Regularization with Multiple Kernels

Francesca Odone and **Lorenzo Rosasco**

RegML 2014

About this class

Goal To introduce and motivate regularization with multiple kernels and take a peak at the field of structured sparsity regularization.

Plan

- Introduction
- Sum of reproducing kernels.
- Solving mkl.
- Applications.

Multiple Kernel Learning

Let $k_1, \ldots, k_p$ a sequence of reproducing kernels and $(\mathcal{H}_1, \|\cdot\|_1), \ldots, (\mathcal{H}_p, \|\cdot\|_p)$ the corresponding RKHSs.

Multiple Kernel Learning (MKL)

Consider the following minimization problem

$$\min_{f=\sum_{j=1}^p f_j,\ \ f_1 \in \mathcal{H}_1, \ldots, f_p \in \mathcal{H}_p} \left\{ \frac{1}{n} \sum_{i=1}^n (f(x_i) - y_i)^2 + 2\lambda \sum_{j=1}^p \|f_j\|_j \right\},$$

Why Multiple Kernels?

We will see in the following that MKL has several applications:

Applications

1. To augment approximation power.
2. As an alternative to model selection.
3. To perform non-linear feature selection.
4. To perform data fusion.

Applications

Augment approximation power

Rather than taking a single kernel we can take a combination of a large number of kernels.

AUGMENT APPROXIMATION POWER

Rather than taking a single kernel we can take a combination of a large number of kernels.

MODEL SELECTION

Many kernels require choosing at least one parameter. Using MKL we can choose the solution as a combination of the different kernels obtained from different regularization parameter values.

For example, if $K_\sigma(x, x') = e^{\frac{-\|x-x'\|^2}{2\sigma^2}}$, we can take $\sigma_1, \ldots, \sigma_p$ and set

$$k_1 = K_{\sigma_1}, \ldots, k_p = K_{\sigma_p}.$$

Applications (cont.)

Non Linear Feature selection

Take $k_j(x,t) = k_j(x^j, t^j)$ so that $f(x) = \sum_{j=1}^{p} f_j(x^j)$.
By using sparse MKL we can select a subset of feature that (individually) depend non linearly to the output.

Applications (cont.)

Non Linear Feature selection

Take $k_j(x,t) = k_j(x^j, t^j)$ so that $f(x) = \sum_{j=1}^{p} f_j(x^j)$.
By using sparse MKL we can select a subset of feature that (individually) depend non linearly to the output.

Data Fusion

We can consider different kernels $k_1, \ldots, k_p$ capturing different features of the data.
In the case of images we can take kernels based colors, texture etc. and combine them to obtain a better model.

Preliminaries: Sum of RKHSs

Let k_1, k_2 be reproducing kernels then $k = k_1 + k_2$ is also a reproducing kernel, and we can consider its RKHS $\mathcal{H}_k$ with inner product $\langle \cdot, \cdot \rangle_k$ and norm $\| \cdot \|_k$.

Can we describe $\mathcal{H}_K$ in terms of the composing RKHSs?

Preliminaries: Sum of RKHSs

Let k_1, k_2 be reproducing kernels then $k = k_1 + k_2$ is also a reproducing kernel, and we can consider its RKHS $\mathcal{H}_k$ with inner product $\langle \cdot, \cdot \rangle_k$ and norm $\|\cdot\|_k$.

Can we describe $\mathcal{H}_K$ in terms of the composing RKHSs?

- If $\mathcal{H}_1 \cap \mathcal{H}_2 = \emptyset$ then

$$\|f\|_k^2 = \|f_1\|_1^2 + \|f_2\|_2^2,$$

  and $\mathcal{H}_k = \mathcal{H}_1 \oplus \mathcal{H}_2$.
- If $\mathcal{H}_1 \cap \mathcal{H}_2 \neq \emptyset$, the norm of $f \in \mathcal{H}_k$ is given by

$$\|f\|_k^2 = \min\left\{ \|f_1\|_1^2 + \|f_2\|_2^2 \right\},$$

  where $f_1 \in \mathcal{H}_{k_1}, f_2 \in \mathcal{H}_{k_2}$ such that $f = f_1 + f_2$.

