A Genetic Algorithm Based Multiple Kernel Learning Algorithm

Wang Fan

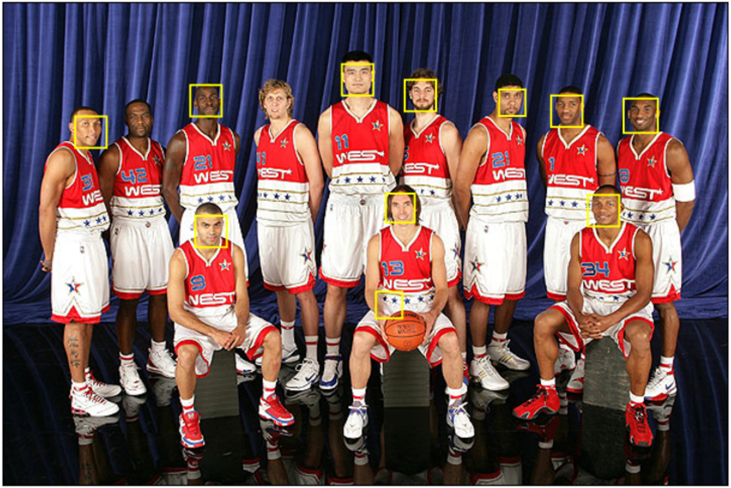
Supervisor: Assoc. Prof. Chua Chek Beng

Division of Mathematical Sciences
School of Physical and Mathematical Sciences
Nanyang Technological University
Singapore

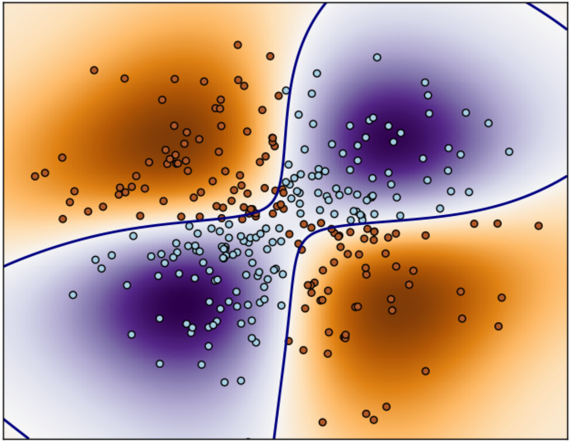
20 April 2016

Introduction

Introduction



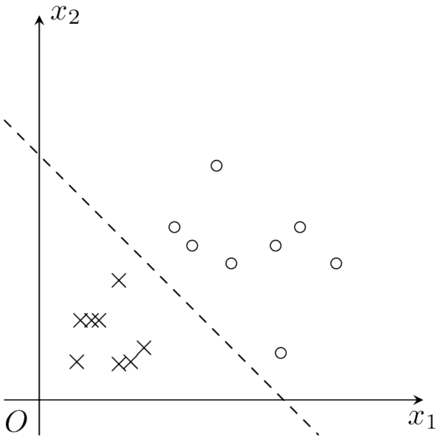
Introduction



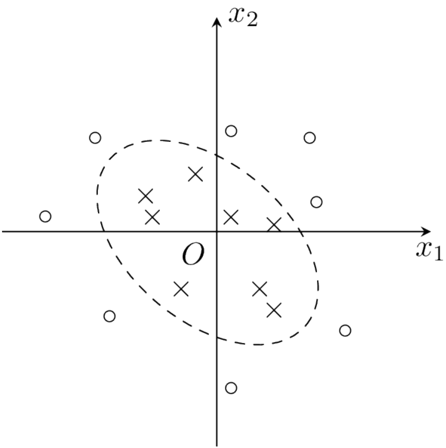
Outline

Outline

Motivation



Motivation



Motivation

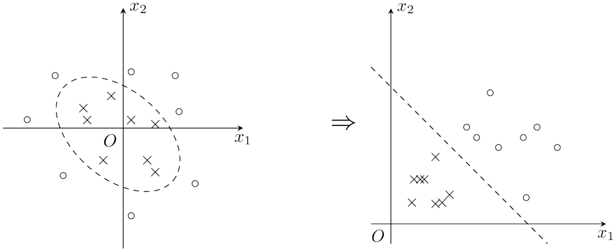
- Kernel method
  - mapping data to a feature space $\Longrightarrow$ data are linear separable

