Multiple kernel, multi-task learning for BCI applications

G. Gasso
With A. Rakotomamonjy, R. Flamary

NEC Labs
Princeton

05/26/2009

Roadmap

BCI Application

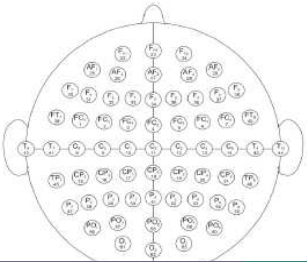
BCI Application

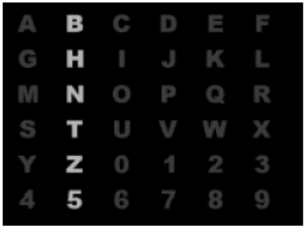
- Brain computer interface
- Recorded P300 Speller Brain Signals
- Characteristics : appearance of a deflection in the EEG signals 300*ms* (P300) after submitting a patient to a stimulus (visual stimulus)
- This deflection corresponds to an evoked potential (P300) to be detected

BCI Applications

Acquisition sytem

- Intensification of row or column of the matrix to spell a letter
- If a row or column containing the target letter is illuminated, a P300 potential is activated
- To recognize a letter, 12 intensifications are done
- To make the recognition reliable, this is repeated 15 times for a letter
- Spelling system consists of 64 EEG channels

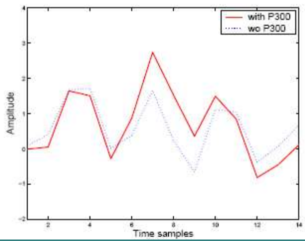




BCI Applications

Dataset

- Different sessions was performed (each session the patient has to recognize a word)
- For one acquisition session, the patient is asked to spell a word of 3 to 5 characters
- 540 to 900 blocks of EEG signals recorded accordingly ($12 \times 15 = 180$ blocks of EEG per character)
- After processing, each EEG is transformed into 14 time frames vector $\Longrightarrow 14 \times 84 = 896$ features vector for one intensification



BCI Applications

Problem setup

- Identify positive signals (with P300) from negative signals
- Select the useful channels or variables
- Handle the variability of the signals over different sessions
- Approach : consider each acquisition session as a task, perform a multi-task learning

Multi-task Learning

Paradigm

- Assume T classification tasks with T datasets

$$\mathcal{D}_t = \left\{(x_i^t, y_i^t)\right\}_{i=1,\cdots,n_t}$$

$$x_i \in \mathcal{X}, \qquad y_i \in \{-1, 1\}, \qquad t = 1, \cdots, T$$

- Tasks are considered similar enough or related in a certain sense
- Aim : learn the decision functions $f_t(x), t = 1, \cdots, T$ in a joint manner
- Tasks share a common subset of relevant features
- Way to ensure this constraint ? Use adequate regularization that favors joint features sparsity pattern across tasks
- For BCI application, the goal is to identify the most important features (from a session to another one)

Multi-task Learning

Formulation

- Consider SVM classifier

$$\min_{f_1,\cdots,f_T} \quad C \cdot \sum_{t=1}^{T} \sum_{i \in \mathcal{D}_t} L(f_t(x_i^t), y_i^t) + \Omega(f_1, \cdots, f_T) \qquad (1)$$

C : regularization parameter

$\mathcal{D}_t$ and $f_t(x)$: dataset and decision function related to task t

$L(y, f(x)) = \max(0, 1 - yf(x))$: hinge loss function

$\Omega(f_1, \cdots, f_T)$: joint sparsity regularizer

- We suppose the decision functions $f_t(x)$ defined as a combination of elementary functions defined over different sets of features $\Longrightarrow$ We adopt a multiple kernel formulation

Multi-task Learning (MTL)

