Learning Multiple Tasks with a Sparse Matrix-Normal Penalty

Yi Zhang and Jeff Schneider

NIPS 2010

Presented by Esther Salazar
Duke University

March 25, 2011

Summary

- **Contribution:** The authors propose a matrix-variate normal penalty with sparse inverse covariances to couple multiple tasks

- The penalty is decomposed into the Kronecker product of row covariance and column covariance with characterize both task and features

- Overfitting and selection of meaningful task and feature structures: They include sparse covariance selection into the matrix normal regularization via ℓ_1 penalties [1] on task and feature inverse covariances

- Empirical studies using two real-world problems:
  - detecting landmines in multiple files and
  - recognizing faces between different subjects

[1] In general ℓ_p-norm of a $(m \times n)$-matrix A: $||A||_p = \left(\sum_{i=1}^{m}\sum_{j=1}^{n}|a_{ij}|^p\right)^{1/p}$

Background

Multi-tasks learning (related work):

- learning a common feature representation shared by tasks (principal components, to select a common subset of features, to used hidden nodes in neural networks)
- directly inferring the relatedness of tasks (mixtures of Gaussians or DPs to model tasks groups, identifying outlier tasks by robust t-processes)

Proposal. Estimate a matrix of model parameters where the rows and columns correspond to tasks and features

Regularization. The authors propose a new regularization approach and show how previous approaches are special cases

Matrix-Variate Normal Distributions

Consider an $m \times p$ matrix W. The vectorized matrix Vec(W) follows a multivariate normal distribution

$$\text{Vec}(W) \sim N(\text{Vec}(M), \Sigma \otimes \Omega)$$

where M $(m \times p)$ is the mean matrix, Ω $(m \times m)$ is the row covariance and Σ $(p \times p)$ is the column covariance.

Log-density:

$$\log P(\mathbf{W}) = -\frac{mp}{2}\log(2\pi) - \frac{p}{2}\log(|\mathbf{\Omega}|) - \frac{m}{2}\log(|\mathbf{\Sigma}|) - \frac{1}{2}tr\{\mathbf{\Omega}^{-1}(\mathbf{W}-\mathbf{M})\mathbf{\Sigma}^{-1}(\mathbf{W}-\mathbf{M})^T\}$$

Maximum Likelihood Estimation

Consider a set of n samples $\{W_i\}_{i=1}^{n}$ generated by a MVN distribution

MLE of M is: $\hat{M} = \frac{1}{n}\sum_{i=1}^{n} W_i$

MLE of Ω and Σ are solutions to the system

$$\begin{cases} \hat{\mathbf{\Omega}} & = & \frac{1}{np}\sum_{i=1}^{n}(\mathbf{W}_i - \hat{\mathbf{M}})\hat{\mathbf{\Sigma}}^{-1}(\mathbf{W}_i - \hat{\mathbf{M}})^T \\ \hat{\mathbf{\Sigma}} & = & \frac{1}{nm}\sum_{i=1}^{n}(\mathbf{W}_i - \hat{\mathbf{M}})^T\hat{\mathbf{\Omega}}^{-1}(\mathbf{W}_i - \hat{\mathbf{M}}) \end{cases}$$

Problems: $\hat{\Omega}$ and $\hat{\Sigma}$ are not identifiable and solutions are not unique. For example, for any $\alpha > 0$, $(\alpha\Omega^*, \frac{1}{\alpha}\Sigma^*)$ will lead to same log density. That means that only the Kronecker product $\Sigma \otimes \Omega$ is identifiable.