Kernel

A kernel is a function K that for all $\mathbf{x}, \mathbf{x}' \in \mathbb{R}^n$ satisfies

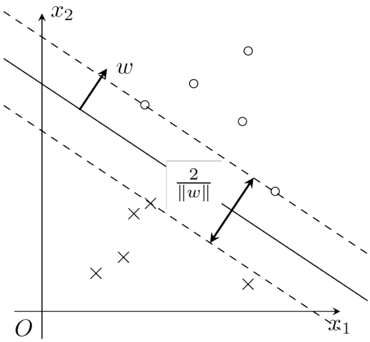
$$K(\mathbf{x}, \mathbf{x}') = \Phi(\mathbf{x}) \cdot \Phi(\mathbf{x}'),$$

where Φ is a mapping from $\mathbb{R}^n$ to Hilbert space $\mathcal{H}$, i.e. $\Phi : \mathbf{x} \mapsto \Phi(\mathbf{x}) \in \mathcal{H}$.



Motivation

- Kernel method
  - mapping data to a feature space $\Longrightarrow$ data are linear separable
- Support Vector Machine looks for a hyperplane that maximizes the distance between two classes.



Motivation

- Kernel method
  - mapping data to a feature space $\Longrightarrow$ data are linear separable
- Support Vector Machine
  - Standard SVM:

$$\begin{aligned} \min_{\mathbf{w},\xi,b} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\xi_i \\ \text{s.t.} \quad & y_i((\mathbf{w}\cdot\Phi(x_i)) - b) \geq 1 - \xi_i, i = 1, 2, \ldots, n \\ & \xi_i \geq 0, i = 1, 2, \ldots, n, \end{aligned}$$

  - Dual form:

$$\begin{aligned} \max_{\alpha} \quad & -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} y_i y_j \Big(\Phi(x_i)\cdot\Phi(x_j)\Big)\alpha_i\alpha_j + \sum_{j=1}^{n}\alpha_j \\ \text{s.t.} \quad & \sum_{i=1}^{n}\alpha_i y_i = 0, \\ & 0 \leq \alpha_i \leq C,\ i = 1, \ldots, n \end{aligned}$$

Motivation

- Kernel method
  - mapping data to a feature space $\Longrightarrow$ data are linear separable
- Problem of kernel methods, in particular, SVM:
  - How to choose a ‘good’ kernel?

Motivation

- Kernel method
  - mapping data to a feature space $\Longrightarrow$ data are linear separable
- Problem of kernel methods, in particular, SVM:
  - How to choose a 'good' kernel?
- Multiple kernel learning is one solution.

Motivation

- Multiple kernel learning
  - MKL learns a linear combination of predefined kernels

$$K(\mathbf{x}, \mathbf{x}') = \sum_{k=1}^{M} \mu_k K_k(\mathbf{x}, \mathbf{x}')$$

Motivation

- Multiple kernel learning
  - MKL learns a linear combination of predefined kernels
  - Primal form: Go Back

$$
\begin{aligned}
\min_{\mathbf{w},b,\xi,\boldsymbol{\mu}} \quad & \frac{1}{2}\sum_{k=1}^{M}\frac{\|\mathbf{w}_k\|^2}{\mu_k} + C\sum_{i=1}^{n}\xi_i \\
\text{s.t.} \quad & y_i\Big(\sum_{k=1}^{M}(\mathbf{w}_k \cdot \Phi_k(x_i)) + b\Big) \geq 1 - \xi_i,\ \forall i = 1,\ldots,n \\
& \xi_i \geq 0,\ \forall i = 1,\ldots,n \\
& \sum_{k=1}^{M}\mu_k = 1,\ \mu_k \geq 0,\ \forall k = 1,\ldots,M.
\end{aligned}
$$

Motivation

- Multiple kernel learning
  - MKL learns a linear combination of predefined kernels
  - Dual form:

$$
\begin{aligned}
\max_{\alpha,\lambda} \quad & \sum_{i=1}^{n} \alpha_i - \lambda \\
\text{s.t.} \quad & \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j K_k(x_i, x_j) \leq \lambda, \ \forall k = 1, \ldots, M. \\
& \sum_{i=1}^{n} \alpha_i y_i = 0, \\
& 0 \leq \alpha_i \leq C, \ \forall i = 1, \ldots, n,
\end{aligned}
$$

  - Quadratically Constrained Linear Programming

Motivation

- Multiple kernel learning
  - MKL learns a linear combination of predefined kernels
- Problems of multiple kernel learning:
  - What predefined kernels to choose?
  - Can we tune kernel parameters automatically?