Sum of RKHSs (cont.)

The RKHS can be endowed with other norms.

- If $\mathcal{H}_1 \cap \mathcal{H}_1 = \emptyset$ we can consider $\|f\| = \|f_1\|_1 + \|f_2\|_2$.

Sum of RKHSs (cont.)

The RKHS can be endowed with other norms.

- If $\mathcal{H}_1 \cap \mathcal{H}_1 = \emptyset$ we can consider $\|f\| = \|f_1\|_1 + \|f_2\|_2$.
- If $\mathcal{H}_1 \cap \mathcal{H}_2 \neq \emptyset$, we can consider

$$\|f\| = \min\left\{ \|f_1\|_1 + \|f_2\|_2 \right\}$$

where $f_1 \in \mathcal{H}_{k_1}, f_2 \in \mathcal{H}_{k_2}$ such that $f = f_1 + f_2$.

Sum of RKHSs (cont.)

The RKHS can be endowed with other norms.

- If $\mathcal{H}_1 \cap \mathcal{H}_1 = \emptyset$ we can consider $\|f\| = \|f_1\|_1 + \|f_2\|_2$.
- If $\mathcal{H}_1 \cap \mathcal{H}_2 \neq \emptyset$, we can consider

$$\|f\| = \min\left\{ \|f_1\|_1 + \|f_2\|_2 \right\}$$

where $f_1 \in \mathcal{H}_{k_1}, f_2 \in \mathcal{H}_{k_2}$ such that $f = f_1 + f_2$.

Note that the above norms are **not** induced by the inner product in $\mathcal{H}_k$.

How should we Regularize with Multiple Kernels?

Based on the previous norms, we can consider two different algorithms:

Tikhonov MKL

$$\min_{f=\sum_{j=1}^{p} f_j,\ \ f_1\in\mathcal{H}_1,\ldots,f_p\in\mathcal{H}_p} \left\{ \frac{1}{n}\sum_{i=1}^{n}(f(x_i)-y_i)^2 + \lambda \sum_{j=1}^{p} \left\|f_j\right\|_j^{\,2} \right\},$$

How should we Regularize with Multiple Kernels?

Based on the previous norms, we can consider two different algorithms:

Tikhonov MKL

$$\min_{f=\sum_{j=1}^{p} f_j,\ \ f_1\in\mathcal{H}_1,\ldots,f_p\in\mathcal{H}_p} \left\{ \frac{1}{n}\sum_{i=1}^{n}(f(x_i)-y_i)^2 + \lambda \sum_{j=1}^{p} \|f_j\|_j^{\,2} \right\},$$

Sparse MKL

$$\min_{f=\sum_{j=1}^{p} f_j,\ \ f_1\in\mathcal{H}_1,\ldots,f_p\in\mathcal{H}_p} \left\{ \frac{1}{n}\sum_{i=1}^{n}(f(x_i)-y_i)^2 + 2\lambda \sum_{j=1}^{p} \|f_j\|_j \right\},$$

The difference between the two regularizers is clear in a simple case:
Take $k_i(x,t) = x^i t^i$, then:

- $f(x) = \sum_{j=1}^{p} f_j(x) = \sum_{j=1}^{p} w^j x^j = \langle w, x \rangle$

The difference between the two regularizers is clear in a simple case:
Take $k_i(x,t) = x^i t^i$, then:

- $f(x) = \sum_{j=1}^{p} f_j(x) = \sum_{j=1}^{p} w^j x^j = \langle w, x \rangle$
- $\sum_{j=1}^{p} \|f_j\|_j^2 = \sum_{j=1}^{p} |w^j|^2 = \|w\|^2$

The difference between the two regularizers is clear in a simple case:
Take $k_i(x,t) = x^i t^i$, then:

- $f(x) = \sum_{j=1}^{p} f_j(x) = \sum_{j=1}^{p} w^j x^j = \langle w, x \rangle$
- $\sum_{j=1}^{p} \|f_j\|_j^2 = \sum_{j=1}^{p} |w^j|^2 = \|w\|^2$
- $\sum_{j=1}^{p} \|f_j\|_j = \sum_{j=1}^{p} |w^j|$

Sparsity Inducing Regularization

In general one can see that the regularizer

$$R(f) = \sum_{j=1}^{p} \left\| f_j \right\|_j$$

forces the norm of some functions to be zero.