Multi-task learning with a sparse matrix-normal penalty

Classical regularization penalties: to assume a multivariate prior on the parameter vector and to perform maximum-a-posterior estimation (ℓ_2 penalty: multivariate Gaussian; ℓ_1 penalty: Laplacian priors[2])

For multi-task learning the use of matrix-variate priors is natural to design regularization penalties

[2] $p(x_i) = \frac{\lambda}{2} \exp(-\lambda |x_i|)$

Multi-task learning with a sparse matrix-normal penalty

Matrix normal penalty

Consider a multi-task learning problem with m tasks in a p-dimensional feature space. The training sets are $\{\mathbf{D}_t\}_{t=1}^m$, where each set $\mathbf{D}_t$ contains n_t examples $\{(\mathbf{x}_i^{(t)}, y_i^{(t)})\}_{i=1}^{n_t}$. We want to learn m models for the m tasks but appropriately share knowledge among tasks. Model parameters are represented by an $m \times p$ matrix $\mathbf{W}$, where parameters for a task correspond to a row.

The total loss to optimize is:

$$\mathcal{L} = \sum_{t=1}^{m} \sum_{i=1}^{n_t} L(y_i^{(t)}, \mathbf{x}_i^{(t)}, \mathbf{W}(t,:)) + \lambda\, tr\{\mathbf{\Omega}^{-1}\mathbf{W}\mathbf{\Sigma}^{-1}\mathbf{W}^T\} \tag{5}$$

where λ controls the strength of the regularization and $L()$ is the convex empirical loss function depending of the specific model we use (square loss for linear regression log-likelihood for logistic regression and so forth)

Special cases regarding to the loss function:

- When we fix $\Omega = I_m$ and $\Sigma = I_p$, the penalty term can be decomposed into standard ℓ_2 norm penalties on the m rows of W
- When we fix $\Omega = I_m$, task are linked only by a shared feature covariance Σ. Additional constrain $tr\{\Sigma\} <= 1$ to avoid $\Sigma \to \infty$
- When we fix $\Sigma = I_p$, tasks are coupled only by a task similarity matrix Ω

We will like to infer Ω and Σ. If we do that jointly we will always set Ω and Σ to be infinity matrices. To avoid that

$$\mathcal{L} = \sum_{t=1}^{m} \sum_{i=1}^{n_t} L(y_i^{(t)}, \mathrm{x}_i^{(t)}, \mathbf{W}(t,:)) + \lambda \left[p \log |\mathbf{\Omega}| + m \log |\mathbf{\Sigma}| + tr\{\mathbf{\Omega}^{-1} \mathbf{W} \mathbf{\Sigma}^{-1} \mathbf{W}^T\} \right]$$

We can infer Ω and Σ as the following problem

$$\min_{\mathbf{\Omega}, \mathbf{\Sigma}} p \log |\mathbf{\Omega}| + m \log |\mathbf{\Sigma}| + tr\{\mathbf{\Omega}^{-1} \mathbf{W} \mathbf{\Sigma}^{-1} \mathbf{W}^T\}$$

Then, the MLE of Ω and Σ is

$$\begin{cases} \hat{\mathbf{\Omega}} &= \frac{1}{p} \mathbf{W} \hat{\mathbf{\Sigma}}^{-1} \mathbf{W}^T + \epsilon \mathbf{I}_m \\ \hat{\mathbf{\Sigma}} &= \frac{1}{m} \mathbf{W}^T \hat{\mathbf{\Omega}}^{-1} \mathbf{W} + \epsilon \mathbf{I}_p \end{cases}$$

where $\epsilon > 0$ (small number)

Sparse covariance selection

Consider the sparsity of Ω^{-1} and Σ^{-1}. Covariance selection aims to select nonzero entries in the Gaussian inverse covariance and discover conditional independence between variables. The idea is to include two additional ℓ_1 penalty terms on the inverse covariances:

$$\mathcal{L} = \sum_{t=1}^{m}\sum_{i=1}^{n_t} L(y_i^{(t)}, \mathbf{x}_i^{(t)}, \mathbf{W}(t,:)) + \lambda[p\log|\mathbf{\Omega}| + m\log|\mathbf{\Sigma}| + tr\{\mathbf{\Omega}^{-1}\mathbf{W}\mathbf{\Sigma}^{-1}\mathbf{W}^T\}]$$
$$+ \lambda_\Omega||\mathbf{\Omega}^{-1}||_{\ell 1} + \lambda_\Sigma||\mathbf{\Sigma}^{-1}||_{\ell 1} \qquad (9)$$