Motivation

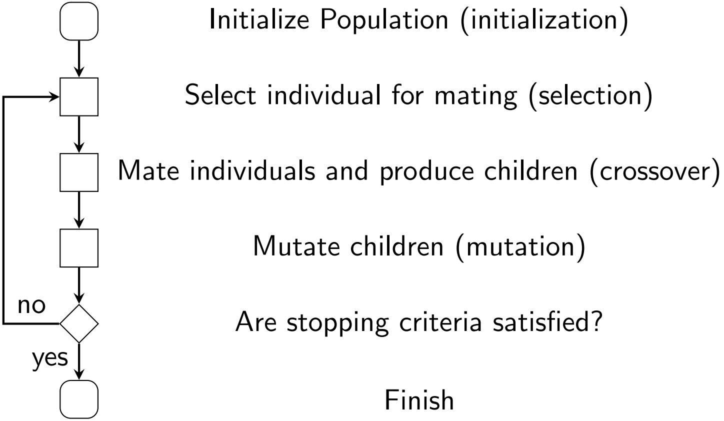
- Multiple kernel learning
  - MKL learns a linear combination of predefined kernels
- Problems of multiple kernel learning:
  - What predefined kernels to choose?
  - Can we tune kernel parameters automatically?
- Genetic Algorithm

Genetic Algorithm

- Inspired by Darwin's theory about evolution

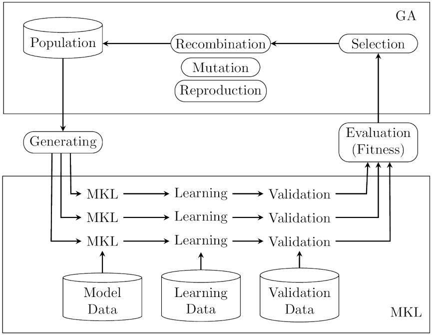
Genetic Algorithm

- Inspired by Darwin's theory about evolution

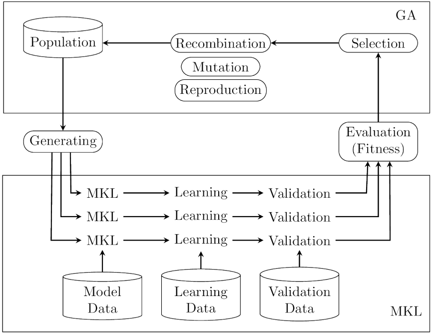


Outline

Overview

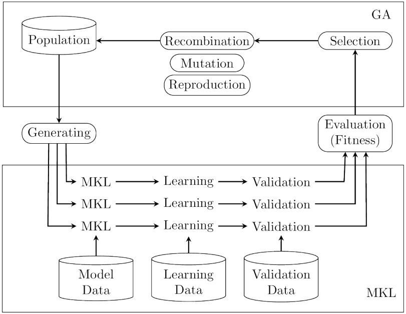

GA
Population
Recombination
Selection
Mutation
Reproduction
Generating
Evaluation
(Fitness)
MKL
Learning
Validation
MKL
Learning
Validation
MKL
Learning
Validation
Model
Data
Learning
Data
Validation
Data
MKL

Overview

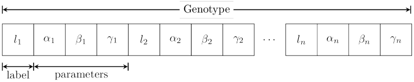


- Problems remaining
  - Determine fitness function
  - Encode parameters
  - Solve MKL problems

Methodology



- Fitness function expresses users' objective
  - cross validation accuracy



Solving MKL Problem

▸ Go to primal form

$$\min_{\boldsymbol{\mu}} J(\boldsymbol{\mu}) \qquad \sum_{k=1}^{M} \mu_k = 1, \ \mu_k \geq 0$$

where

$$J(\boldsymbol{\mu}) = \begin{cases} \min\limits_{\mathbf{w},b,\xi} & \frac{1}{2}\sum_{k=1}^{M} \frac{\|\mathbf{w}\|^2}{\mu_k} + C\sum_{i=1}^{n} \xi_i \\ \text{s.t.} & y_i \Big(\sum_{k=1}^{M} \big(\mathsf{w}_k \cdot \Phi_k(x_i)\big) + b \Big) \geq 1 - \xi_i, \ \forall i = 1, \dots, n \\ & \xi_i \geq 0, \ \forall i = 1, \dots, n. \end{cases}$$

