Generalized Canonical Correlation Analysis (GCCA)

Yash Sharma

(0801CS171093)

Contents

1. Introduction

1.1. Abstract

For multiple multivariate datasets, we derive conditions under which Generalized Canonical Correlation Analysis improves classification performance of the projected datasets, compared to standard Canonical Correlation Analysis using only two data sets.

1.2. Introduction

In generalized canonical correlation analysis several sets of variables are analyzed simultaneously. This makes the method suited for the analysis of various types of data. For example, in marketing research, subjects may be asked to rate a set of objects on a set of attributes. For each individual, a data matrix can then be constructed where the objects are represented row-wise and the attributes column wise. Then, using generalized canonical correlation analysis a graphical representation, sometimes referred to as a perceptual map, can be made on the basis of the individual's observation matrices.

This approach has some attractive properties that makes the method well fit for the analysis of multiple-set data. First of all, computationally, the method is straightforward as its solution is based on an Eigen equation. Secondly, the method is closely related to several well-known multivariate techniques. In particular, principal component analysis and multiple correspondence analysis.

1.3. Constraint Functions, h(x, R)

There are four constraints placed on the canonical vectors that are natural to use in MCCA.

a) NORM - The canonical coefficient vectors each have unit norm.

$$h(x, R) = x^H_i x_i = 1, 1 \leq i \leq m$$

This objective function has the same flavor as other machine learning algorithms such as PCA.

b) AVGNORM - The vector of canonical vectors, x, has unit norm.

$$h(x, R) = x^H x = \sum_{i=1}^{m} x^H_i x_i = 1$$

c) VAR - The canonical variates each have unit variance.

$$x^H_i R_{ii} x_i = 1, 1 \leq i \leq m.$$

This is the natural extension of the CCA constraint functions.

d) AVGVAR - The canonical variates have average variance of 1/m.

$$\sum_{i=1}^{m} x^H_i R_{ii} \; x_i = 1$$

This may be written $tr(X^H RX) = 1$, where $X = blkdiag(x_1, . . . , x_m)$.

1.4. Objective Functions, J(Φ(x))

1. SUMCORR - Maximize the sum of the correlations between each of the canonical variates.

$$J(\Phi(x)) = \max_{(x1,\ldots,xm)} \sum_j^m \sum_i^m x_i^H R_{ij}\, x_j = max_{x1,\ldots xm}\ 1^H \Phi(x) 1$$

This is the natural extension of the CCA objective function.

2. SSQCORR - Maximize the sum of the squares of the correlations between each of the canonical variates.

$$J(\Phi(x)) = \max_{(x1,\ldots,xm)} \sum_j^m \sum_i^m (x_i^H R_{ij}\, x_j)^2 = max_{x1,\ldots xm}\ ||\ \Phi(x)||_F^2$$
$$= \max_{(x1,\ldots xm)} \sum_{i=1}^{m} \lambda_i^2(\ \Phi(x))$$

where λi are the eigenvalues of φ(x).

3. MAXVAR - Maximize the largest eigenvalue of Φ, λ1(Φ(x)).

$$J(\Phi(x)) = \max_{(x1,\ldots,xm)}\ \lambda_1(\Phi(x))$$

MAXVAR was created to find the canonical vectors that give Φ(x) the best approximation to a rank-1 matrix.

4. MINVAR - Minimize the smallest eigenvalue of Φ, λm(Φ(x)).

$$J(\Phi(x)) = \min_{(x1,\ldots,xm)}\ \lambda_d(\Phi(x))$$

Instead of maximizing the energy in the top eigenvalue, we wish to minimize the energy in the last eigenvalue.

5. GENVAR - Minimize the generalized variance of w, which is equivalent to minimizing the determinant of the correlation matrix of w.

$$J(\Phi(x)) = \min_{(x1,\ldots,xm)} |\Phi(x)| = \min_{(x1,\ldots,xm)} \prod_{i=1}^{m} \lambda_i(\Phi(x))$$

2. Mathematical Formulation

2.1. Formulation (for SUMCOR – GCCA)

CCA maximizes the correlation between two views. For more than two views, a natural extension is to maximize the sum of the pairwise correlations, which is the idea of SUMCOR-GCCA.