Multi-kernel formulation : simple case

- Suppose we have only one task
- Let suppose three kernel spaces $\mathcal{H}_1$, $\mathcal{H}_1$ and $\mathcal{H}_3$ defined by kernels $k_1(x,z)$, $k_2(x,z)$ and $k_3(x,z)$
- A decision function $f(x)$ takes the form

$$f(x) = f_1(x) + f_2(x) + f_3(x) + b \qquad \text{with} \quad f_m \in \mathcal{H}_m$$

- Example : $f_m(x) = \langle w_m, \phi_m(x) \rangle$ with $\phi_m(x)$ the mapping function
- Each kernel could be defined over a set of features according to some priori knowledge
- Regularizer

$$\Omega(f) = \frac{1}{2} \sum_{m=1}^{3} \|f_m\|_{\mathcal{H}_m} \quad \text{or} \quad \Omega(f) = \frac{1}{2} \left(\sum_{m=1}^{3} \|f_m\|_{\mathcal{H}_m} \right)^2$$

Multi-task Learning (MTL)

Multi-kernel Learning extended to MTL

- Three kernel spaces $\mathcal{H}_m$, $m = 1, \cdots, 3$ and four tasks with functions $f_t(x), \;\; t = 1, \cdots, 4$
- The decision function $f_t(x)$ of task t takes the form

$$f_t(x) = f_{t,1}(x) + f_{t,2}(x) + f_{t,3}(x) + b_t \quad \text{with} \quad f_{t,m} \in \mathcal{H}_m$$

- Regularizer

$$\Omega(f_1, \cdots, f_T) = \frac{1}{2} \sum_{m=1}^{3} \left(\sum_{t=1}^{4} \|f_{t,m}\|^2_{\mathcal{H}_m} \right)^{1/2}$$

- The term $\|f_{\cdot,m}\| = \left(\sum_{t=1}^{4} \|f_{t,m}\|^2_{\mathcal{H}_m} \right)^{1/2}$ measures the importance of kernel k_m across the tasks. If the kernel does not influence the decision functions f_t, we want the term $\|f_{\cdot,m}\|$ small or null
- Equivalent to $\ell_1 - \ell_2$ penalization (mixed-norm regularization)

Multi-task Learning

General problem

- Gathering all elements

$$\min_{f_1,\cdots,f_T} \quad C \cdot \sum_{t=1}^{T} \sum_{i \in \mathcal{D}_t} L(f_t(x_i^t), y_i^t) + \frac{1}{2} \sum_{m=1}^{M} \|f_{\cdot,m}\|$$

with

$$\|f_{\cdot,m}\| = \left(\sum_{t=1}^{T} \|f_{t,m}\|_{\mathcal{H}_m}^2 \right)^{1/2}$$

- We retrieve a multiple kernel formulation over $\|f_{\cdot,m}\|$
- Any multiple kernel solver can be used to address the learning problem

Multi-task Learning

Variational approach

$$\begin{array}{ll} \min_{f_1,\cdots,f_T,\mathbf{d}} & C\sum_{t=1}^{T}\sum_{i\in\mathcal{D}} L(f_t(x_i^t),y_i^t) + \sum_{m=1}^{M}\frac{\|f_{\cdot,m}\|^2}{d_m} \\ \text{s.t} & \sum_m d_m = 1, \quad d_m \geq 0 \quad \forall m \end{array}$$

Variables d_m : extra-parameters. The values of d_m stress the importance of the corresponding kernels k_m in the solution. $d_m = 0$ means kernel k_m discarded from the solution.