λ_Ω and λ_Σ control the strength of ℓ_1 penalties and therefore the sparsity of task and feature structures

We can iteratively optimize Ω and Σ until convergence

$$\begin{cases} \hat{\mathbf{\Omega}} & = & \text{argmin}_{\mathbf{\Omega}}\ p\log|\mathbf{\Omega}| + tr\{\mathbf{\Omega}^{-1}(\mathbf{W}\mathbf{\Sigma}^{-1}\mathbf{W}^T)\} + \frac{\lambda_\Omega}{\lambda}||\mathbf{\Omega}^{-1}||_{\ell 1} \\ \hat{\mathbf{\Sigma}} & = & \text{argmin}_{\mathbf{\Sigma}}\ m\log|\mathbf{\Sigma}| + tr\{\mathbf{\Sigma}^{-1}(\mathbf{W}^T\hat{\mathbf{\Omega}}^{-1}\mathbf{W})\} + \frac{\lambda_\Sigma}{\lambda}||\mathbf{\Sigma}^{-1}||_{\ell 1} \end{cases}$$

We can use graphical lasso as a basic solver and (11) as a ℓ_1 regularized "flip-flop" algorithm

$$\begin{cases} \hat{\mathbf{\Omega}} & = & glasso(\frac{1}{p}\mathbf{W}\hat{\mathbf{\Sigma}}^{-1}\mathbf{W}^T, \frac{\lambda_\Omega}{\lambda}) \\ \hat{\mathbf{\Sigma}} & = & glasso(\frac{1}{m}\mathbf{W}^T\hat{\mathbf{\Omega}}^{-1}\mathbf{W}, \frac{\lambda_\Sigma}{\lambda}) \end{cases}$$

Algorithm:

1) Estimate $\mathbf{W}$ by solving (5), using $\mathbf{\Omega} = \mathbf{I}_m$ and $\mathbf{\Sigma} = \mathbf{I}_p$;

2) Infer $\mathbf{\Omega}$ and $\mathbf{\Sigma}$ in (9) (by solving (11) until convergence), using the estimated $\mathbf{W}$ from step 1);

3) Estimate $\mathbf{W}$ by solving (5), using the inferred $\mathbf{\Omega}$ and $\mathbf{\Sigma}$ from step 2).

Additional constrains:

- To ignore variances and restrict our attention to correlation structures

$$\begin{aligned} \Omega_{ii} &= 1 \quad i = 1, 2, \ldots, m \\ \Sigma_{jj} &= 1 \quad j = 1, 2, \ldots, p \end{aligned}$$

  diagonal entries may be fixed as a constant if we prefer tasks to be equally regularized

- If one wants to iterative over steps 2) and 3) of the algorithm until convergence, we may consider the constraints

$$\begin{aligned} \Omega_{ii} &= c_1 \quad i = 1, 2, \ldots, m \\ \Sigma_{jj} &= c_2 \quad j = 1, 2, \ldots, p \end{aligned}$$

Empirical Studies

Data sets:

- **The landmine detection data set** from Y. Xue, X. Liao, L. Carin, and B. Krishnapuram(2006). Each example in the data set is represented by a 9-dimensional feature vector extracted from radar imaging. They jointly learn 19 tasks. The model parameters W are 19×10 matrix (19 tasks and 10 features including intercept).
- **The face recognition data set** is the Yale face database, which contains 165 images of 15 subjects. We use the first 8 subjects to construct 28 binary classification tasks, each to classify two subjects