Solving MKL Problem

Go to primal form

$$\min_{\boldsymbol{\mu}} J(\boldsymbol{\mu}) \qquad \sum_{k=1}^{M} \mu_k = 1,\ \mu_k \geq 0$$

where

$$J(\boldsymbol{\mu}) = \begin{cases} \min\limits_{\mathbf{w},b,\xi} & \dfrac{1}{2}\sum_{k=1}^{M} \dfrac{\|\mathbf{w}\|^2}{\mu_k} + C\sum_{i=1}^{n}\xi_i \\ \text{s.t.} & y_i\left(\sum_{k=1}^{M}\left(\mathrm{w}_k \cdot \Phi_k(x_i)\right) + b\right) \geq 1 - \xi_i,\ \forall i = 1, \ldots, n \\ & \xi_i \geq 0,\ \forall i = 1, \ldots, n. \end{cases}$$

- For a given $\boldsymbol{\mu}$, $J(\boldsymbol{\mu})$ is the objective value of a standard SVM problem where the kernel is $K = \sum_{k=1}^{M} \mu_k K_k$

Solving MKL Problem

Go to primal form

$$\min_{\boldsymbol{\mu}} J(\boldsymbol{\mu}) \qquad \sum_{k=1}^{M} \mu_k = 1, \;\; \mu_k \geq 0$$

where

$$J(\boldsymbol{\mu}) = \begin{cases} \min\limits_{\mathbf{w},b,\xi} & \frac{1}{2} \sum_{k=1}^{M} \frac{\|\mathbf{w}\|^2}{\mu_k} + C \sum_{i=1}^{n} \xi_i \\ \text{s.t.} & y_i \left(\sum_{k=1}^{M} \left(\mathrm{w}_k \cdot \Phi_k(x_i) \right) + b \right) \geq 1 - \xi_i, \;\; \forall i = 1, \ldots, n \\ & \xi_i \geq 0, \;\; \forall i = 1, \ldots, n. \end{cases}$$

- For a given $\boldsymbol{\mu}$, $J(\boldsymbol{\mu})$ is the objective value of a standard SVM problem where the kernel is $K = \sum_{k=1}^{M} \mu_k K_k$
- It can be optimized by conditional gradient method and gradient projection method

Condition Gradient Method

Algorithm 1 Condition gradient method for Multiple Kernel Learning

Require: $\mu_k^1 = \frac{1}{M}\ \forall k = 1, \ldots, M$

for $t = 1, 2, \ldots$ **do**

compute $J(\boldsymbol{\mu}^t)$ by standard SVM solver with kernel $K = \sum_{k=1}^{M} \mu_k K_k$

compute $\nabla J(\boldsymbol{\mu}^t)$

$\bar{\boldsymbol{\mu}}^t \leftarrow \arg\min_{\boldsymbol{\mu} \in X} \nabla J(\boldsymbol{\mu}^t)^\top (\boldsymbol{\mu} - \boldsymbol{\mu}^t)$

compute step size α^t ▷ by Armijo Rule

$\boldsymbol{\mu}^{t+1} \leftarrow \boldsymbol{\mu}^t + \alpha^t (\bar{\boldsymbol{\mu}}^t - \boldsymbol{\mu}^t)$

if stopping criterion met **then**

break

end if

end for

Gradient Projection Method

Algorithm 2 Gradient projection method for Multiple Kernel Learning

Require: $\mu_k^1 = \frac{1}{M}\ \forall k = 1, \ldots, M$

for $t = 1, 2, \ldots$ **do**

compute $J(\boldsymbol{\mu}^t)$ and $\nabla J(\boldsymbol{\mu}^t)$

compute step size s^t ▷ by Armijo Rule

$\bar{\boldsymbol{\mu}}^t \leftarrow Proj_X(\boldsymbol{\mu}^t - s^t \nabla J(\boldsymbol{\mu}^t))$

$\boldsymbol{\mu}^{t+1} \leftarrow \bar{\boldsymbol{\mu}}^t$

if stopping criterion met **then**

break

end if

end for

Gradient Projection Method

Algorithm 3 Gradient projection method for Multiple Kernel Learning

Require: $\mu_k^1 = \frac{1}{M}\ \forall k = 1, \dots, M$

for $t = 1, 2, \dots$ **do**

compute $J(\boldsymbol{\mu}^t)$ and $\nabla J(\boldsymbol{\mu}^t)$

compute step size s^t ▷ by Armijo Rule

$\bar{\boldsymbol{\mu}}^t \leftarrow Proj_X(\boldsymbol{\mu}^t - s^t \nabla J(\boldsymbol{\mu}^t))$

