Second order optimization of kernel parameters

Olivier Chapelle & Alain Rakotomamonjy
Presented by Francis Bach

Dec 12th, 2008
Nips Workshop on *Automatic Selection of Optimal Kernels*

Introduction

Multiple Kernel Learning (MKL)

Given M kernel functions $K_1, \ldots, K_M$ that are potentially well suited for a given problem, find a positive linear combination of these kernels such that the resutling kernel K is "optimal" in some sense,

$$K(\mathbf{x}, \mathbf{x}') = \sum_{m=1}^{M} d_m K^m(\mathbf{x}, \mathbf{x}'), \text{ with } d_m \geq 0, \sum_m d_m = 1.$$

Need to learn together the kernel coefficients d_m and the SVM parameters.

Previous work

- [Lanckriet et al., 04]: Semi-definite programming
- [Bach et al., 04]: SMO
- [Sonnenburg et al., 06]: Semi-infinite linear programming
- [Rakotomamonjy et al., 08]: Gradient descent, *simpleMKL*
  [Chapelle et al., 02]: Gradient descent for general kernel

All solve the same problem, but use different optimization techniques. SimpleMKL has been shown to be more efficient.

We propose a Newton type optimization technique for MKL which turns out to be even more efficient than simpleMKL.

Previous work

- [Lanckriet et al., 04]: Semi-definite programming
- [Bach et al., 04]: SMO
- [Sonnenburg et al., 06]: Semi-infinite linear programming
- [Rakotomamonjy et al., 08]: Gradient descent, *simpleMKL*
  [Chapelle et al., 02]: Gradient descent for general kernel

All solve the same problem, but use different optimization techniques. SimpleMKL has been shown to be more efficient.

We propose a Newton type optimization technique for MKL which turns out to be even more efficient than simpleMKL.

Objective function

- Consider a hard margin SVM with a kernel K. The following objective function is maximized:

$$\Omega(K) \quad := \quad \max_{\alpha_i} \sum_{i=1}^{n} \alpha_i y_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j K(\mathbf{x}_i, \mathbf{x}_j)$$

$$\text{under constraint} \quad 0 \leq \alpha_i y_i \leq C \quad \text{and} \quad \sum_{i=1}^{n} \alpha_i = 0.$$

- Since finding the maximum margin solution seems to give good empirical results, it has been proposed to extend this idea for MKL: find the kernel that maximizes the margin or equivalently

$$\min_{d_m \geq 0} \quad \Omega\left(\sum_{m=1}^{M} d_m K^m\right)$$

Problem

- The SVM objective function has been derived for finding an hyperplane for a given kernel, not for learning the kernel matrix.
- Illustration of the problem: since $\Omega(dK) = \Omega(K)/d$, Ω can be trivially minimized.
- This is usually fixed by adding the constraint $\sum d_m \leq 1$. But is the L_1 norm on $\mathbf{d}$ the most appropriate?

Hyperparameter view

- A more principle approach is to consider the d_m as *hyperparameters* and tune them on a model selection criterion.
- A convenient criterion is a bound on the generalization error [Bousquet, Herrmann, 03], $T(K)\Omega(K)$, where $T(K)$ is the re-centered trace, $T(K) = \sum_i K(\mathbf{x}_i, \mathbf{x}_i) - \frac{1}{n}\sum_{i,j} K(\mathbf{x}_i, \mathbf{x}_j)$.
- Because $\Omega(dK) = \Omega(K)/d$, this is equivalent to minimize $\Omega(K)$ under constraint $T(K) =$ constant, or

$$\min_{d_m} \quad \Omega\left(\sum d_m K^m\right),$$

$$\text{under constraint} \quad \sum d_m T(K^m) = 1 \ \text{ and } \ d_m \geq 0.$$

$\longrightarrow$ The linear constraint on d_m appears naturally.
$\longrightarrow$ Identical to the "standard" view if the K_i are *centered* and *normalized*.

Optimization

No need for complex optimization techniques.
Simply define:

$$J(\mathbf{d}) := \Omega\left(\sum d_m K^m\right)$$

and perform a gradient based optimization of J which is twice differentiable almost everywhere.

For a given $\mathbf{d}$, let $\alpha^\star$ be the SVM solution.

$$g_m := \frac{\partial J}{\partial d_m} = -\frac{1}{2}\sum_{i,j}\alpha_i^\star\alpha_j^\star K^m(\mathbf{x}_i, \mathbf{x}_j).$$

Second order

We consider a hard margin SVM. L_2 penalization of the slacks can be implemented by adding the identity in the set of base kernels (resulting in automatic tuning of C). L_1 penalization is slightly more complex: see our extended abstract.

