(\mathbf{F}^{t+1}, \mathbf{W}^{t+1}\mathbf{S}^t)$$

We conclude that JGUFS objective function monotonically decreases under the optimization in Algorithm. 1.

Algorithm 1 Optimization to JGUFS objective function

Input: Data matrix $\mathbf{X}\in\mathbb{R}^{n\times d}$, λ, β, and γ, c, the dimension of projected subspace c;

Output: Rank features based on the values of $\|w_i\|_2|_{i=1}^d$ in descending order and then select the top-ranked ones.

1: Initialization. Construct the initial graph affinity matrix $\mathbf{A}$ based on the 'HeatKernel' function; Calculate $\mathbf{F}\in\mathbb{R}^{n\times c}$ by the c eigenvectors of the graph Laplacian $\mathbf{L_A} = \mathbf{D_A} - \frac{\mathbf{A}^T+\mathbf{A}}{2}$ corresponding to the c smallest eigenvalues; Initialize $\mathbf{M}\in\mathbb{R}^{d\times d}$ as an identity matrix;;

2: **while** not converged **do**

3: Update $\mathbf{S}$ by solving:

$$\min_{S_i\mathbf{1}=1, s_i\geq 0} \|s_i - (a_i - \frac{\alpha}{2}d_i)\|_F^2,$$

where, $d_{ij} = \|f_i - f_j\|_2^2$ and d_i as a vector with the j-th element equal to d_{ij}. Similarly, we get a_i and s_i.

4: Update $\mathbf{W}$ by:

$$\mathbf{W} = (\mathbf{X}^T\mathbf{X} + \gamma\mathbf{M})^{-1}\mathbf{X}^T\mathbf{F}$$

5: Update $\mathbf{M}$ by:

$$m_{ii} = \frac{1}{2\|w\|_2} = \frac{1}{2\sqrt{w_i w_i^T + \delta}}$$

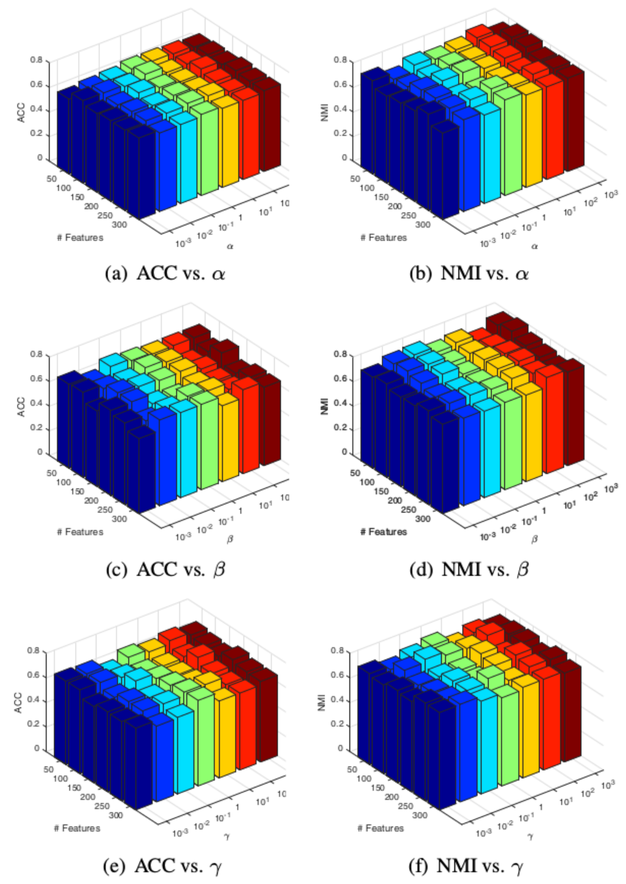
6: Update $\mathbf{F}$ by:

$$d_{ij} \leftarrow \frac{(\lambda\mathbf{F})_{ij}}{\mathbf{RF} + \lambda\mathbf{F}\mathbf{F}^T\mathbf{F}}$$

7: **end while**

Analysis

Figure 1 illustrates the clustering performance of JGUFS on COIL20 with different settings of parameters. From this figure, we find that JGUFS provides excellent performance when the parameters are set as different values in a wide range. Further, we can observe that even if a small number of features are selected, JGUFS can still achieve relatively good clustering results.



(a) ACC vs. α (b) NMI vs. α
(c) ACC vs. β (d) NMI vs. β
(e) ACC vs. γ (f) NMI vs. γ

Figure 1: Performance of JGUFS algorithm for large variation of set of control parameters.

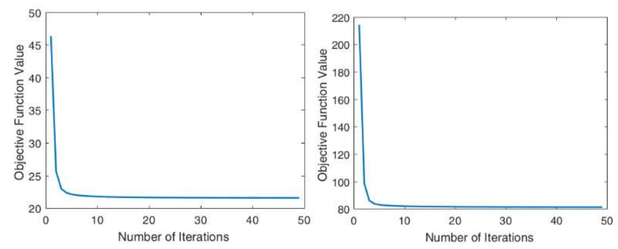


Figure 2: Convergence speed of JGUFS for UMIST and COIL20 data sets.

Figure 2 shows the convergence curves of the JGUFS objective function in terms of the number of iterations on UMIST and COIL20 from which we can observe that JGUFS has a relatively fast convergence speed.