$\boldsymbol{\mu}^{t+1} \leftarrow \bar{\boldsymbol{\mu}}^t$

if stopping criterion met **then**

break

end if

end for

- By theorem of Bonnans and Shapiro:
  $\frac{\partial J}{\partial \mu_k} = -\frac{1}{2} \sum_{i,j} \alpha_i^* \alpha_j^* y_i y_j K_k(x_i, x_j) \quad \forall k = 1, \dots, M.$

Outline

Experimental Result

- Modification of Armijo Rule
  - Original: step size starts from 1 for each iteration
  - Modified: step size starts from previous one

Experimental Result

- Modification of Armijo Rule
  - Original: step size starts from 1 for each iteration
  - Modified: step size starts from previous one

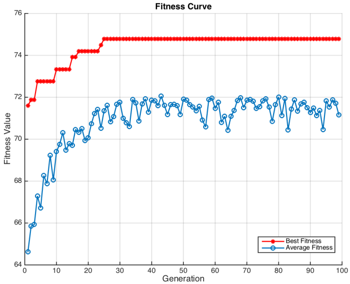
		Accuracy(%)	Time(seconds)
liver	original	63.0435	6.8091
	modification	63.0435	6.4954
ionosphere	original	92.4786	10.5235
	modification	92.4786	7.7344
sonar	original	81.9712	1.3397
	modification	81.9712	0.9920
wbc	original	96.5447	38.1509
	modification	96.5447	28.1774
heart	original	79.7778	12.9388
	modification	79.7778	5.4557

Experimental Result

- Modification of Armijo Rule
  - Original: step size starts from 1 for each iteration
  - Modified: step size starts from previous one
- Test of GA based MKL algorithm

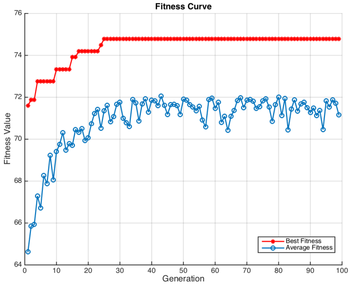
Test of GA based MKL algorithm

		Best fitness	Time*(s)
	Standard MKL	68.4058	-
liver	Single kernel	73.6232	494.226
	GA based MKL	74.4928	3370.76



Test of GA based MKL algorithm

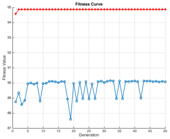
		Best fitness	Time*(s)
	Standard MKL	68.4058	-
liver	Single kernel	73.6232	494.226
	GA based MKL	74.4928	3370.76

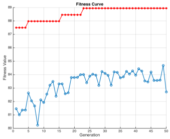


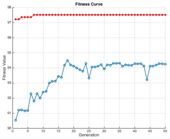
Discovery:

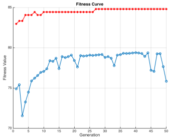
- population is in the process of evolution
- classification performance improved
- calcuation time increased

Application of GA based MKL algorithm









		Best fitness	Time*(s)
ionosphere	Standard MKL	91.7379	-
	Single kernel	94.8718	40.9043
	GA based MKL	94.8718	2022.77
sonar	Standard MKL	80.2885	-
	Single kernel	87.9808	27.4986
	GA based MKL	88.9423	410.021
wbc	Standard MKL	97.3646	-
	Single kernel	97.511	193.566
	GA based MKL	97.511	3805.23
heart	Standard MKL	82.2222	-
	Single kernel	84.4444	30.7937
	GA based MKL	84.8148	1816.41

$$\boldsymbol{\mu}_{\text{ionosphere}} = \begin{bmatrix} 0.9981 \\ 0 \\ 0.0019 \\ 0 \end{bmatrix} \quad \text{and} \quad \boldsymbol{\mu}_{\text{wbc}} = \begin{bmatrix} 0 \\ 0.0001 \\ 0.9999 \\ 0 \end{bmatrix}.$$

Outline

GA Based MKL Algorithm...

- is suitable for choosing 'good' kernels
- is free to set any types of kernels
- is highly automatic to train models
- gives higher accuracy when solving classification problems

The near future

- decrease exeucution time by improving gradient methods
- prove the convergence of modification of Armijo Rule

Thank you!