Rearranging the optimization problem

- Using the fact that $\|f_{\cdot,m}\|^2 = \sum_{t=1}^{T}\|f_{t,m}\|^2_{\mathcal{H}_m}$, we obtain

$$\begin{array}{ll} \min_{\mathbf{d}} & J(\mathbf{d}) = \sum_t J_t(\mathbf{d}) \\ \text{s.t} & \sum_m d_m = 1, \quad d_m \geq 0 \quad \forall m \end{array}$$

with $\quad J_t(\mathbf{d}) = \min_{f_t} \; C\sum_{i\in\mathbf{D}_t} L(f_t(x_i^t),y_i^t) + \sum_m \frac{\|f_{t,m}\|^2}{d_m}$

Multi-task Learning

Solver : two-level optimization

1. Fix $\mathbf{d}$ and solve for each task

$$\min_{f_t} \; C \sum_{i \in \mathbf{D}_t} L(f_t(x_i^t), y_i^t) + \sum_m \frac{\|f_{t,m}\|^2}{d_m}$$

- Recall that $f_t(x) = \sum_m f_{t,m}(x) + b_t$
- Each function f_t is retrieved from the solution of an SVM defined over kernel $K(x,z) = \sum_m d_m k_m(x,z)$

$$\begin{array}{ll} \max\limits_{\alpha_i^t} & -\frac{1}{2}\sum_{i,j} \alpha_i^t \alpha_j^t y_i^t y_j^t \sum_m d_m K_m(x_i^t, x_j^t) + \sum_i \alpha_i^t \\ \text{s.t} & \sum_i \alpha_i^t y_i^t = 0, \quad \text{and} \quad 0 \le \alpha_i^t \le C \quad \forall i \end{array}$$

2. Knowing the solutions f_t, we can update the parameters d_m by a projected gradient algorithm. The gradient is simply $\sum_t \nabla_{\mathbf{d}} J_t(\mathbf{d})$

Application on BCI

- 896 variables $\Longrightarrow$ 896 kernels
- Tasks : 4 acquisition sessions
- 1/3 data used for training

Algorithms	AUC	# variables
$\text{MTL}_{\ell_1-\ell_2}$	85.72 ± 1.8	192 ± 11
FullMKL	86.17 ± 1.8	214 ± 12
SepMKL	84.15 ± 1.8	272 ± 13

High AUC means good algorithm

FullMKL : multiple kernel SVM trained on the entire available training set

SepMKL : tasks are trained separately (state of art)

Handling non-convex joint sparsity regularizer

Non-convex regularization

- Instead of the $\ell_1 - \ell_2$ penalty, we consider a non-convex "pseudo-norm" $\ell_p - \ell_2$ penalty with $0 < p < 1$
- Aim : emphasize the sparse behavior of the solution
- Proposed regularization closely related to the spirit of non-convex group lasso algorithms that was issued from consistency results of the convex group lasso
- Non-convex regularizer : $\Omega(f_1, \cdots, f_T) = \sum_{m=1}^{M} g(\|f_{\cdot,m}\|)$ with $g(u) = u^p, \quad 0 < p < 1, \quad \text{and} \quad u \geq 0$

  Remark : any other penalty function $g(u)$ could be used as well

Handling non-convex joint sparsity regularizer

DC algorithm

- Assume the optimization problem $\min_\theta J(\theta)$ where J is a non-convex objective function
- Decompose $J(\theta)$ as a difference of two convex functions : $J(\theta) = J_1(\theta) - J_2(\theta)$
- Solve iteratively $\theta^{(i+1)} = \mathrm{argmin}_\theta\, J_1(\theta) - \langle \nabla_\theta J_2(\theta^{(i)}), \theta - \theta^{(i)} \rangle$ until convergence

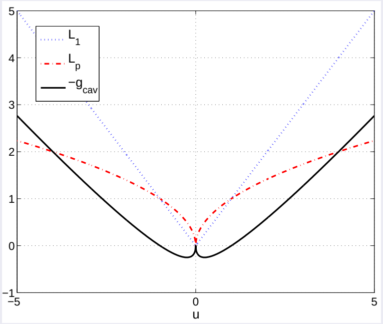
Handling non-convex joint sparsity regularizer