To compute the Hessian of J, we first need to compute [Chapelle et al., 02]:

$$\frac{\partial \alpha^\star_{\mathsf{sv}}}{\partial d_m} = -K^{-1}_{\mathsf{sv},\mathsf{sv}} K^m_{\mathsf{sv},\mathsf{sv}} \alpha^\star_{\mathsf{sv}},$$

where sv is the set of support vectors.
The Hessian is then:

$$H = Q^\top K^{-1}_{\mathsf{sv},\mathsf{sv}} Q \succeq 0 \quad \text{with } Q := [\cdots K^m_{\mathsf{sv},\mathsf{sv}} \alpha^\star_{\mathsf{sv}} \cdots]_{1 \le m \le M}.$$

Search direction

The step direction s is a constrained Newton step found by minimizing the quadratic problem:

$$\min \quad \frac{1}{2}\mathbf{s}^\top H\mathbf{s} + \mathbf{s}^\top \mathbf{g},$$

$$\text{under constraints} \quad \sum s_m T(K^m) = 0 \;\; \text{and} \;\; \mathbf{s} + \mathbf{d} \geq 0.$$

The quadratic form corresponds to the second order expansion of J.

The constraints ensure that any solution on the segment $[\mathbf{d}, \mathbf{d} + \mathbf{s}]$ satisfies the original constraints.

Finally backtracking is performed in case $J(\mathbf{d} + \mathbf{s}) \geq J(\mathbf{d})$.

Complexity

For each iteration:

- SVM training: $O(nn_{\mathsf{sv}} + n_{\mathsf{sv}}^3)$.
- Inverting $K_{\mathsf{sv},\mathsf{sv}}$ is $O(n_{\mathsf{sv}}^3)$, but might already be available as a by-product of the SVM training.
- Computing H: $O(Mn_{\mathsf{sv}}(M + n_{\mathsf{sv}}))$
- Finding s: $O(M^3)$.

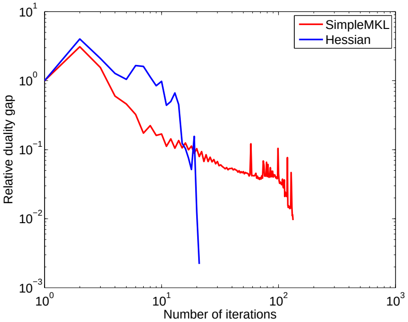
The number of iterations is usually less than 10.

$\longrightarrow$ When $M < n_{\mathsf{sv}}$, computing s is not more expensive than the SVM training.

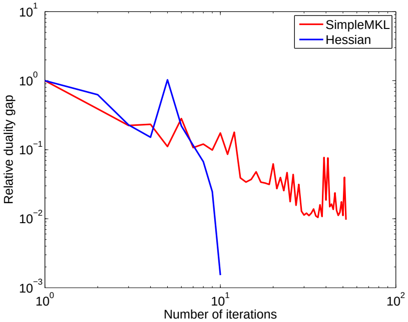
Experiments

Comparison with simpleMKL on several UCI datasets as in [Rakotomamonjy et al., 08]
Kernels are centered and normalized.

Relative duality gap as a function of the number of iterations:

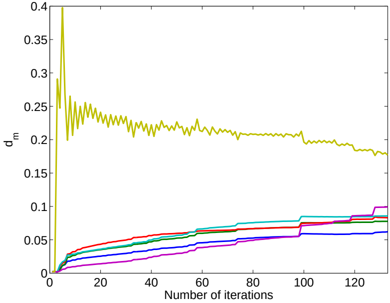


Ionosphere
$n = 246, M = 442$

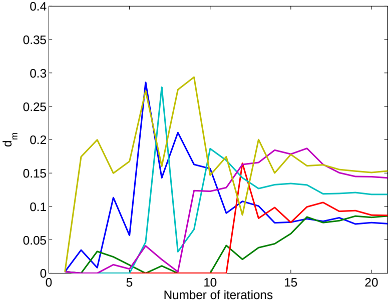


Liver
$n = 241, M = 91$

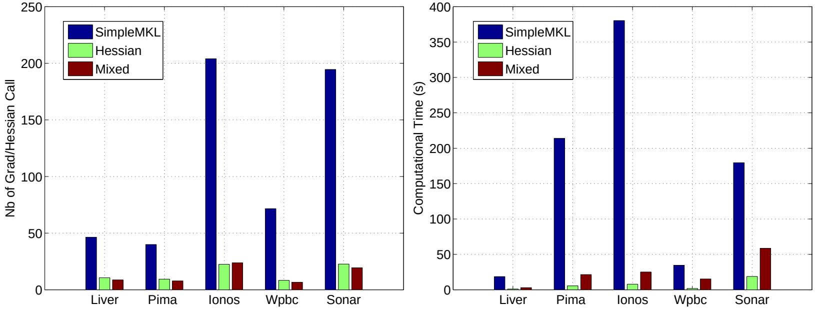
Example of convergence behavior of the weights d_m on Ionosphere:



SimpleMKL



HessianMKL



- Stopping criterion: duality gap ≤ 0.01.
- Mixed strategy: one initial gradient step followed by Newton type optimization.
- ≈ 1 SVM call per iteration for HessianSVM (>1 if backtracking necessary) but much more for simpleMKL (because of line search).

Conclusion

- Simple optimization strategy for MKL: requires just standard SVM training and small QP (whose size is the number of kernels).
- Very fast method because:
  1. The number of SVM trainings is small (of the order of 10)
  2. The extra cost required for computing the Newton type direction is not prohibitive.
- As an aside, MKL should be considered as a model selection problem. From this point of view, need for centering and normalizing the kernel matrices.