Some of the kernels will play no role in the solution!

A KEY OBSERVATION

Let $k = \sum_{j=1}^{p} \beta_j k_j$, with $\beta_j >$ for all $j = 1, \ldots, p$. Then

$$\|f\|_k^2 = \min\left\{ \sum_{j=1}^{p} \frac{\|f_j\|_j^2}{\beta_j} \right\}$$

where $f_j \in \mathcal{H}_{k_j}$,for $j = 1, \ldots, p$ and $f = \sum_{j=1}^{p} f_j$.

A KEY OBSERVATION (CONT.)

Consider the functional

$$\min_{k \in \mathcal{K}} \|f\|_k$$

where

$$\mathcal{K} = \{k \ : \ k = \sum_{j=1}^{p} \beta_j k_j, \ \ \beta_j \geq 0, \ j = 1, \ldots, p, \ \ \sum_{j=1}^{p} \beta_j = 1\}.$$

A KEY OBSERVATION (CONT.)

Consider the functional

$$\min_{k \in \mathcal{K}} \|f\|_k$$

where

$$\mathcal{K} = \{k \ : \ k = \sum_{j=1}^{p} \beta_j k_j, \ \ \beta_j \geq 0, \ j = 1, \ldots, p, \ \ \sum_{j=1}^{p} \beta_j = 1\}.$$

It is possible to prove that:

$$\min_{k \in \mathcal{K}} \|f\|_k = \min \left\{ \sum_{j=1}^{p} \|f_j\|_j \right\},$$

where $f_j \in \mathcal{H}_{k_j}$,for $j = 1, \ldots, p$ and $f = \sum_{j=1}^{p} f_j$.

Learning the Kernel Function

Using the previous observation one can prove that the problem

$$\min_{f=\sum_{j=1}^{p} f_j,\ \ f_1\in\mathcal{H}_1,\ldots,f_p\in\mathcal{H}_p} \left\{ \frac{1}{n}\sum_{i=1}^{n}(f(x_i)-y_i)^2 + 2\lambda\Big(\sum_{j=1}^{p}\|f_j\|_j\Big)^2 \right\},$$

is equivalent to the double minimization

$$\min_{k\in\mathcal{K}}\min_{f\in\mathcal{H}_k} \left\{ \frac{1}{n}\sum_{i=1}^{n}(f(x_i)-y_i)^2 + 2\lambda\sum_{j=1}^{p}\|f\|_k^2 \right\}.$$

Note that we took the square of the penalty in the first problem, this can be shown to be equivalent to changing the regularization parameter, (see, e.g., Borwein and Lewis, 2000, Section 3.2).

Learning the Kernel Function (cont.)

Considering

$$\min_{k \in \mathcal{K}} \min_{f \in \mathcal{H}_k} \left\{ \frac{1}{n} \sum_{i=1}^{n} (f(x_i) - y_i)^2 + 2\lambda \sum_{j=1}^{p} \|f\|^2 \right\},$$

we have a new interpretation of the algorithm.

We are *learning* a new kernel given by a (convex) combination of basis kernels.

Algorithms for MKL

There are many algorithms to solve MKL:

- Block Coordinate
- Active Sets Methods
- Greedy (approximate) methods
- ...

We are going to describe an optimization procedure using a proximal method.