Models

- **STL:** learn ℓ_2 regularized logistic regression for each task separately
- **MTL-C:** clustered multi-task learning, which encourages task clustering in regularization
- **MTL-F:** multi-task feature learning, which corresponds to $\Omega = I_m$

Different configurations of the proposed framework:

- **MTL(**$I_m \& I_p$**)**
- **MTL(**$\Omega \& I_p$**)**
- **MTL(**$I_m \& \Sigma$**)**
- **MTL(**$\Omega \& \Sigma$**)**
- **MTL(**$\Omega \& \Sigma$**)**$_{\Omega_{ii}=\Sigma_{jj}=1}$
- **MTL(**$\Omega \& \Sigma$**)**$_{\Omega_{ii}=1}$

Results on Landmine Detection

Avg AUC Score	30 samples	40 samples	80 samples	160 samples
STL	64.85(0.52)	67.62(0.64)	71.86(0.38)	76.22(0.25)
MTL-C [21]	67.09(0.44)	68.95(0.40)	72.89(0.31)	76.64(0.17)
MTL-F [2]	72.39(0.79)	74.75(0.63)	77.12(0.18)	**78.13(0.12)**
MTL($\mathbf{I}_m$&$\mathbf{I}_p$)	66.10(0.65)	69.91(0.40)	73.34(0.28)	76.17(0.22)
MTL(Ω&$\mathbf{I}_p$)	**74.88(0.29)**	75.83(0.28)	76.93(0.15)	77.95(0.17)
MTL($\mathbf{I}_m$&Σ)	72.71(0.65)	74.98(0.32)	**77.35(0.14)**	**78.13(0.14)**
MTL(Ω&Σ)	**75.10(0.27)**	76.16(0.15)	**77.32(0.24)**	**78.21(0.17)***
MTL$(\Omega\&\Sigma)_{\Omega_{ii}=\Sigma_{jj}=1}$	**75.31(0.26)***	**76.64(0.13)***	**77.56(0.16)***	**78.01(0.12)**
MTL$(\Omega\&\Sigma)_{\Omega_{ii}=1}$	**75.19(0.22)**	76.25(0.14)	**77.22(0.15)**	**78.03(0.15)**

Table 1: Average AUC scores (%) on landmine detection: means (and standard errors) over 30 random runs. For each column, the best model is marked with * and competitive models (by paired t-tests) are shown in **bold**.

Results on Face Recognition

Avg Classification Errors	3 samples per class	5 samples per class	7 samples per class
STL	10.97(0.46)	7.62(0.30)	4.75(0.35)
MTL-C [21]	11.09(0.49)	7.87(0.34)	5.33(0.34)
MTL-F [2]	10.78(0.60)	6.86(0.27)	4.20(0.31)
MTL($\mathbf{I}_m$&$\mathbf{I}_p$)	10.88(0.48)	7.51(0.28)	5.00(0.35)
MTL($\boldsymbol{\Omega}$&$\mathbf{I}_p$)	9.98(0.55)	6.68(0.30)	4.12(0.38)
MTL($\mathbf{I}_m$&$\boldsymbol{\Sigma}$)	**9.87(0.59)**	**6.25(0.27)**	4.06(0.34)
MTL($\boldsymbol{\Omega}$&$\boldsymbol{\Sigma}$)	**9.81(0.49)**	**6.23(0.29)**	4.11(0.36)
MTL($\boldsymbol{\Omega}$&$\boldsymbol{\Sigma}$)$_{\Omega_{ii}=\Sigma_{jj}=1}$	**9.67(0.57)***	**6.21(0.28)**	4.02(0.32)
MTL($\boldsymbol{\Omega}$&$\boldsymbol{\Sigma}$)$_{\Omega_{ii}=1}$	**9.67(0.51)***	**5.98(0.29)***	**3.53(0.34)***

Table 2: Average classification errors (%) on face recognition: means (and standard errors) over 30 random runs. For each column, the best model is marked with * and competitive models (by paired t-tests) are shown in **bold**.