Application of DC principle

- The non-convex part is $\Omega(f_1, \cdots, f_T) = \sum_{m=1}^{M} g(\|f_{\cdot,m}\|)$ with

$$g(u) = u^p, \quad 0 < p < 1$$

- Decomposition : $g(u) = u - (u - u^p)$



Handling non-convex joint sparsity regularizer

Application of DC principle

- $$\min_{f_1,\cdots,f_T} \quad C \cdot \sum_{t=1}^{T} \sum_{i \in \mathcal{D}_t} L(f_t(x_i^t), y_i^t) + \frac{1}{2} \sum_{m=1}^{M} g\left(\|f_{\cdot,m}\|\right)$$
- Decomposition : $g(u) = u - (u - u^p)$
- It leads to

$$J_1 = C \sum_t \sum_{i \in \mathcal{D}_t} L(f_t(x_i^t), y_i^t) + \sum_m \|f_{\cdot,m}\|$$

$$J_2 = \sum_m (-\|f_{\cdot,m}\| + \|f_{\cdot,m}\|^p)$$

Handling non-convex joint sparsity regularizer

Application of DC principle

- Applying the DC algorithm, the non-convex MTL optimization problem boils down to solve iteratively a reweigthed $\ell_1 - \ell_2$ multi-task problem

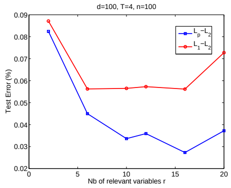
$$\begin{array}{ll} \min_{f_1,\cdots,f_T,\mathbf{d}} & C\sum_t \sum_{i\in\mathcal{D}_t} L(f_t(x_i^t), y_i^t) + \sum_m \beta_m^2 \frac{\|f_{\cdot,m}\|^2}{d_m} \\ \text{s.t} & \sum_m d_m = 1, \quad d_m \geq 0 \quad \forall m \end{array}$$

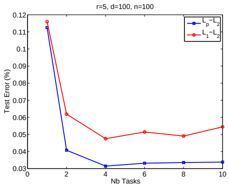
- At each iteration, the weights are given by $\beta_m = \frac{p}{\|f_{\cdot,m}^{(i)}\|^{1-p}}$
- The optimization problem of each iteration can be cast into the convex MTL using kernel $k'_m(x,z) = \frac{k_m(x,z)}{\beta_m^2}$
- The iterative scheme is applied until convergence of the weigths β_m

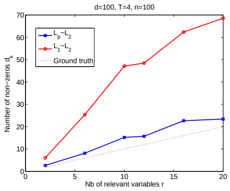
Application on a simple example

Toy problem

- T binary classification tasks with n samples $x \in \mathbb{R}^d$ for each task
- The classes follow gaussian distributions with means μ, $-\mu$ and random covariance matrix in $\mathbb{R}^r$ where r is the number of relevant variables. The remaining $d - r$ variables are generated randomly and are considered as spurious variables
- Kernels : each dimension defined a kernel $k_m, \quad m = 1, \cdots, d$







Application on BCI data

- 896 kernels
- 4 Tasks

Algorithms	AUC	# variables
$\text{MTL}_{\ell_1-\ell_2}$	85.72 ± 1.8	192 ± 11
FullMKL	86.17 ± 1.8	214 ± 12
SepMKL	84.15 ± 1.8	272 ± 13
$\text{MTL}_{\ell_p-\ell_2}$	86.37 ± 1.3	43 ± 6

For $\text{MTL}_{\ell_p-\ell_2}$, $p = 0.5$

FullMKL : multiple kernel SVM trained on entire training set

SepMKL : tasks are trained separately

- The recognition performances for $\text{MTL}_{\ell_p-\ell_2}$ are slightly improved with however an important reduction of the number of variables $\Longrightarrow$ few channels really needed

Conclusion

- MTL-MKL learning algorithm
- Yields sligth improvments on BCI data for one subject inter-session variability
- Extend its application to tackle inter-subject variabilities
- Test on other learning problems
- Speed-up the learning algorithm by considering an online version ?