One can see that the solution of the problem

$$\min_{f=\sum_{j=1}^{p} f_j,\ \ f_1\in\mathcal{H}_1,\ldots,f_p\in\mathcal{H}_p} \left\{ \frac{1}{n}\sum_{i=1}^{n}(f(x_i)-y_i)^2 + 2\lambda\sum_{j=1}^{p}\left\|f_j\right\|_j \right\},$$

is of the form

$$f^* = \sum_{j=1}^{p} f_j^*,$$

where

$$f_j^*(x) = \sum_{i=1}^{n} \alpha_i^j k_i(x_i, x).$$

We can reduce the minimization to a finite dimensional problem

Some Notation

We need some notation:

$$c = (c^1, \ldots, c^p)^T \text{ with } c^j = (c_1^j, \ldots, c_n^j)^T,$$

$$\mathbf{K} = \left.\begin{pmatrix} \mathbf{K}_1 & \ldots & \mathbf{K}_p \\ \vdots & \ddots & \vdots \\ \mathbf{K}_1 & \ldots & \mathbf{K}_p \end{pmatrix}\right\} p \text{ times} \qquad \text{with } [\mathbf{K}_j]_{ii'} = k_j(x_i, x_{i'}),$$

$$\mathbf{y} = (\underbrace{y^T, \ldots, y^T}_{p \text{ times}})^T$$

$$\mathbf{k}(x) = (\mathbf{k}_1(x), \ldots, \mathbf{k}_p(x))^T$$

with

$$\mathbf{k}_j(x) = \big(k_j(x_1, x), \ldots, k_j(x_n, x)\big)$$

Iterative Soft Thresholding

We can write the solution as $f^*(x) = c_1^T \mathbf{k}_1(x) + \cdots + c_p^T \mathbf{k}_p(x))$ where the coefficients are given by the following iteration

set $c^0 = 0$
for t$= 1, \ldots, t_{\max}$

$$c^t = \text{Prox}_{\lambda/\eta}\left(c^{t-1} - \frac{1}{\eta n}(\mathbf{K}c^{t-1} - \mathbf{y})\right)$$

The map $\text{Prox}_{\lambda/\eta}$ is the so called proximal operator.

Proximal Operator and Soft Thresholding

The proximal operator can be computed in a simple closed form.

The proximal operator can be computed in a simple closed form.
In fact

$$\text{Prox}_{\lambda/\eta}\left(c^{t-1} - \frac{1}{\eta n}(\mathbf{K}c^{t-1} - \mathbf{y})\right) = \hat{\mathbf{S}}_{\lambda/\eta}\left(\mathbf{K}, c^{t-1} - \frac{1}{\eta n}(\mathbf{K}c^{t-1} - \mathbf{y})\right)$$

where the soft-thresholding operator $\hat{\mathbf{S}}_\tau(\mathbf{K}, c)$ acts component-wise as

$$\hat{\mathbf{S}}_\tau(\mathbf{K}, c)_j = \frac{c_j^T}{\sqrt{c_j^T \mathbf{K}_j c_j}}(\sqrt{c_j^T \mathbf{K}_j c_j} - \tau)_+.$$

Remarks on Computations

- **[Step Size].** η is a step-size that can be chosen a priori or adaptively.

Remarks on Computations

- [**Step Size**]. η is a step-size that can be chosen a priori or adaptively.
- [**Regularization Path**]. When computing the solution for several regularization parameter values it helps to use a continuation strategy:
  1. take a grid of values for τ, e.g. $\tau_1 < \cdots < \tau_q$.
  2. Compute the solution c^q corresponding to the larger value.
  3. Use this solution to initialize the algorithm for the next value τ_{q-1}.

Applications of MKL

Applications

1. To augment approximation power.
2. As an alternative to model selection.
3. To perform non-linear feature selection.
4. To perform data fusion.

Final Remarks

Multiple kernel learning: Sparse vs Tikhonov regularization.

- Sparse regularization gives more interpretable models, it seems preferable when the number of base kernels is large, is computationally demanding.
- Tikhonov regularization seems to work well when the basis kernels are few and well designed. It is computationally efficient.

Final Remarks

Multiple kernel learning: Sparse vs Tikhonov regularization.

- Sparse regularization gives more interpretable models, it seems preferable when the number of base kernels is large, is computationally demanding.
- Tikhonov regularization seems to work well when the basis kernels are few and well designed. It is computationally efficient.

Related Topics:

- Learning with structured kernels, e.g. hierarchical kernels
- Structured sparsity regularization, group lasso etc.