Let $\{r_j \epsilon\ R^{d_j \times N}\}_{j=1}^{J}$ be a dataset containing J meanzero views, where d_j is the feature dimensionality of View j. The objective function of SUMCOR-GCCA is:

$$\max(\boldsymbol{x}_i)_{i=1}^{J} \ \sum_j^J \sum_i^J (\boldsymbol{x}_i^T\ \boldsymbol{R}_i \boldsymbol{R}_j^T\ \boldsymbol{x}_j)^2$$

$$\text{s.t. } \boldsymbol{x}_j^T\ \boldsymbol{R}_j \boldsymbol{R}_j^T \boldsymbol{x}_j = 1,\ \ j = 1, 2, \cdots, J.$$

2.2. Manopt Software for Optimization on Manifolds

Many of the problems discussed in Appendix B do not yield closed form solutions because either the cost function is unwieldy or because the constraint functions complicate the derivations. For these problems we use the Manopt software provided at www.manopt.org. The Manopt software specializes in solving constrained optimization problems when the constraints are manifolds. This software package is able to solve nonlinear optimization problems.

After selecting the appropriate manifold and providing the cost and gradient functions, we use the trustregions solver to find a solution for our problems. This returns the minimized cost and the point that achieved the minimum cost. If our objective function has a cost function that seeks a maximum, we provide the negative of the true cost function and the gradient is computed from this negative cost.

3. Documentation of API

3.1. Package Organization:

Parameters:

n_components (int), (default 2).

reg_param (float) : It is the hyper parameter which is set by us to avoid overfitting and underfitting.

x_list = more than 2 mean – zero data matrix N instances

Attributes:

view_len = number of views.

cov_mat = covariance matrix for different views.

eigvals = list of eigen value matrices.

Vxx = covariance matrix of view x with respect of view x.

Vyy = covariance matrix of view y with respect of view y.

Vxy = covariance matrix of view x with respect of view y.

U, S, V = output of singular value decomposition(SVD).

wx, wy = Canonical matrices.

3.2. Methods:

fit(self, *x_list) : fit model to data.

transform(self, *x_list) : Apply the dimension reduction learned on the trained data.

cal_cov_mat(self, *x_list) : calculate covariance matrix of metirices.

cal_eig(self, *x_list) : calculate and sort eigen values and eigen vectors

normalize(mat) : normalize the given matrix.

4. Example

4.1. Example 1 :

X = [[9 6 8], [5 9 4], [8 9 3]]

Y = [[9 8 6], [2 2 8], [4 1 6]]

V_{xx} = [[56.66666667 57. 38.66666667]
[57. 66. 37.]
[38.66666667 37. 29.66666667]]

V_{yy} = [[33.66666667 26.66666667 31.33333333]
[26.66666667 23. 23.33333333]
[31.33333333 23.33333333 45.33333333]]

V_{xy} = [[41. 30. 47.33333333]
[36. 25. 54.]
[30.66666667 25. 32.66666667]]

W_x = [[-0.24708491 -0.00049486 0.00133797]
[-0.24421733 0.0089494 -0.00045317]
[-0.29406964 -0.01088892 -0.00120086]]

W_y = [[-0.31463455 -0.01090497 0.00201159]
[-0.35119495 -0.02222785 -0.00185324]
[-0.31037719 0.02228451 -0.00050822]]

5. Learning Outcome

We are able to learning different optimizations that can be done in order to improve correlation among different sets of variables. This can be done with the use of different objective functions and constraint functions. There are total 4 constraint functions named as Norm, AVGNORM, VAR, AVGVAR and 5 objective functions as SUMCORR, SSQCORR, MAXVAR, MINVAR, GENVAR so combining these function total 4*5 = 20 optimizations can be possible by considering any of the objectives and constraint function.

6. Bibliography

[1] N. A. Asendorf, “Informative data fusion: Beyond canonical correlation analysis,” Ph.D. dissertation, The University of Michigan, Ann Arbor, MI, 2015.

[2] N. Boumal, B. Mishra, P.-A. Absil, and R. Sepulchre, “Manopt, a matlab toolbox for optimization on manifolds,” Journal of Machine Learning Research, vol. 15, pp. 1455–1459, 2014.

[3] J. Kettenring, “Canonical analysis of several sets of variables,” Biometrika, vol. 58, no. 3, pp. 433–451, 1971.

[4] Fu-Chang Liu, Quan-Sen Sun, Jian Zhang, “Generalized Canonical Correlation Analysis Using GSVD,” , 2008.